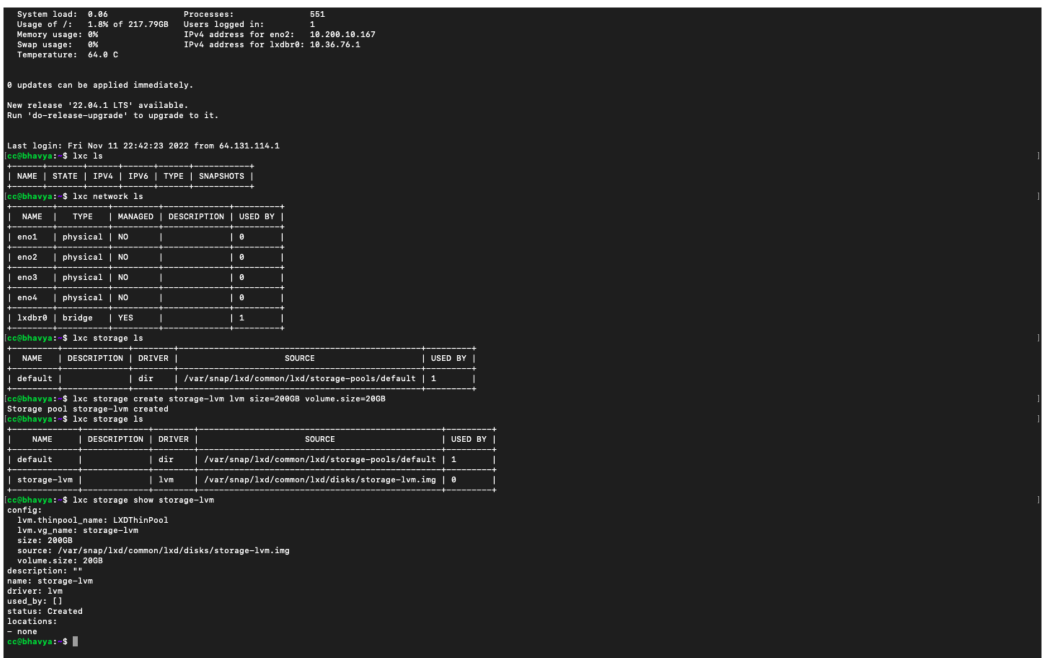
CS553 Homework #6

Sort on Hadoop/Spark

Pradyothan Govrineni (pgovrineni@hawk.iit.edu) A20438408
Bhavya Chawla (bchawla@hawk.iit.edu) A20516957

Screenshots for Hadoop File System and Hadoop Install:

>**Using lxc init**, following command to start the interactive configuration process,



> **lxc-launch** - To create and start containers from images, and lxc shell <container name> to connect to the container, doing sudo apt update && sudo apt upgrade, to download package information from all configured sources and an updated list of packages from the Internet.

```
+--------+----------+---------+-------------+---------+
| eno4   | physical | NO      |             | 0       |
+--------+----------+---------+-------------+---------+
| lxdbr0 | bridge   | YES     |             | 1       |
+--------+----------+---------+-------------+---------+
cc@bhavya:~$ lxc storage ls
+---------+-------------+--------+------------------------------------------------+---------+
|  NAME   | DESCRIPTION | DRIVER |                     SOURCE                     | USED BY |
+---------+-------------+--------+------------------------------------------------+---------+
| default |             | dir    | /var/snap/lxd/common/lxd/storage-pools/default | 1       |
+---------+-------------+--------+------------------------------------------------+---------+
cc@bhavya:~$ lxc storage create storage-lvm lvm size=200GB volume.size=20GB
Storage pool storage-lvm created
cc@bhavya:~$ lxc storage ls
+-------------+-------------+--------+------------------------------------------------+---------+
|    NAME     | DESCRIPTION | DRIVER |                     SOURCE                     | USED BY |
+-------------+-------------+--------+------------------------------------------------+---------+
| default     |             | dir    | /var/snap/lxd/common/lxd/storage-pools/default | 1       |
+-------------+-------------+--------+------------------------------------------------+---------+
| storage-lvm |             | lvm    | /var/snap/lxd/common/lxd/disks/storage-lvm.img | 0       |
+-------------+-------------+--------+------------------------------------------------+---------+
cc@bhavya:~$ lxc storage show storage-lvm
config:
  lvm.thinpool_name: LXDThinPool
  lvm.vg_name: storage-lvm
  size: 200GB
  source: /var/snap/lxd/common/lxd/disks/storage-lvm.img
  volume.size: 20GB
description: ""
name: storage-lvm
driver: lvm
used_by: []
status: Created
locations:
- none
cc@bhavya:~$ sudo pvdisplay
  --- Physical volume ---
  PV Name               /dev/loop3
  VG Name               storage-lvm
  PV Size               186.26 GiB / not usable 2.86 MiB
  Allocatable           yes (but full)
  PE Size               4.00 MiB
  Total PE              47683
  Free PE               0
  Allocated PE          47683
  PV UUID               TY0aX2-Hp7G-2O2V-e97S-gz4o-eFs9-woMuo2

cc@bhavya:~$ sudo losetup
NAME       SIZELIMIT OFFSET AUTOCLEAR RO BACK-FILE                                      DIO LOG-SEC
/dev/loop1         0      0         1  1 /var/lib/snapd/snaps/lxd_22753.snap              0     512
/dev/loop2         0      0         1  1 /var/lib/snapd/snaps/snapd_16292.snap            0     512
/dev/loop0         0      0         1  1 /var/lib/snapd/snaps/core20_1593.snap            0     512
/dev/loop3         0      0         0  0 /var/snap/lxd/common/lxd/disks/storage-lvm.img   1     512
cc@bhavya:~$ lxc launch images:ubuntu/22.04 namenode --storage storage-lvm --config limits.cpu=4 --config limits.memory=8GB
Creating namenode
Starting namenode
cc@bhavya:~$ lxc ls
+----------+---------+---------------------+------+-----------+-----------+
|   NAME   |  STATE  |        IPV4         | IPV6 |   TYPE    | SNAPSHOTS |
+----------+---------+---------------------+------+-----------+-----------+
| namenode | RUNNING | 10.36.76.180 (eth0) |      | CONTAINER | 0         |
+----------+---------+---------------------+------+-----------+-----------+
cc@bhavya:~$ lxc shell namenode
```

> Similarly Creating required Datanodes for given configuration and storage

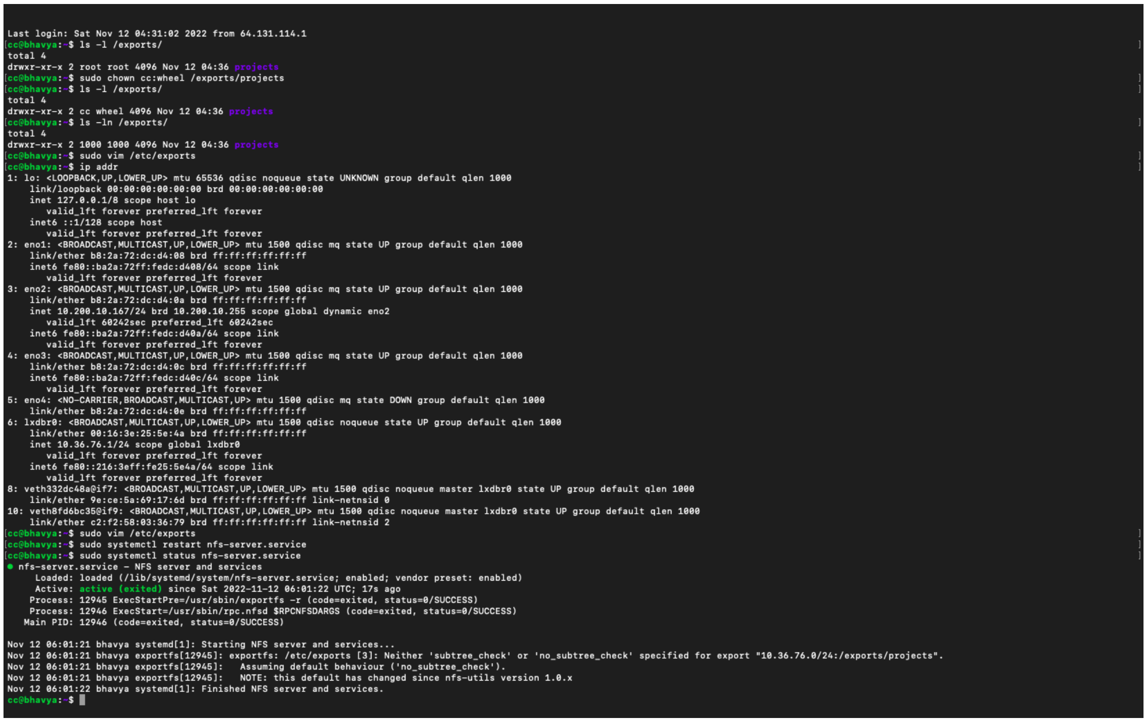
```
Get:5 http://archive.ubuntu.com/ubuntu jammy-updates/restricted amd64 Packages [417 kB]
Fetched 1,340 kB in 1s (1,478 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
All packages are up to date.
root@namenode:~# apt install htop
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libnl-3-200 libnl-genl-3-200
Suggested packages:
  lm-sensors lsof strace
The following NEW packages will be installed:
  htop libnl-3-200 libnl-genl-3-200
0 upgraded, 3 newly installed, 0 to remove and 0 not upgraded.
Need to get 200 kB of archives.
After this operation, 589 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://archive.ubuntu.com/ubuntu jammy/main amd64 libnl-3-200 amd64 3.5.0-0.1 [59.1 kB]
Get:2 http://archive.ubuntu.com/ubuntu jammy/main amd64 libnl-genl-3-200 amd64 3.5.0-0.1 [12.4 kB]
Get:3 http://archive.ubuntu.com/ubuntu jammy/main amd64 htop amd64 3.0.5-7build2 [128 kB]
Fetched 200 kB in 0s (1,029 kB/s)
Selecting previously unselected package libnl-3-200:amd64.
(Reading database ... 16130 files and directories currently installed.)
Preparing to unpack .../libnl-3-200_3.5.0-0.1_amd64.deb ...
Unpacking libnl-3-200:amd64 (3.5.0-0.1) ...
Selecting previously unselected package libnl-genl-3-200:amd64.
Preparing to unpack .../libnl-genl-3-200_3.5.0-0.1_amd64.deb ...
Unpacking libnl-genl-3-200:amd64 (3.5.0-0.1) ...
Selecting previously unselected package htop.
Preparing to unpack .../htop_3.0.5-7build2_amd64.deb ...
Unpacking htop (3.0.5-7build2) ...
Setting up libnl-3-200:amd64 (3.5.0-0.1) ...
Setting up libnl-genl-3-200:amd64 (3.5.0-0.1) ...
Setting up htop (3.0.5-7build2) ...
Processing triggers for libc-bin (2.35-0ubuntu3.1) ...
root@namenode:~# df -h
Filesystem                            Size  Used Avail Use% Mounted on
/dev/storage-lvm/containers_namenode   19G  586M   17G   4% /
none                                  492K  4.0K  488K   1% /dev
tmpfs                                 100K     0  100K   0% /dev/lxd
tmpfs                                 100K     0  100K   0% /dev/.lxd-mounts
tmpfs                                  63G     0   63G   0% /dev/shm
tmpfs                                  26G  120K   26G   1% /run
tmpfs                                 5.0M     0  5.0M   0% /run/lock
tmpfs                                 4.0M     0  4.0M   0% /sys/fs/cgroup
tmpfs                                 763M     0  763M   0% /run/user/0
root@namenode:~# exit
logout
cc@bhavya:~$ lxc launch images:ubuntu/22.04 datanode1 --storage storage-lvm --config limits.cpu=4 --config limits.memory=8GB
Creating datanode1
Starting datanode1
cc@bhavya:~$ lxc ls
+-----------+---------+---------------------+------+-----------+-----------+
|   NAME    |  STATE  |        IPV4         | IPV6 |   TYPE    | SNAPSHOTS |
+-----------+---------+---------------------+------+-----------+-----------+
| datanode1 | RUNNING | 10.36.76.218 (eth0) |      | CONTAINER | 0         |
+-----------+---------+---------------------+------+-----------+-----------+
| namenode  | RUNNING | 10.36.76.180 (eth0) |      | CONTAINER | 0         |
+-----------+---------+---------------------+------+-----------+-----------+
cc@bhavya:~$
```

Following the Tutorial for 2. NFS Setup and 3. Hadoop

sudo apt install nfs-kernel-server

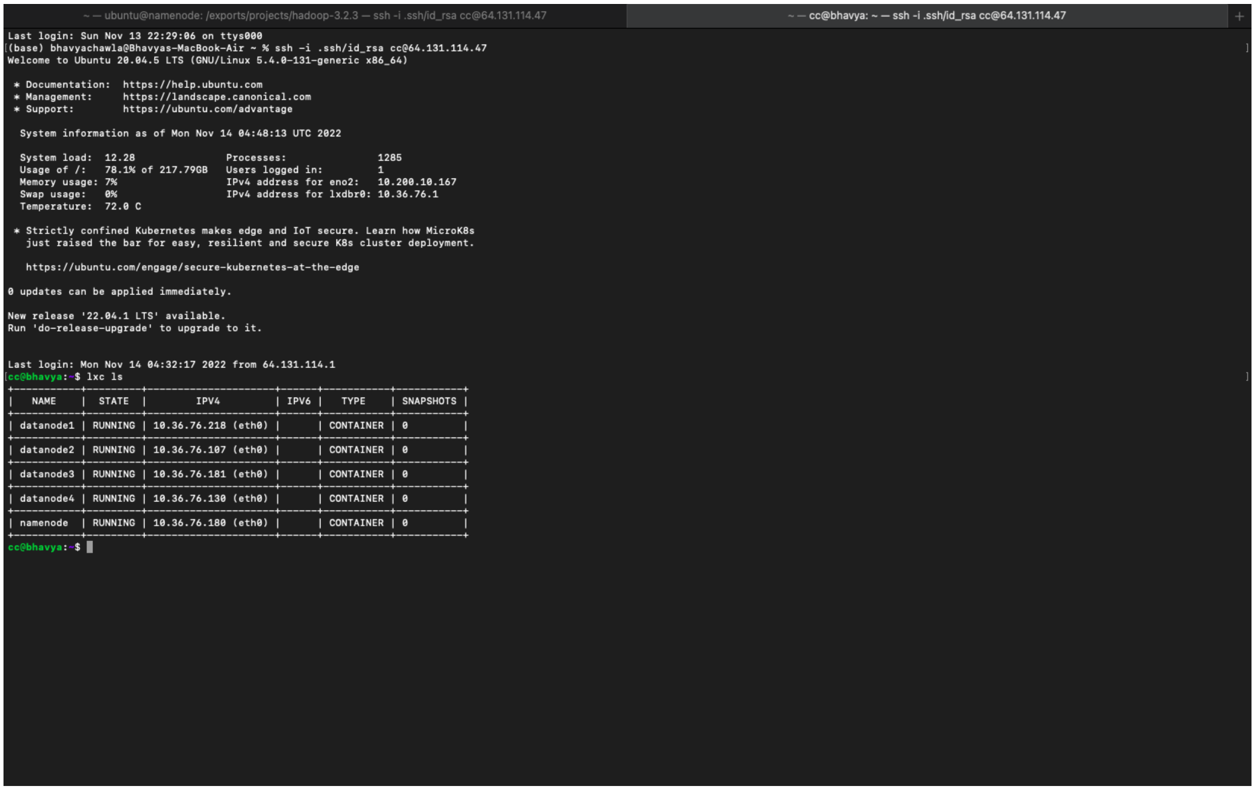
sudo apt install nfs-common

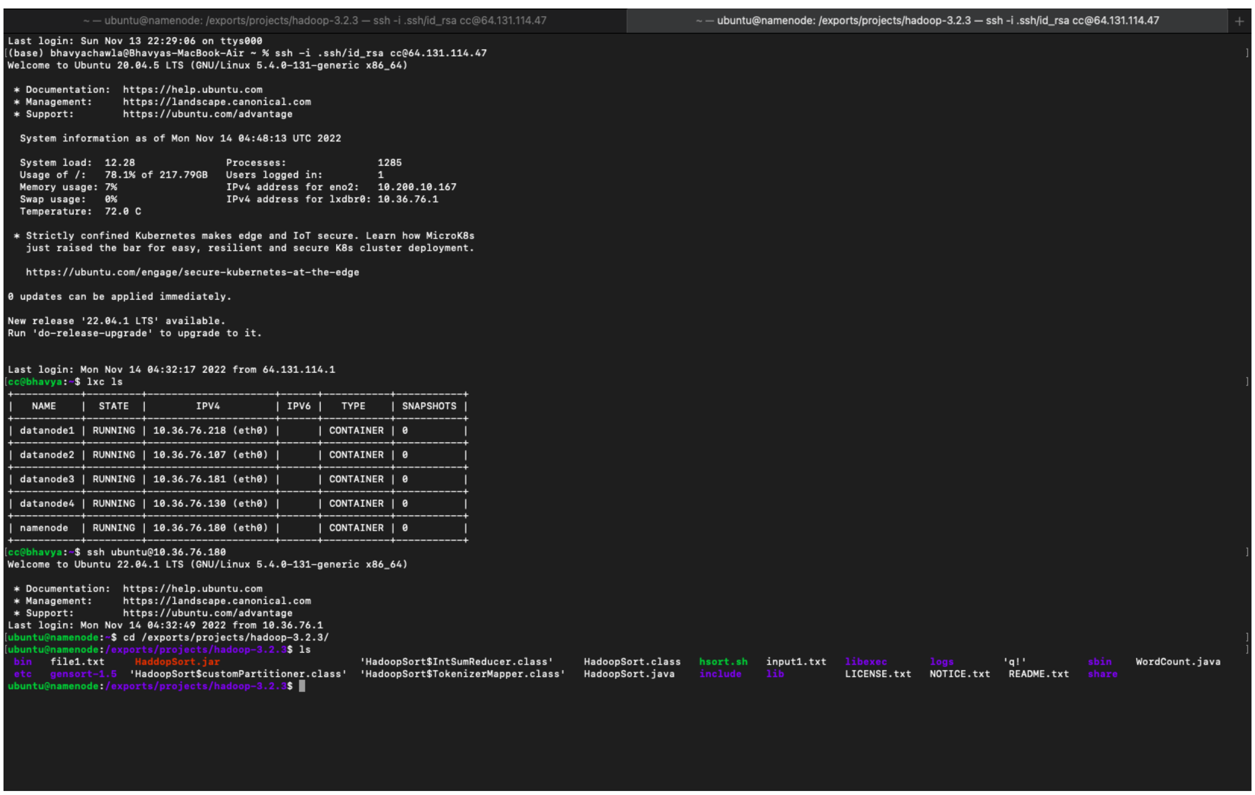
1. The Hadoop framework (Index of /hadoop/common/hadoop-3.2.3)
2. The Spark framework (https://archive.apache.org/dist/spark/spark-3.0.0-preview2)



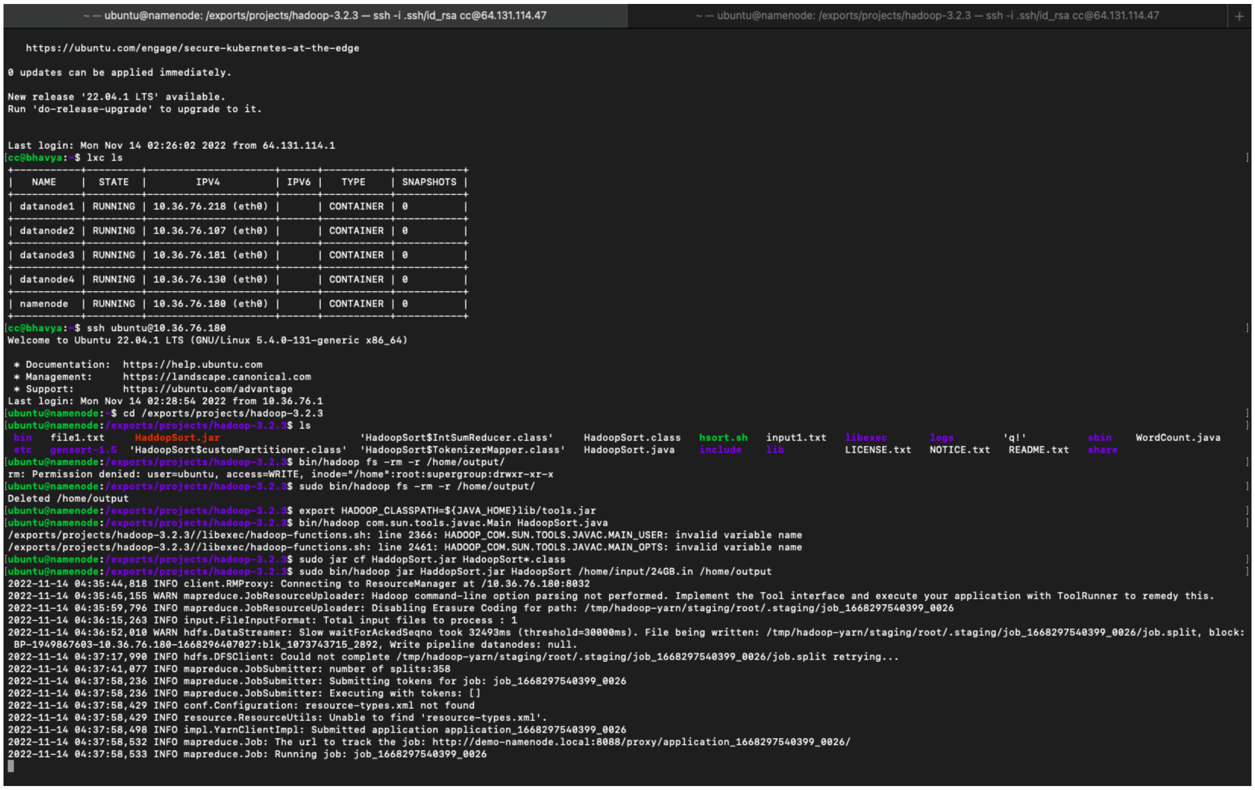
```
Last login: Sat Nov 12 04:31:02 2022 from 64.131.114.1
cc@bhavya:~$ ls -l /exports/
total 4
drwxr-xr-x 2 root root 4096 Nov 12 04:36 projects
cc@bhavya:~$ sudo chown cc:wheel /exports/projects
cc@bhavya:~$ ls -l /exports/
total 4
drwxr-xr-x 2 cc wheel 4096 Nov 12 04:36 projects
cc@bhavya:~$ ls -ln /exports/
total 4
drwxr-xr-x 2 1000 1000 4096 Nov 12 04:36 projects
cc@bhavya:~$ sudo vim /etc/exports
cc@bhavya:~$ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eno1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group default qlen 1000
    link/ether b8:2a:72:dc:d4:08 brd ff:ff:ff:ff:ff:ff
    inet6 fe80::ba2a:72ff:fedc:d408/64 scope link
       valid_lft forever preferred_lft forever
3: eno2: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group default qlen 1000
    link/ether b8:2a:72:dc:d4:0a brd ff:ff:ff:ff:ff:ff
    inet 10.200.10.167/24 brd 10.200.10.255 scope global dynamic eno2
       valid_lft 60242sec preferred_lft 60242sec
    inet6 fe80::ba2a:72ff:fedc:d40a/64 scope link
       valid_lft forever preferred_lft forever
4: eno3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group default qlen 1000
    link/ether b8:2a:72:dc:d4:0c brd ff:ff:ff:ff:ff:ff
    inet6 fe80::ba2a:72ff:fedc:d40c/64 scope link
       valid_lft forever preferred_lft forever
5: eno4: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc mq state DOWN group default qlen 1000
    link/ether b8:2a:72:dc:d4:0e brd ff:ff:ff:ff:ff:ff
6: lxdbr0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default qlen 1000
    link/ether 00:16:3e:25:5e:4a brd ff:ff:ff:ff:ff:ff
    inet 10.36.76.1/24 scope global lxdbr0
       valid_lft forever preferred_lft forever
    inet6 fe80::216:3eff:fe25:5e4a/64 scope link
       valid_lft forever preferred_lft forever
8: veth332dc48a@if7: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue master lxdbr0 state UP group default qlen 1000
    link/ether 9e:ce:5a:69:17:6d brd ff:ff:ff:ff:ff:ff link-netnsid 0
10: veth8fd6bc35@if9: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue master lxdbr0 state UP group default qlen 1000
    link/ether c2:f2:58:03:36:79 brd ff:ff:ff:ff:ff:ff link-netnsid 2
cc@bhavya:~$ sudo vim /etc/exports
cc@bhavya:~$ sudo systemctl restart nfs-server.service
cc@bhavya:~$ sudo systemctl status nfs-server.service
● nfs-server.service - NFS server and services
     Loaded: loaded (/lib/systemd/system/nfs-server.service; enabled; vendor preset: enabled)
     Active: active (exited) since Sat 2022-11-12 06:01:22 UTC; 17s ago
    Process: 12945 ExecStartPre=/usr/sbin/exportfs -r (code=exited, status=0/SUCCESS)
    Process: 12946 ExecStart=/usr/sbin/rpc.nfsd $RPCNFSDARGS (code=exited, status=0/SUCCESS)
   Main PID: 12946 (code=exited, status=0/SUCCESS)

Nov 12 06:01:21 bhavya systemd[1]: Starting NFS server and services...
Nov 12 06:01:21 bhavya exportfs[12945]: exportfs: /etc/exports [3]: Neither 'subtree_check' or 'no_subtree_check' specified for export "10.36.76.0/24:/exports/projects".
Nov 12 06:01:21 bhavya exportfs[12945]:   Assuming default behaviour ('no_subtree_check').
Nov 12 06:01:21 bhavya exportfs[12945]:   NOTE: this default has changed since nfs-utils version 1.0.x
Nov 12 06:01:22 bhavya systemd[1]: Finished NFS server and services.
cc@bhavya:~$
```

For example setup for 4 Small config.





For running Hadoop Sort,



```
~ — ubuntu@namenode: /exports/projects/hadoop-3.2.3 — ssh -i .ssh/id_rsa cc@64.131.114.47

   https://ubuntu.com/engage/secure-kubernetes-at-the-edge

0 updates can be applied immediately.

New release '22.04.1 LTS' available.
Run 'do-release-upgrade' to upgrade to it.


Last login: Mon Nov 14 02:26:02 2022 from 64.131.114.1
cc@bhavya:~$ lxc ls
+-----------+---------+----------------------+------+-----------+-----------+
|   NAME    |  STATE  |         IPV4         | IPV6 |   TYPE    | SNAPSHOTS |
+-----------+---------+----------------------+------+-----------+-----------+
| datanode1 | RUNNING | 10.36.76.218 (eth0)  |      | CONTAINER | 0         |
+-----------+---------+----------------------+------+-----------+-----------+
| datanode2 | RUNNING | 10.36.76.107 (eth0)  |      | CONTAINER | 0         |
+-----------+---------+----------------------+------+-----------+-----------+
| datanode3 | RUNNING | 10.36.76.181 (eth0)  |      | CONTAINER | 0         |
+-----------+---------+----------------------+------+-----------+-----------+
| datanode4 | RUNNING | 10.36.76.130 (eth0)  |      | CONTAINER | 0         |
+-----------+---------+----------------------+------+-----------+-----------+
| namenode  | RUNNING | 10.36.76.180 (eth0)  |      | CONTAINER | 0         |
+-----------+---------+----------------------+------+-----------+-----------+
cc@bhavya:~$ ssh ubuntu@10.36.76.180
Welcome to Ubuntu 22.04.1 LTS (GNU/Linux 5.4.0-131-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage
Last login: Mon Nov 14 02:28:54 2022 from 10.36.76.1
ubuntu@namenode:~$ cd /exports/projects/hadoop-3.2.3
ubuntu@namenode:/exports/projects/hadoop-3.2.3$ ls
 bin   file1.txt     HaddopSort.jar                       'HadoopSort$IntSumReducer.class'     HadoopSort.class   hsort.sh   input1.txt   libexec      logs        'q!'        sbin    WordCount.java
 etc   gensort-1.5  'HadoopSort$customPartitioner.class'  'HadoopSort$TokenizerMapper.class'   HadoopSort.java    include    lib          LICENSE.txt  NOTICE.txt   README.txt  share
ubuntu@namenode:/exports/projects/hadoop-3.2.3$ bin/hadoop fs -rm -r /home/output/
rm: Permission denied: user=ubuntu, access=WRITE, inode="/home":root:supergroup:drwxr-xr-x
ubuntu@namenode:/exports/projects/hadoop-3.2.3$ sudo bin/hadoop fs -rm -r /home/output/
Deleted /home/output
ubuntu@namenode:/exports/projects/hadoop-3.2.3$ export HADOOP_CLASSPATH=${JAVA_HOME}lib/tools.jar
ubuntu@namenode:/exports/projects/hadoop-3.2.3$ bin/hadoop com.sun.tools.javac.Main HadoopSort.java
/exports/projects/hadoop-3.2.3//libexec/hadoop-functions.sh: line 2366: HADOOP_COM.SUN.TOOLS.JAVAC.MAIN_USER: invalid variable name
/exports/projects/hadoop-3.2.3//libexec/hadoop-functions.sh: line 2461: HADOOP_COM.SUN.TOOLS.JAVAC.MAIN_OPTS: invalid variable name
ubuntu@namenode:/exports/projects/hadoop-3.2.3$ sudo jar cf HaddopSort.jar HadoopSort*.class
ubuntu@namenode:/exports/projects/hadoop-3.2.3$ sudo bin/hadoop jar HaddopSort.jar HadoopSort /home/input/24GB.in /home/output
2022-11-14 04:35:44,818 INFO client.RMProxy: Connecting to ResourceManager at /10.36.76.180:8032
2022-11-14 04:35:45,155 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this.
2022-11-14 04:35:59,796 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/root/.staging/job_1668297540399_0026
2022-11-14 04:36:15,263 INFO input.FileInputFormat: Total input files to process : 1
2022-11-14 04:36:52,010 WARN hdfs.DataStreamer: Slow waitForAckedSeqno took 32493ms (threshold=30000ms). File being written: /tmp/hadoop-yarn/staging/root/.staging/job_1668297540399_0026/job.split, block:
 BP-1949867603-10.36.76.180-1668296407027:blk_1073743715_2892, Write pipeline datanodes: null.
2022-11-14 04:37:17,990 INFO hdfs.DFSClient: Could not complete /tmp/hadoop-yarn/staging/root/.staging/job_1668297540399_0026/job.split retrying...
2022-11-14 04:37:41,077 INFO mapreduce.JobSubmitter: number of splits:358
2022-11-14 04:37:58,236 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1668297540399_0026
2022-11-14 04:37:58,236 INFO mapreduce.JobSubmitter: Executing with tokens: []
2022-11-14 04:37:58,429 INFO conf.Configuration: resource-types.xml not found
2022-11-14 04:37:58,429 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
2022-11-14 04:37:58,498 INFO impl.YarnClientImpl: Submitted application application_1668297540399_0026
2022-11-14 04:37:58,532 INFO mapreduce.Job: The url to track the job: http://demo-namenode.local:8088/proxy/application_1668297540399_0026/
2022-11-14 04:37:58,533 INFO mapreduce.Job: Running job: job_1668297540399_0026
```

1. How many threads, mappers, reducers, you used in each experiment?

   Ans. The mapper used for spark and hadoop was maptopair function which takes a key to map between 0 to 9 and a to z. The reducers were reducebykey, which utilize an associative and commutative reduction function to combine the values for each key. Also 48 threads were used in each experiment.

2. How many times did you have to read and write the dataset for each experiment?

   Ans.   Internal sorting was utilized for MySort and Linuxsort when the data set fit in the RAM; this only required a single read and write of the dataset. When the dataset, however, cannot fit in memory, we utilize external sort, which requires two reads and two writes. Hadoop requires input to be read twice and written twice. The first read occurs at the beginning of a map task, the second read occurs by reducers during the shuffling phase, and the first write occurs at the conclusion of a map task, the third write occurs following the reducers task. Instead of storing interim findings on disk, Spark saves them in memory. Spark will save the interim findings on the disk in cases when the file size is greater than the RAM.

3. What speedup and efficiency did you achieve?

   Ans. For the speed and efficiency, instances in spark go at a faster rate than hadoop, even though the workers are comparable between each other. This is because Spark saves the data in memory before writing while hadoop straight away writes the data into sorting.

4. Conclusions?Which seems to be best at 1 node scale (1 large.instance)?Is there a difference between 1 small.instance and 1 large.instance? How about 4 nodes (4 small.instance)?

   Ans. One large instance performs better than one small instance at a size of one node because it has greater memory and core power. Additionally, as the magnitude of the data grows, this is increasingly obvious. Hadoop sort and Spark sort outperform each other for the same dataset at 4 nodes. This is due to the fact that there are more workers available for the master node to delegate jobs to concurrently. Linux and shared memory sort do not apply to these 4 nodes because they run on a single node. Although it may appear that Linux and MySort perform quicker than Hadoop and Spark, this is because the overhead costs are low and the dataset size is modest.

5. What speedup do you achieve with strong scaling between 1 to 4 nodes? What speedup do you achieve with weak scaling between 1 to 4 nodes?

   Ans. Hadoop behaved quite differently when scaling strongly from 3GB on 1small to 3GB on 1large. On a single big instance, it is quicker. Additionally, while examining different datasets, Hadoop's speed appeared to improve by at least two times. Hadoop sort speed was enhanced by the poor scalability, where the example from 3GB on 1 tiny instance was 38921 seconds. Weak scalability was seen with 12 GB on 4 tiny instances at 2993987.

6. How many small.instances do you need with Hadoop to achieve the same level of performance as your shared memory sort?

   Ans: We need about 8 small instances with Hadoop to achieve the same level of performance as shared memory sort.

7. How about how many small.instances do you need with Spark to achieve the same level of performance as you did with your shared memory sort?

   Ans: We need approximately 1.21 times in spark to achieve the same performance as memory sort

8. Can you predict which would be best if you had 100 small.instances? How about 1000?

   Ans: With 100 tiny instances, it is possible to sort very big datasets efficiently and economically. Although 1000 instances can increase performance, there are other problems with such a huge number of nodes, like scheduling and synchronization. Without an effective management structure, this can lead to new issues and produce unexpected outcomes. Additionally, having such a high number of instances offers redundancy since even if the likelihood of a failure rises with the number of nodes, one node being down has little impact on the system's performance.

9. Compare your results with those from the Sort Benchmark (http://sortbenchmark.org), specifically the winners in 2013 and 2014 who used Hadoop and Spark. Also, what can you learn from the CloudSort benchmark, a report can be found at (http://sortbenchmark.org/2014_06_CloudSort_v_0_4.pdf).

   Ans: Winners from 2013 and 2014 sorted at a rate of over 1.4 & 4.3 TB/sec. Hadoop was used in 2013 whereas Spark was used in 2014.

   If Spark has enough memory and all the parameters are set, it will perform better as the size of the datasets grows.

   Similarly results show that Hadoop and Spark initially perform better for small datasets, but as the size grows, so does the performance gap between them.

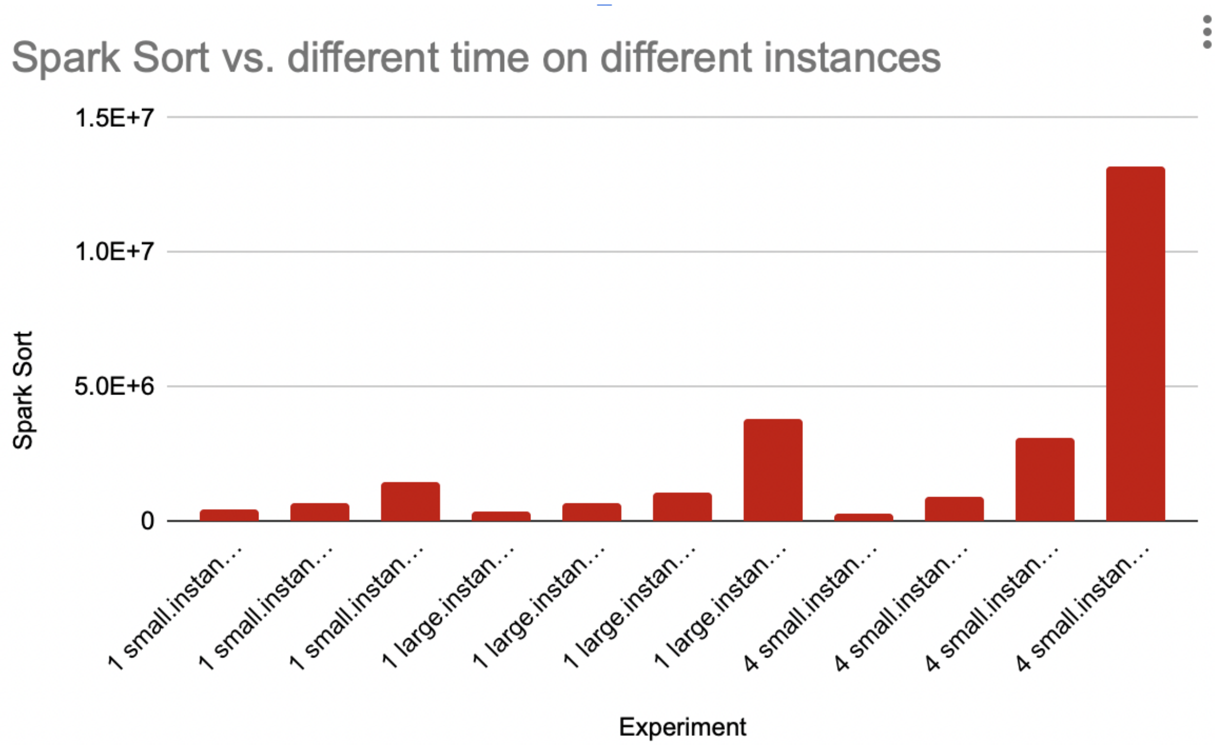
   Due to lower disk I/Os, Spark performs better than Hadoop because it needs anywhere between 2 and 3 times as much memory.

   In the study, a new method for external sort benchmarking is proposed called CloudSort. From the standpoint of Total Cost of Ownership, CloudSort compares external sort.

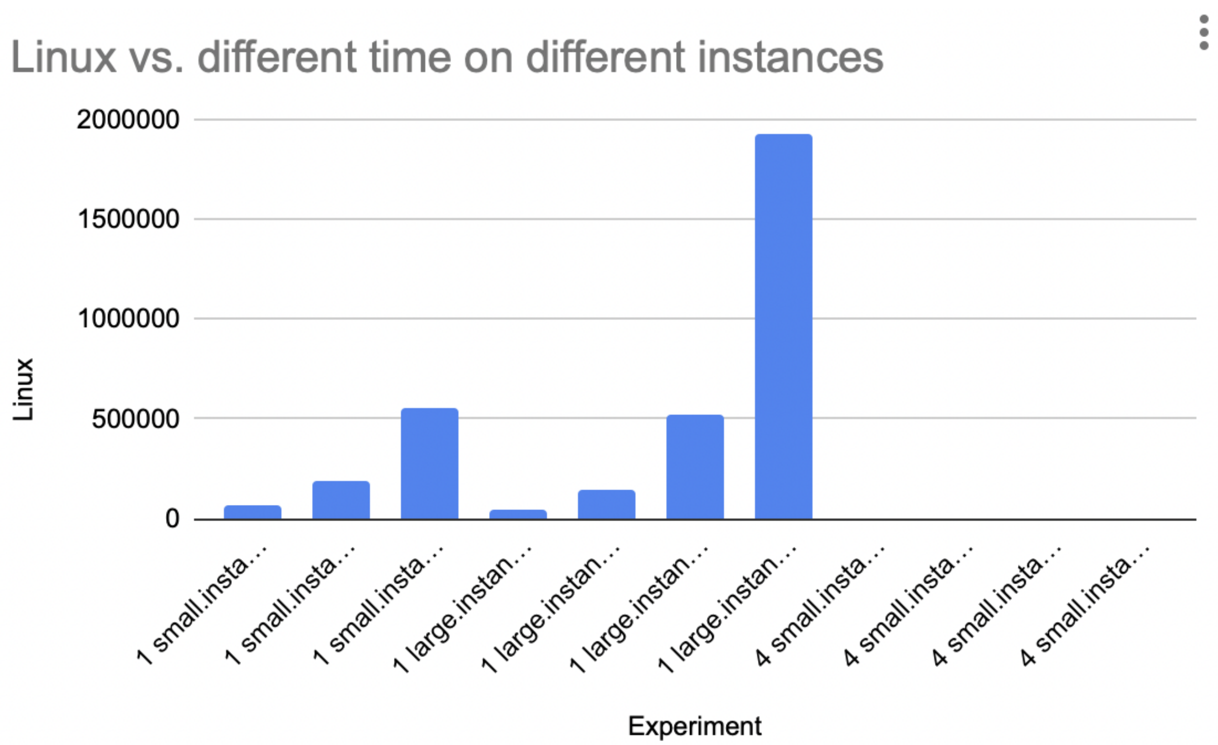
   the assumption that the present benchmarking techniques ignore data center administration and maintenance expenses.

   Because cloud storage costs already in management and maintenance costs, they are perfect for this and provide a complete TCO.

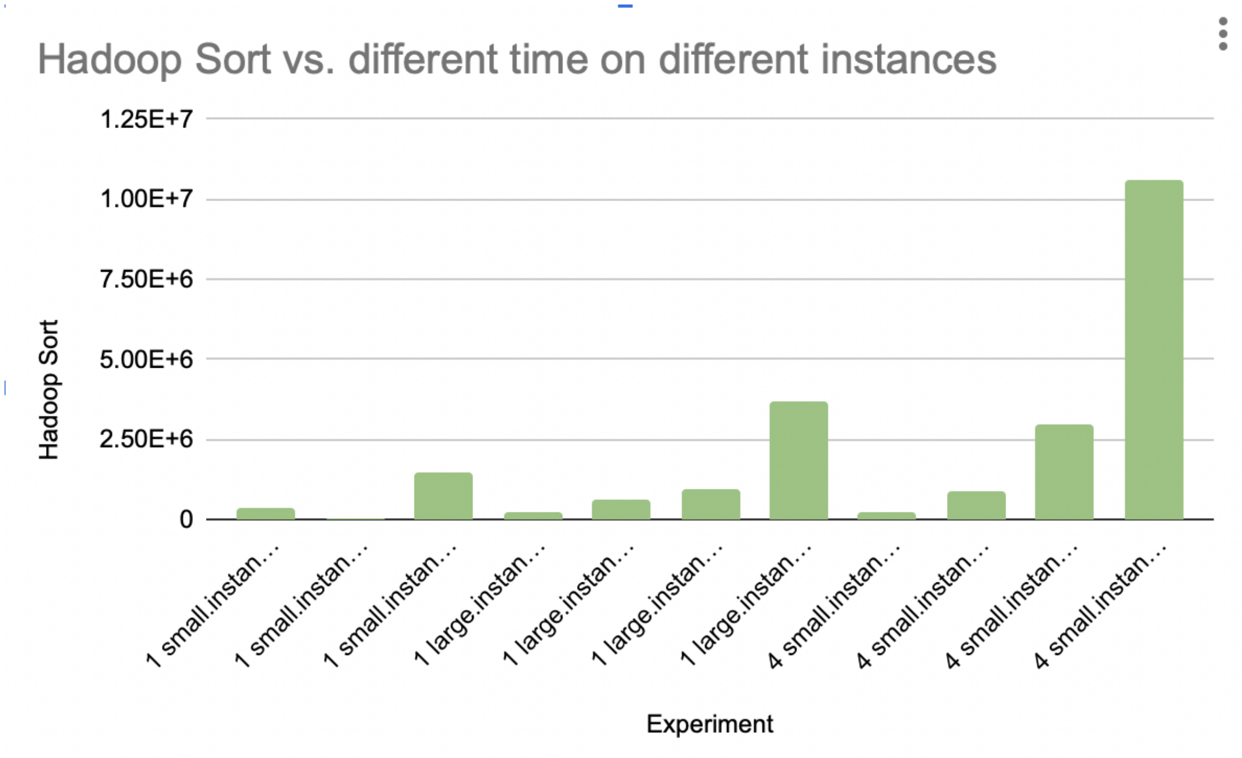
   They suggest using public clouds for running external sort. Since external sort is indicative of many IO-intensive workloads, they utilize it as the sorting method to benchmark. It is simple and straightforward to adapt to new technologies, and it covers all aspects including RAM, CPU, OS, file system, IO network, and storage.



Spark Sort vs. different time on different instances
1.5E+7
1.0E+7
5.0E+6
0
Spark Sort
1 small.instan…
1 small.instan…
1 small.instan…
1 large.instan…
1 large.instan…
1 large.instan…
1 large.instan…
4 small.instan…
4 small.instan…
4 small.instan…
4 small.instan…
Experiment



Linux vs. different time on different instances
2000000
1500000
1000000
500000
0
Linux
1 small.insta…
1 small.insta…
1 small.insta…
1 large.instan…
1 large.instan…
1 large.instan…
1 large.instan…
4 small.insta…
4 small.insta…
4 small.insta…
4 small.insta…
Experiment



Experiment	Linux	Hadoop Sort	Spark Sort
1 small.instance, 3 GB dataset	68839	389021	458102
1 small.instance, 6 GB dataset	189872	62168	672901
1 small.instance, 12 GB dataset	550002	1500023	1418762
1 large.instance, 3 GB dataset	45122	247089	349241
1 large.instance, 6 GB dataset	146990	626976	699690
1 large.instance, 12 GB dataset	519890	942987	1054982
1 large.instance, 24 GB dataset	1932102	3699283	3819872
4 small.instances, 3 GB dataset	N/A	231987	289021

4 small.instances, 6 GB dataset	N/A	861987	928092
4 small.instances, 12 GB dataset	N/A	2993987	3098762
4 small.instances, 24 GB dataset	N/A	10598973	13196928

experiment	shared memory	linux	hadoop sort	spark sort
1 small instance, 3 Gb dataset	1 thread/8GB memory	8 GB memory, 48 threads	1 reducer	driver memory 4 GB, exxecutory memory 2GB, executor core 1
1 small instance, 6 Gb dataset	1 thread/8GB memory	9 GB memory, 48 threads	2 reducers	driver memory 4 GB, exxecutory memory 2GB, executor core 1
1 small instance, 12 Gb dataset	48 thread/8GB memory	10 GB memory, 48 threads	2 reducers	driver memory 4 GB, exxecutory memory 2GB, executor core 1
1 large instance, 3 Gb dataset	1 thread/8GB memory	11 GB memory, 48 threads	2 reducers	driver memory 4 GB, exxecutory memory 2GB, executor core 1
1 large instance, 6 Gb dataset	1 thread/8GB memory	12 GB memory, 48 threads	2 reducers	driver memory 4 GB, exxecutory

				memory 7GB, executor core 5
1 large instance, 12 Gb dataset	48 thread/8GB memory	13 GB memory, 48 threads	2 reducers	driver memory 4 GB, exxecutory memory 6GB, executor core 2
1 large instance, 24 Gb dataset	48 thread/8GB memory	14 GB memory, 48 threads	1 reducers	driver memory 4 GB, exxecutory memory 2GB, executor core 1
4 small instances, 3 GB dataset	N/A	N/A	7 reducers	driver memory 4 GB, exxecutory memory 2GB, executor core 1
4 small instances, 6 GB dataset	N/A	N/A	4 reducers	driver memory 4 GB, exxecutory memory 2GB, executor core 1
4 small instances, 12 GB dataset	N/A	N/A	7 reducers	driver memory 4 GB, exxecutory memory 2GB, executor core 1
4 small instances, 24 GB dataset	N/A	N/A	7 reducers	driver memory 4 GB, exxecutory memory 2GB, executor core 1