IERG4330/IEMS5730 Spring 2022
Homework 3

Release date: Mar 11, 2022

Due date: 11:59:00 pm, Mar 28, 2022

The solution will be posted right after the deadline, so no late homework will be accepted!

Every Student MUST include the following statement, together with his/her signature in the submitted homework.

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

Signed (Student Eddie Christopher Fox III) Date: March 28, 2022

Name Eddie Christopher Fox III SID 1155160788

Submission notice:

- Submit your homework via the elearning system

General homework policies:

A student may discuss the problems with others. However, the work a student turns in must be created COMPLETELY by oneself ALONE. A student may not share ANY written work or pictures, nor may one copy answers from any source other than one's own brain.

Each student **MUST LIST** on the homework paper the **name of every person he/she has discussed or worked with**. If the answer includes content from any other source, the student **MUST STATE THE SOURCE**. Failure to do so is cheating and will result in sanctions. Copying answers from someone else is cheating even if one lists their name(s) on the homework.

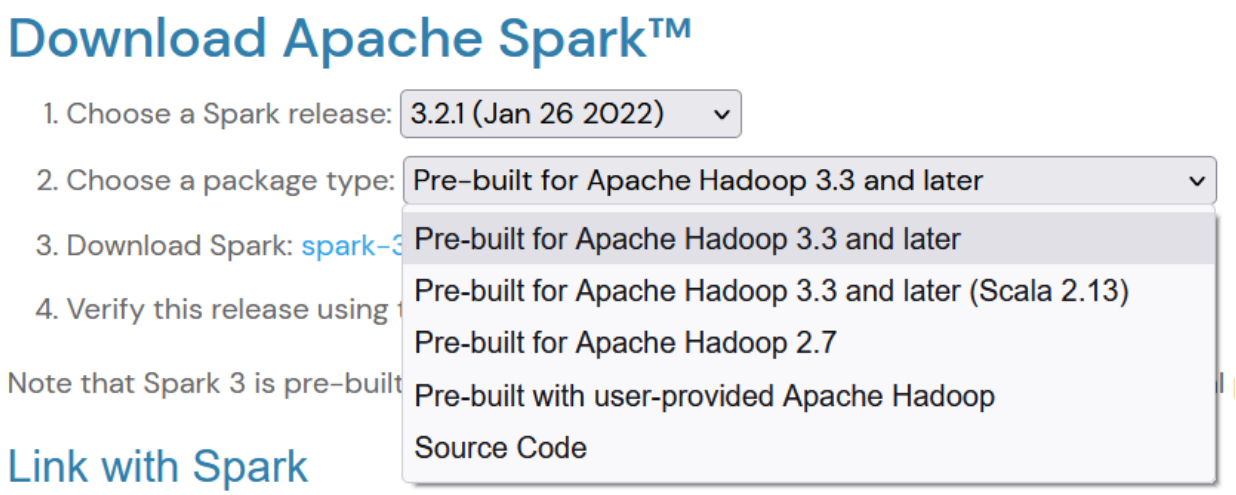
If there is information you need to solve a problem but the information is not stated in the problem, try to find the data somewhere. If you cannot find it, state what data you need, make a reasonable estimate of its value, and justify any assumptions you make. You will be graded not only on whether your answer is correct, but also on whether you have done an intelligent analysis.

IEMS 5730: Spring 2022 Assignment #3

Question 1 Section A: Setup of standalone Spark Cluster

I first start my four VM's back up and update the external IP addresses in Putty before logging back into them, then start up dfs and YARN using the start-dfs.sh and start-yarn.sh in the /usr/local/hadoop/sbin folder of the master node. I then confirm through JPS that everything is working as it should be, before running the installation commands.

According to [1], to install Spark in a standalone manner, you simply place a compiled version of each Spark on each node. From the download page [2], there are a number of possible downloads.



I decide to choose Spark 3.2.1 with Pre-build for Apache Hadoop 2.7. Even though my Hadoop is 2.9.2, it should be safer to use one built for an earlier Hadoop version. It is not worth risking a version pre-build for a later version of Hadoop.

URL for Spark 3.2.1 prebuilt for Apache Hadoop 2.7:

https://dlcdn.apache.org/spark/spark-3.2.1/spark-3.2.1-bin-hadoop2.7.tgz

[3] suggests installing Scala before Spark, so I do this on all nodes using "sudo apt-get install scala -y" on every node [4].

```
EddieCFox@master:/usr/local/hadoop$ scala -version
Scala code runner version 2.11.12 -- Copyright 2002-2017, LAMP/EPFL
EddieCFox@master:/usr/local/hadoop$ java -version
openjdk version "1.8.0_312"
OpenJDK Runtime Environment (build 1.8.0_312-8u312-b07-0ubuntu1~18.04-b07
OpenJDK 64-Bit Server VM (build 25.312-b07, mixed mode)
EddieCFox@master:/usr/local/hadoop$
```

I can confirm afterwards using version commands that Scala 2.11.12 and Java 1.8.0_312 (Java 8) are installed on all four nodes.

I navigate to the home directory of each node, where I traditionally dump the raw files for every installation, then run the following commands on each one.

“sudo wget https://dlcdn.apache.org/spark/spark-3.2.1/spark-3.2.1-bin-hadoop2.7.tgz”

```
EddieCFox@master:~$ sudo wget https://dlcdn.apache.org/spark/spark-3.2.1/spark-3.2.1-bin-hadoop2.7.tgz
--2022-03-21 09:05:04--  https://dlcdn.apache.org/spark/spark-3.2.1/spark-3.2.1-bin-hadoop2.7.tgz
Resolving dlcdn.apache.org (dlcdn.apache.org)... 151.101.2.132, 2a04:4e42::644
Connecting to dlcdn.apache.org (dlcdn.apache.org)|151.101.2.132|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 272637746 (260M) [application/x-gzip]
Saving to: ‘spark-3.2.1-bin-hadoop2.7.tgz’

spark-3.2.1-bin-hadoop2.7.tgz                   100%[===================================================================================================>] 260.01M   220MB/s    in 1.2s

2022-03-21 09:05:06 (220 MB/s) - ‘spark-3.2.1-bin-hadoop2.7.tgz’ saved [272637746/272637746]

EddieCFox@master:~$ ls
apache-hive-2.3.8-bin  apache-hive-2.3.8-bin.tar.gz  hadoop-2.9.2.tar.gz  pig-0.17.0  pig-0.17.0.tar.gz  spark-3.2.1-bin-hadoop2.7.tgz
EddieCFox@master:~$
```

“tar -xvf spark-3.2.1-bin-hadoop2.7.tgz”

```
spark-3.2.1-bin-hadoop2.7/sbin/stop-thriftserver.sh
spark-3.2.1-bin-hadoop2.7/sbin/stop-worker.sh
spark-3.2.1-bin-hadoop2.7/sbin/stop-workers.sh
spark-3.2.1-bin-hadoop2.7/sbin/workers.sh
spark-3.2.1-bin-hadoop2.7/yarn/
spark-3.2.1-bin-hadoop2.7/yarn/spark-3.2.1-yarn-shuffle.jar
EddieCFox@master:~$ ls
apache-hive-2.3.8-bin  apache-hive-2.3.8-bin.tar.gz  hadoop-2.9.2.tar.gz  pig-0.17.0  pig-0.17.0.tar.gz  spark-3.2.1-bin-hadoop2.7  spark-3.2.1-bin-hadoop2.7.tgz
EddieCFox@master:~$ cd spark-3.2.1-bin-hadoop2.7/
EddieCFox@master:~/spark-3.2.1-bin-hadoop2.7$ ls
LICENSE  NOTICE  R  README.md  RELEASE  bin  conf  data  examples  jars  kubernetes  licenses  python  sbin  yarn
EddieCFox@master:~/spark-3.2.1-bin-hadoop2.7$ cd bin
EddieCFox@master:~/spark-3.2.1-bin-hadoop2.7/bin$ ls
beeline              find-spark-home      load-spark-env.sh  pyspark2.cmd     spark-class      spark-shell      spark-sql       spark-submit       sparkR
beeline.cmd          find-spark-home.cmd  pyspark            run-example      spark-class.cmd  spark-shell.cmd  spark-sql.cmd   spark-submit.cmd   sparkR.cmd
docker-image-tool.sh load-spark-env.cmd   pyspark.cmd        run-example.cmd  spark-class2.cmd spark-shell2.cmd spark-sql2.cmd  spark-submit2.cmd  sparkR2.cmd
EddieCFox@master:~/spark-3.2.1-bin-hadoop2.7/bin$
```

After extracting the spark folder, it creates a “spark-3.2.1-bin-hadoop2.7” folder that itself has various folders, including commands within a bin folder.

```
EddieCFox@master:~$ sudo mv spark-3.2.1-bin-hadoop2.7 /usr/local/spark
EddieCFox@master:~$ cd /usr/local
EddieCFox@master:/usr/local$ ls
bin  etc  games  hadoop  include  lib  man  sbin  share  spark  src
EddieCFox@master:/usr/local$ cd spark
EddieCFox@master:/usr/local/spark$ ls
LICENSE  NOTICE  R  README.md  RELEASE  bin  conf  data  examples  jars  kubernetes  licenses  python  sbin  yarn
EddieCFox@master:/usr/local/spark$
```

Per [4], I use “sudo mv spark-3.2.1-bin-hadoop2.7 /usr/local/spark” to move the extracted Spark download folder into /usr/local as a new directory named Spark, then navigate to /usr/local/spark to confirm the contents are the same as the extracted download.

```
export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
export PATH=/home/EddieCFox/pig-0.17.0/bin:$PATH
export PIG_CLASSPATH=/usr/local/hadoop/etc/hadoop
export PATH=$PATH:/usr/local/spark/bin
```

As suggested in [4], I add the bin folder of the new /usr/local/spark directory to the path of the .bashrc file, then run “source ~/.bashrc” to reload bash.

```
EddieCFox@master:/usr/local$ vim ~/.bashrc
EddieCFox@master:/usr/local$ source ~/.bashrc
EddieCFox@master:/usr/local$
```

```
EddieCFox@master:/usr/local$ spark-shell
Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
22/03/21 09:20:30 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Spark context Web UI available at http://master:4040
Spark context available as 'sc' (master = local[*], app id = local-1647854431647).
Spark session available as 'spark'.
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 3.2.1
      /_/

Using Scala version 2.12.15 (OpenJDK 64-Bit Server VM, Java 1.8.0_312)
Type in expressions to have them evaluated.
Type :help for more information.

scala>
```

I run “spark-shell” and receive this as the output on all 4 nodes.

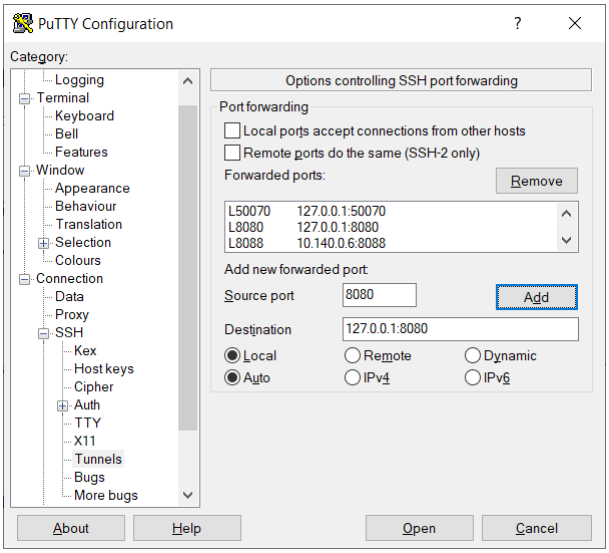
```
scala> :help
All commands can be abbreviated, e.g., :he instead of :help.
:completions <string>    output completions for the given string
:edit <id>|<line>        edit history
:help [command]          print this summary or command-specific help
:history [num]           show the history (optional num is commands to show)
:h? <string>             search the history
:imports [name name ...] show import history, identifying sources of names
:implicits [-v]          show the implicits in scope
:javap <path|class>      disassemble a file or class name
:line <id>|<line>        place line(s) at the end of history
:load <path>             interpret lines in a file
:paste [-raw] [path]     enter paste mode or paste a file
:power                   enable power user mode
:quit                    exit the interpreter
:replay [options]        reset the repl and replay all previous commands
:require <path>          add a jar to the classpath
:reset [options]         reset the repl to its initial state, forgetting all session entries
:save <path>             save replayable session to a file
:sh <command line>       run a shell command (result is implicitly => List[String])
:settings <options>      update compiler options, if possible; see reset
:silent                  disable/enable automatic printing of results
:type [-v] <expr>        display the type of an expression without evaluating it
:kind [-v] <type>        display the kind of a type. see also :help kind
:warnings                show the suppressed warnings from the most recent line which had any

scala> :quit
EddieCFox@master:/usr/local$
```

[4] notes that is output means that Spark is installed successfully on each individual node, however we need to use additional commands to make this a standalone cluster.

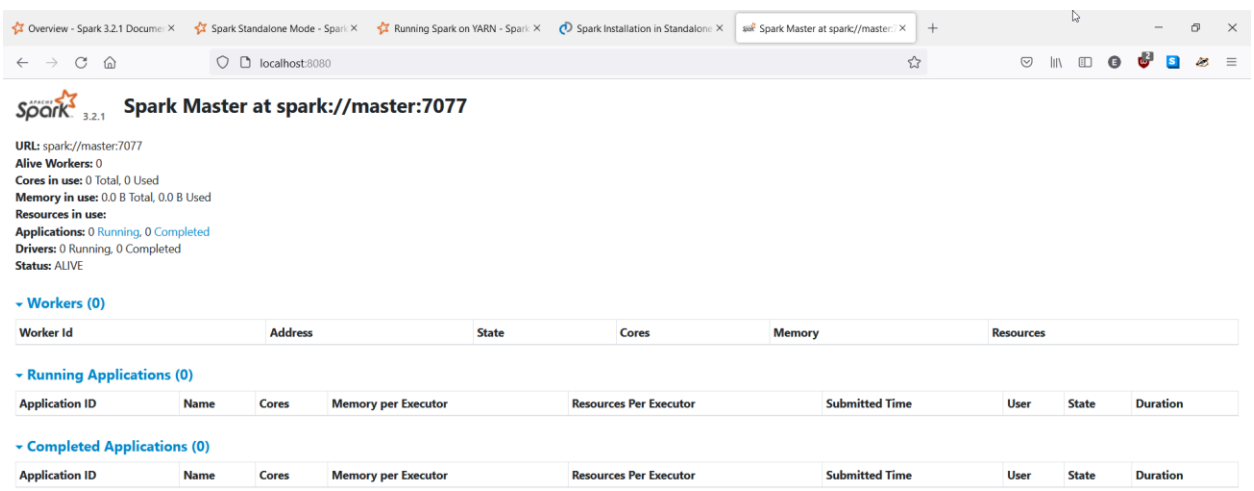
[1] notes that the default port for Spark’s web UI is 8080, so I exit master node and modify Putty’s tunnel settings to allow this port.

Screenshot on next page.




```
EddieCFox@master:/usr/local/spark$ ./sbin/start-master.sh
starting org.apache.spark.deploy.master.Master, logging to /usr/local/spark/logs/spark-EddieCFox-org.apache.spark.deploy.master.Master-1-master.out
EddieCFox@master:/usr/local/spark$ jps
6993 ResourceManager
6452 SecondaryNameNode
8068 Jps
8022 Master
6187 NameNode
EddieCFox@master:/usr/local/spark$
```

I start the Spark standalone master server on the master node and use jps to confirm it is there as "Master".



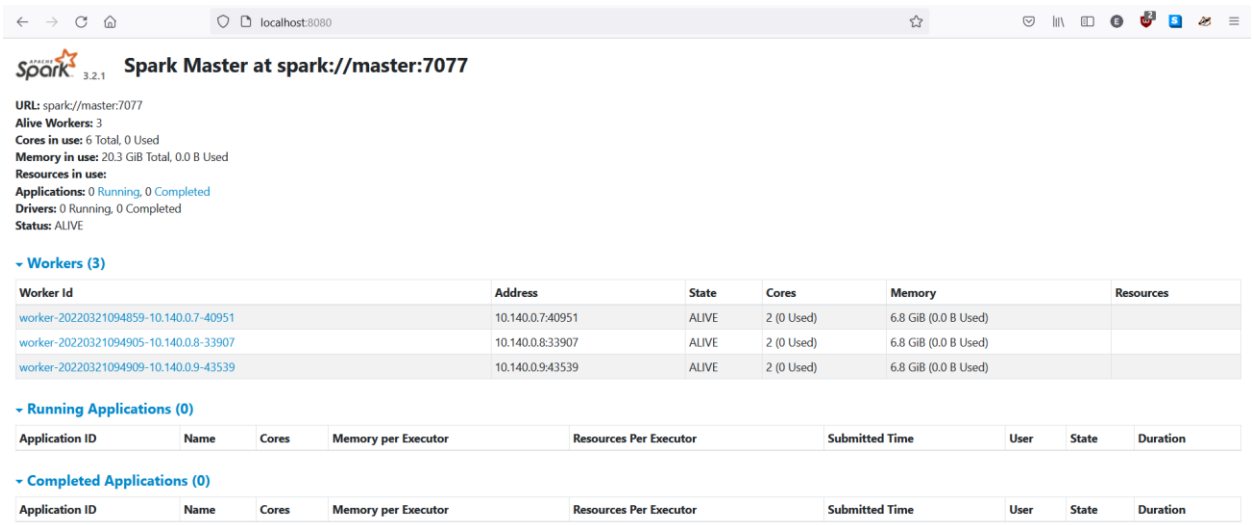
The web interface comes up appropriately.

```
EddieCFox@slave1:/usr/local/spark/sbin$ ls
decommission-slave.sh   spark-daemon.sh          start-master.sh                start-slaves.sh        stop-all.sh               stop-mesos-shuffle-service.sh  stop-worker.sh
decommission-worker.sh  spark-daemons.sh         start-mesos-dispatcher.sh      start-thriftserver.sh  stop-history-server.sh    stop-slave.sh                  stop-workers.sh
slaves.sh               start-all.sh             start-mesos-shuffle-service.sh start-worker.sh        stop-master.sh            stop-slaves.sh                 workers.sh
spark-config.sh         start-history-server.sh  start-slave.sh                 start-workers.sh       stop-mesos-dispatcher.sh  stop-thriftserver.sh
```

When looking in the sbin folder of spark, I noticed there was both start-worker.sh and start-slave.sh executables. Unsure which one to use, I did some research and determined in [5] that slaves was renamed to workers, but both are usable.

As noted in [1], I run "./sbin/start-worker.sh <master-spark-URL>" on each slave node, for a final command of "./sbin/start-worker.sh spark://master:7077"

```
EddieCFox@slave1:/usr/local/spark$ ./sbin/start-worker.sh spark://master:7077
starting org.apache.spark.deploy.worker.Worker, logging to /usr/local/spark/logs/spark-EddieCFox-org.apache.spark.deploy.worker.Worker-1-slave1.out
EddieCFox@slave1:/usr/local/spark$ jps
6490 Worker
6539 Jps
5773 NodeManager
5149 DataNode
EddieCFox@slave1:/usr/local/spark$
```



The worker nodes now show up as alive under the master.

To verify the functionality of the standalone cluster, I run a sample script.

"./bin/spark-submit \

--class org.apache.spark.examples.SparkPi \

--master spark://master:7077 \

--executor-memory 5G \

--total-executor-cores 6 \

/usr/local/spark/examples/jars/spark-examples_2.12-3.2.1.jar \

1000"

Script is based on the sample script for "Spark standalone cluster in client deploy mode" in [6]. I change executor memory to 5g and cores to 6 since I have 2 cores and 20g memory total.

```
22/03/21 10:11:39 INFO TaskSchedulerImpl: Adding task set 0.0 with 1000 tasks resource profile 0
22/03/21 10:11:54 WARN TaskSchedulerImpl: Initial job has not accepted any resources; check your cluster UI to ensure that workers are registered and have sufficient resources
```

I initially encountered this error when –executor-memory was 10G, however my mistake was believing it would execute with 10 gb of memory (RAM) overall instead of per machine, so the master was likely not accepting the job because it was requesting 10gb per machine when only 6.7 was available.

I run Sparkpi successfully on the next run using 5G of executor memory. Output in the screenshot below is a subset of the total because the console generated so many lines it lost the earlier ones, plus I want to keep the length down, so I will skip to near the end, as there is information on 1000 tasks.

```
22/03/21 10:15:49 INFO StandaloneSchedulerBackend: Granted executor ID app-20220321101549-0001/0 on hostPort 10.140.0.7:40951 with 2 core(s), 5.0 GiB RAM
22/03/21 10:15:49 INFO StandaloneAppClient$ClientEndpoint: Executor added: app-20220321101549-0001/1 on worker-20220321094909-10.140.0.9-43539 (10.140.0.9:43539) with 2 core(s)
22/03/21 10:15:49 INFO StandaloneSchedulerBackend: Granted executor ID app-20220321101549-0001/1 on hostPort 10.140.0.9:43539 with 2 core(s), 5.0 GiB RAM
22/03/21 10:15:49 INFO StandaloneAppClient$ClientEndpoint: Executor added: app-20220321101549-0001/2 on worker-20220321094905-10.140.0.8-33907 (10.140.0.8:33907) with 2 core(s)
22/03/21 10:15:49 INFO StandaloneSchedulerBackend: Granted executor ID app-20220321101549-0001/2 on hostPort 10.140.0.8:33907 with 2 core(s), 5.0 GiB RAM
22/03/21 10:15:49 INFO StandaloneAppClient$ClientEndpoint: Executor updated: app-20220321101549-0001/2 is now RUNNING
22/03/21 10:15:49 INFO StandaloneAppClient$ClientEndpoint: Executor updated: app-20220321101549-0001/1 is now RUNNING
22/03/21 10:15:49 INFO StandaloneAppClient$ClientEndpoint: Executor updated: app-20220321101549-0001/0 is now RUNNING
22/03/21 10:15:49 INFO StandaloneSchedulerBackend: SchedulerBackend is ready for scheduling beginning after reached minRegisteredResourcesRatio: 0.0
22/03/21 10:15:50 INFO SparkContext: Starting job: reduce at SparkPi.scala:38
22/03/21 10:15:50 INFO DAGScheduler: Got job 0 (reduce at SparkPi.scala:38) with 1000 output partitions
22/03/21 10:15:50 INFO DAGScheduler: Final stage: ResultStage 0 (reduce at SparkPi.scala:38)
22/03/21 10:15:50 INFO DAGScheduler: Parents of final stage: List()
22/03/21 10:15:50 INFO DAGScheduler: Missing parents: List()
22/03/21 10:15:50 INFO DAGScheduler: Submitting ResultStage 0 (MapPartitionsRDD[1] at map at SparkPi.scala:34), which has no missing parents
22/03/21 10:15:50 INFO MemoryStore: Block broadcast_0 stored as values in memory (estimated size 4.0 KiB, free 366.3 MiB)
22/03/21 10:15:50 INFO MemoryStore: Block broadcast_0_piece0 stored as bytes in memory (estimated size 2.3 KiB, free 366.3 MiB)
22/03/21 10:15:50 INFO BlockManagerInfo: Added broadcast_0_piece0 in memory on master:43249 (size: 2.3 KiB, free: 366.3 MiB)
22/03/21 10:15:50 INFO SparkContext: Created broadcast 0 from broadcast at DAGScheduler.scala:1478
22/03/21 10:15:50 INFO DAGScheduler: Submitting 1000 missing tasks from ResultStage 0 (MapPartitionsRDD[1] at map at SparkPi.scala:34) (first 15 tasks are for partitions Vector(0, 1, 2, 3,
4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14))
22/03/21 10:15:50 INFO TaskSchedulerImpl: Adding task set 0.0 with 1000 tasks resource profile 0
22/03/21 10:15:52 INFO CoarseGrainedSchedulerBackend$DriverEndpoint: Registered executor NettyRpcEndpointRef(spark-client://Executor) (10.140.0.9:36350) with ID 1,  ResourceProfileId 0
22/03/21 10:15:52 INFO CoarseGrainedSchedulerBackend$DriverEndpoint: Registered executor NettyRpcEndpointRef(spark-client://Executor) (10.140.0.8:35376) with ID 2,  ResourceProfileId 0
22/03/21 10:15:52 INFO BlockManagerMasterEndpoint: Registering block manager 10.140.0.8:37751 with 2.5 GiB RAM, BlockManagerId(2, 10.140.0.8, 37751, None)
22/03/21 10:15:52 INFO BlockManagerMasterEndpoint: Registering block manager 10.140.0.9:45443 with 2.5 GiB RAM, BlockManagerId(1, 10.140.0.9, 45443, None)
22/03/21 10:15:52 INFO TaskSetManager: Starting task 0.0 in stage 0.0 (TID 0) (10.140.0.9, executor 1, partition 0, PROCESS_LOCAL, 4582 bytes) taskResourceAssignments Map()
22/03/21 10:15:52 INFO TaskSetManager: Starting task 1.0 in stage 0.0 (TID 1) (10.140.0.9, executor 1, partition 1, PROCESS_LOCAL, 4582 bytes) taskResourceAssignments Map()
22/03/21 10:15:52 INFO CoarseGrainedSchedulerBackend$DriverEndpoint: Registered executor NettyRpcEndpointRef(spark-client://Executor) (10.140.0.7:41618) with ID 0,  ResourceProfileId 0
22/03/21 10:15:52 INFO TaskSetManager: Starting task 2.0 in stage 0.0 (TID 2) (10.140.0.8, executor 2, partition 2, PROCESS_LOCAL, 4582 bytes) taskResourceAssignments Map()
22/03/21 10:15:52 INFO TaskSetManager: Starting task 3.0 in stage 0.0 (TID 3) (10.140.0.8, executor 2, partition 3, PROCESS_LOCAL, 4582 bytes) taskResourceAssignments Map()
22/03/21 10:15:52 INFO BlockManagerMasterEndpoint: Registering block manager 10.140.0.7:46145 with 2.5 GiB RAM, BlockManagerId(0, 10.140.0.7, 46145, None)
22/03/21 10:15:52 INFO BlockManagerInfo: Added broadcast_0_piece0 in memory on 10.140.0.8:37751 (size: 2.3 KiB, free: 2.5 GiB)
22/03/21 10:15:52 INFO BlockManagerInfo: Added broadcast_0_piece0 in memory on 10.140.0.9:45443 (size: 2.3 KiB, free: 2.5 GiB)
22/03/21 10:15:52 INFO TaskSetManager: Starting task 4.0 in stage 0.0 (TID 4) (10.140.0.7, executor 0, partition 4, PROCESS_LOCAL, 4582 bytes) taskResourceAssignments Map()
22/03/21 10:15:52 INFO TaskSetManager: Starting task 5.0 in stage 0.0 (TID 5) (10.140.0.7, executor 0, partition 5, PROCESS_LOCAL, 4582 bytes) taskResourceAssignments Map()
22/03/21 10:15:53 INFO BlockManagerInfo: Added broadcast_0_piece0 in memory on 10.140.0.7:46145 (size: 2.3 KiB, free: 2.5 GiB)
22/03/21 10:15:53 INFO TaskSetManager: Starting task 6.0 in stage 0.0 (TID 6) (10.140.0.9, executor 1, partition 6, PROCESS_LOCAL, 4582 bytes) taskResourceAssignments Map()
22/03/21 10:15:53 INFO TaskSetManager: Starting task 7.0 in stage 0.0 (TID 7) (10.140.0.9, executor 1, partition 7, PROCESS_LOCAL, 4582 bytes) taskResourceAssignments Map()
22/03/21 10:15:53 INFO TaskSetManager: Finished task 1.0 in stage 0.0 (TID 1) in 800 ms on 10.140.0.9 (executor 1) (1/1000)
22/03/21 10:15:53 INFO TaskSetManager: Finished task 0.0 in stage 0.0 (TID 0) in 835 ms on 10.140.0.9 (executor 1) (2/1000)
22/03/21 10:15:53 INFO TaskSetManager: Starting task 8.0 in stage 0.0 (TID 8) (10.140.0.8, executor 2, partition 8, PROCESS_LOCAL, 4582 bytes) taskResourceAssignments Map()
22/03/21 10:15:53 INFO TaskSetManager: Starting task 9.0 in stage 0.0 (TID 9) (10.140.0.8, executor 2, partition 9, PROCESS_LOCAL, 4582 bytes) taskResourceAssignments Map()
22/03/21 10:15:53 INFO TaskSetManager: Finished task 3.0 in stage 0.0 (TID 3) in 810 ms on 10.140.0.8 (executor 2) (3/1000)
22/03/21 10:15:53 INFO TaskSetManager: Finished task 2.0 in stage 0.0 (TID 2) in 815 ms on 10.140.0.8 (executor 2) (4/1000)
22/03/21 10:15:53 INFO TaskSetManager: Starting task 10.0 in stage 0.0 (TID 10) (10.140.0.9, executor 1, partition 10, PROCESS_LOCAL, 4582 bytes) taskResourceAssignments Map()
22/03/21 10:15:53 INFO TaskSetManager: Finished task 6.0 in stage 0.0 (TID 6) in 63 ms on 10.140.0.9 (executor 1) (5/1000)
22/03/21 10:15:53 INFO TaskSetManager: Starting task 11.0 in stage 0.0 (TID 11) (10.140.0.9, executor 1, partition 11, PROCESS_LOCAL, 4582 bytes) taskResourceAssignments Map()
22/03/21 10:15:53 INFO TaskSetManager: Finished task 7.0 in stage 0.0 (TID 7) in 79 ms on 10.140.0.9 (executor 1) (6/1000)
22/03/21 10:15:53 INFO TaskSetManager: Starting task 12.0 in stage 0.0 (TID 12) (10.140.0.8, executor 2, partition 12, PROCESS_LOCAL, 4582 bytes) taskResourceAssignments Map()
```

```
22/03/21 10:15:56 INFO TaskSetManager: Finished task 993.0 in stage 0.0 (TID 993) in 23 ms on 10.140.0.7 (executor 0) (996/1000)
22/03/21 10:15:56 INFO TaskSetManager: Finished task 992.0 in stage 0.0 (TID 992) in 23 ms on 10.140.0.7 (executor 0) (997/1000)
22/03/21 10:15:56 INFO TaskSetManager: Finished task 996.0 in stage 0.0 (TID 996) in 17 ms on 10.140.0.9 (executor 1) (998/1000)
22/03/21 10:15:56 INFO TaskSetManager: Finished task 998.0 in stage 0.0 (TID 998) in 16 ms on 10.140.0.8 (executor 2) (999/1000)
22/03/21 10:15:56 INFO TaskSetManager: Finished task 999.0 in stage 0.0 (TID 999) in 17 ms on 10.140.0.8 (executor 2) (1000/1000)
22/03/21 10:15:56 INFO DAGScheduler: ResultStage 0 (reduce at SparkPi.scala:38) finished in 5.794 s
22/03/21 10:15:56 INFO TaskSchedulerImpl: Removed TaskSet 0.0, whose tasks have all completed, from pool
22/03/21 10:15:56 INFO DAGScheduler: Job 0 is finished. Cancelling potential speculative or zombie tasks for this job
22/03/21 10:15:56 INFO TaskSchedulerImpl: Killing all running tasks in stage 0: Stage finished
22/03/21 10:15:56 INFO DAGScheduler: Job 0 finished: reduce at SparkPi.scala:38, took 5.943657 s
Pi is roughly 3.1416569514165693
22/03/21 10:15:56 INFO SparkUI: Stopped Spark web UI at http://master:4040
22/03/21 10:15:56 INFO StandaloneSchedulerBackend: Shutting down all executors
22/03/21 10:15:56 INFO CoarseGrainedSchedulerBackend$DriverEndpoint: Asking each executor to shut down
22/03/21 10:15:56 INFO MapOutputTrackerMasterEndpoint: MapOutputTrackerMasterEndpoint stopped!
22/03/21 10:15:56 INFO MemoryStore: MemoryStore cleared
22/03/21 10:15:56 INFO BlockManager: BlockManager stopped
22/03/21 10:15:56 INFO BlockManagerMaster: BlockManagerMaster stopped
22/03/21 10:15:56 INFO OutputCommitCoordinator$OutputCommitCoordinatorEndpoint: OutputCommitCoordinator stopped!
22/03/21 10:15:56 INFO SparkContext: Successfully stopped SparkContext
22/03/21 10:15:56 INFO ShutdownHookManager: Shutdown hook called
22/03/21 10:15:56 INFO ShutdownHookManager: Deleting directory /tmp/spark-6bd8c301-224b-4a20-a603-3b082f91ee0a
22/03/21 10:15:56 INFO ShutdownHookManager: Deleting directory /tmp/spark-c7f5eece-93e4-434e-b587-f469825adb1a
EddieCFox@master:/usr/local/spark$
```

The value of Pi is estimated to be 3.1416569 which is pretty close to the actual opening digits: 3.1415926. With this, the validation of the standalone Spark cluster is complete.

Question 1 Section B [Bonus]: Multi-node Spark Cluster over Hadoop YARN

I refer to [7] and [8] when performing the following steps on the master node.

First, the suggestions from [7]:

```
export HADOOP_CONF_DIR=/usr/local/hadoop/etc/hadoop
export SPARK_HOME=/usr/local/spark
export LD_LIBRARY_PATH=/usr/local/hadoop/lib/native:$LD_LIBRARY_PATH
```

I first add these lines to my .bashrc and then use source to reload bash. These notes the directory where hadoop configuration files are located, where I extracted Spark to (/usr/local/spark) and where the libraries of hadoop are located.

I already performed the suggestions in [8] in regards to exporting the bin of Spark to the path when installing Spark as a standalone cluster.

```
EddieCFox@master:/usr/local/spark/conf$ pwd
/usr/local/spark/conf
EddieCFox@master:/usr/local/spark/conf$ ls
fairscheduler.xml.template  log4j.properties.template  metrics.properties.template  spark-defaults.conf.template  spark-env.sh.template  workers.template
EddieCFox@master:/usr/local/spark/conf$ mv spark-defaults.conf.template spark-defaults.conf
EddieCFox@master:/usr/local/spark/conf$ ls
fairscheduler.xml.template  log4j.properties.template  metrics.properties.template  spark-defaults.conf  spark-env.sh.template  workers.template
EddieCFox@master:/usr/local/spark/conf$
```

I rename spark-defaults.conf.template to spark-defaults.conf as suggested in [7].

[9] notes that spark-defaults.conf is one of the paths that spark will read configurations from, and that the format is the key then value, separated by whitespace. I choose to tab separate.

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

# Default system properties included when running spark-submit.
# This is useful for setting default environmental settings.

# Example:
# spark.master                     spark://master:7077
# spark.eventLog.enabled           true
# spark.eventLog.dir               hdfs://namenode:8021/directory
# spark.serializer                 org.apache.spark.serializer.KryoSerializer
# spark.driver.memory              5g
# spark.executor.extraJavaOptions  -XX:+PrintGCDetails -Dkey=value -Dnumbers="one two three"
spark.master    yarn
```

I set the spark.master as yarn and decline to change anything else.

```
EddieCFox@master:/usr/local/spark/conf$ ls
fairscheduler.xml.template  log4j.properties.template  metrics.properties.template  spark-defaults.conf  spark-env.sh.template  workers.template
EddieCFox@master:/usr/local/spark/conf$ vim spark-env.sh.template
EddieCFox@master:/usr/local/spark/conf$ cp spark-env.sh.template spark-env.sh
EddieCFox@master:/usr/local/spark/conf$ ls
fairscheduler.xml.template  log4j.properties.template  metrics.properties.template  spark-defaults.conf  spark-env.sh  spark-env.sh.template  workers.template
EddieCFox@master:/usr/local/spark/conf$
```

[8] leads me to modify spark-env.sh to add some variables. When vim'ing into spark-env.sh.template, it says that it is read only and to copy the contents into spark-env.sh instead of trying to rename the template like I did for spark-defaults.conf.template, so I do so.

```
# Options for native BLAS, like Intel MKL, OpenBLAS, and so on.
# You might get better performance to enable these options if using native BLAS (see SPARK-21305).
# - MKL_NUM_THREADS=1        Disable multi-threading of Intel MKL
# - OPENBLAS_NUM_THREADS=1   Disable multi-threading of OpenBLAS

export HADOOP_CONF_DIR=/usr/local/hadoop/etc/hadoop
export YARN_CONF_DIR=/usr/local/hadoop/etc/hadoop
```

Here I explicitly point to my hadoop / yarn configuration folders.

Final command, running SparkPi again in cluster mode.

"./bin/spark-submit \

  --class org.apache.spark.examples.SparkPi \

  --master yarn \

  --deploy-mode cluster \

  --executor-memory 2G \

  --num-executors 3 \

  /usr/local/spark/examples/jars/spark-examples_2.12-3.2.1.jar \

  1000"

```
EddieCFox@master:/usr/local/spark$ ./bin/spark-submit \
>   --class org.apache.spark.examples.SparkPi \
>   --master yarn \
>   --deploy-mode cluster \
>   --executor-memory 2G \
>   --num-executors 3 \
>   /usr/local/spark/examples/jars/spark-examples_2.12-3.2.1.jar \
>   1000
22/03/21 12:59:11 INFO client.RMProxy: Connecting to ResourceManager at master/10.140.0.6:8040
22/03/21 12:59:12 INFO yarn.Client: Requesting a new application from cluster with 3 NodeManagers
22/03/21 12:59:12 INFO yarn.Client: Verifying our application has not requested more than the maximum memory capability of the cluster (8192 MB per container)
22/03/21 12:59:12 INFO yarn.Client: Will allocate AM container, with 1408 MB memory including 384 MB overhead
22/03/21 12:59:12 INFO yarn.Client: Setting up container launch context for our AM
22/03/21 12:59:12 INFO yarn.Client: Setting up the launch environment for our AM container
22/03/21 12:59:12 INFO yarn.Client: Preparing resources for our AM container
22/03/21 12:59:12 WARN yarn.Client: Neither spark.yarn.jars nor spark.yarn.archive is set, falling back to uploading libraries under SPARK_HOME.
22/03/21 12:59:15 INFO yarn.Client: Uploading resource file:/tmp/spark-22f78257-83a2-495c-93ac-2bd5fba9f0b6/__spark_libs__1135271826327111593.zip -> hdfs://master:9000/user/EddieCFox/.spark
Staging/application_1647867389066_0001/__spark_libs__1135271826327111593.zip
22/03/21 12:59:17 INFO yarn.Client: Uploading resource file:/usr/local/spark/examples/jars/spark-examples_2.12-3.2.1.jar -> hdfs://master:9000/user/EddieCFox/.sparkStaging/application_16478
67389066_0001/spark-examples_2.12-3.2.1.jar
22/03/21 12:59:17 INFO yarn.Client: Uploading resource file:/tmp/spark-22f78257-83a2-495c-93ac-2bd5fba9f0b6/__spark_conf__9108762432570838910.zip -> hdfs://master:9000/user/EddieCFox/.spark
Staging/application_1647867389066_0001/__spark_conf__.zip
22/03/21 12:59:17 INFO spark.SecurityManager: Changing view acls to: EddieCFox
22/03/21 12:59:17 INFO spark.SecurityManager: Changing modify acls to: EddieCFox
22/03/21 12:59:17 INFO spark.SecurityManager: Changing view acls groups to:
22/03/21 12:59:17 INFO spark.SecurityManager: Changing modify acls groups to:
22/03/21 12:59:17 INFO spark.SecurityManager: SecurityManager: authentication disabled; ui acls disabled; users  with view permissions: Set(EddieCFox); groups with view permissions: Set();
users  with modify permissions: Set(EddieCFox); groups with modify permissions: Set()
22/03/21 12:59:17 INFO yarn.Client: Submitting application application_1647867389066_0001 to ResourceManager
22/03/21 12:59:18 INFO impl.YarnClientImpl: Submitted application application_1647867389066_0001
22/03/21 12:59:19 INFO yarn.Client: Application report for application_1647867389066_0001 (state: ACCEPTED)
22/03/21 12:59:19 INFO yarn.Client:
         client token: N/A
         diagnostics: [Mon Mar 21 12:59:18 +0000 2022] Scheduler has assigned a container for AM, waiting for AM container to be launched
         ApplicationMaster host: N/A
         ApplicationMaster RPC port: -1
         queue: default
         start time: 1647867557974
         final status: UNDEFINED
         tracking URL: http://master:8088/proxy/application_1647867389066_0001/
         user: EddieCFox
22/03/21 12:59:20 INFO yarn.Client: Application report for application_1647867389066_0001 (state: ACCEPTED)
22/03/21 12:59:21 INFO yarn.Client: Application report for application_1647867389066_0001 (state: ACCEPTED)
22/03/21 12:59:22 INFO yarn.Client: Application report for application_1647867389066_0001 (state: ACCEPTED)
22/03/21 12:59:23 INFO yarn.Client: Application report for application_1647867389066_0001 (state: ACCEPTED)
22/03/21 12:59:24 INFO yarn.Client: Application report for application_1647867389066_0001 (state: ACCEPTED)
22/03/21 12:59:25 INFO yarn.Client: Application report for application_1647867389066_0001 (state: RUNNING)
22/03/21 12:59:25 INFO yarn.Client:
         client token: N/A
         diagnostics: N/A
```

```
22/03/21 12:59:25 INFO yarn.Client:
         client token: N/A
         diagnostics: N/A
         ApplicationMaster host: slave2
         ApplicationMaster RPC port: 40825
         queue: default
         start time: 1647867557974
         final status: UNDEFINED
         tracking URL: http://master:8088/proxy/application_1647867389066_0001/
         user: EddieCFox
22/03/21 12:59:26 INFO yarn.Client: Application report for application_1647867389066_0001 (state: RUNNING)
22/03/21 12:59:27 INFO yarn.Client: Application report for application_1647867389066_0001 (state: RUNNING)
22/03/21 12:59:28 INFO yarn.Client: Application report for application_1647867389066_0001 (state: RUNNING)
22/03/21 12:59:29 INFO yarn.Client: Application report for application_1647867389066_0001 (state: RUNNING)
22/03/21 12:59:30 INFO yarn.Client: Application report for application_1647867389066_0001 (state: RUNNING)
22/03/21 12:59:31 INFO yarn.Client: Application report for application_1647867389066_0001 (state: RUNNING)
22/03/21 12:59:32 INFO yarn.Client: Application report for application_1647867389066_0001 (state: RUNNING)
22/03/21 12:59:33 INFO yarn.Client: Application report for application_1647867389066_0001 (state: RUNNING)
22/03/21 12:59:34 INFO yarn.Client: Application report for application_1647867389066_0001 (state: RUNNING)
22/03/21 12:59:35 INFO yarn.Client: Application report for application_1647867389066_0001 (state: RUNNING)
22/03/21 12:59:36 INFO yarn.Client: Application report for application_1647867389066_0001 (state: RUNNING)
22/03/21 12:59:37 INFO yarn.Client: Application report for application_1647867389066_0001 (state: RUNNING)
22/03/21 12:59:38 INFO yarn.Client: Application report for application_1647867389066_0001 (state: RUNNING)
22/03/21 12:59:39 INFO yarn.Client: Application report for application_1647867389066_0001 (state: FINISHED)
22/03/21 12:59:39 INFO yarn.Client:
         client token: N/A
         diagnostics: N/A
         ApplicationMaster host: slave2
         ApplicationMaster RPC port: 40825
         queue: default
         start time: 1647867557974
         final status: SUCCEEDED
         tracking URL: http://master:8088/proxy/application_1647867389066_0001/
         user: EddieCFox
22/03/21 12:59:39 INFO util.ShutdownHookManager: Shutdown hook called
22/03/21 12:59:39 INFO util.ShutdownHookManager: Deleting directory /tmp/spark-a8e6ca65-0f8f-4a90-b86b-236893290298
22/03/21 12:59:39 INFO util.ShutdownHookManager: Deleting directory /tmp/spark-22f78257-83a2-495c-93ac-2bd5fba9f0b6
EddieCFox@master:/usr/local/spark$
```

The job is successful, but because it is in cluster mode instead of client, the result is written to the web UI, which I did not set up monitoring for. The log is written to slave2 port 8042 but I do not have a tunnel set up for this. Regardless, we have already confirmed that Spark over YARN as configured by me is able to run a Spark command successfully in cluster mode.

Application Overview

User:	EddieCFox
Name:	org.apache.spark.examples.SparkPi
Application Type:	SPARK
Application Tags:	
Application Priority:	0 (Higher Integer value indicates higher priority)
YarnApplicationState:	FINISHED
Queue:	default
FinalStatus Reported by AM:	SUCCEEDED
Started:	Mon Mar 21 12:59:17 +0000 2022
Elapsed:	20sec
Tracking URL:	History
Log Aggregation Status:	DISABLED
Application Timeout (Remaining Time):	Unlimited
Diagnostics:	
Unmanaged Application:	false
Application Node Label expression:	<Not set>
AM container Node Label expression:	<DEFAULT_PARTITION>

Application Metrics

Total Resource Preempted:	<memory:0, vCores:0>
Total Number of Non-AM Containers Preempted:	0
Total Number of AM Containers Preempted:	0
Resource Preempted from Current Attempt:	<memory:0, vCores:0>
Number of Non-AM Containers Preempted from Current Attempt:	0
Aggregate Resource Allocation:	148714 MB-seconds, 53 vcore-seconds
Aggregate Preempted Resource Allocation:	0 MB-seconds, 0 vcore-seconds

Show 20 entries Search:

Attempt ID	Started	Node	Logs	Nodes blacklisted by the app	Nodes blacklisted by the system
appattempt_1647867389066_0001_000001	Mon Mar 21 20:59:17 +0800 2022	http://slave2:8042	Logs	0	0

Question 2 Section A: Naïve Implementation of Page Rank in Spark

Note: I will do the remaining questions with the school's Spark cluster because I am running low on Google Cloud trial credits due to leaving the nodes online for several weeks.

```
[s1155160788@dicvmd10 Assignment3]$ spark-submit --version
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 2.3.0.2.6.5.0-292
      /_/

Using Scala version 2.11.8, Java HotSpot(TM) 64-Bit Server VM, 1.8.0_112
Branch HEAD
Compiled by user jenkins on 2018-05-11T08:28:14Z
Revision b6b578c8dff89bfc07981ea3bda074262c3800ad
Url git@github.com:hortonworks/spark2.git
Type --help for more information.
[s1155160788@dicvmd10 Assignment3]$
```

On the IE DIC cluster, we can see that Spark is version 2.3, so I will need to keep this in mind when programming.

```
[s1155160788@dicvmd10 Assignment3]$ wget https://snap.stanford.edu/data/web-Google.txt.gz
--2022-03-22 12:01:27--  https://snap.stanford.edu/data/web-Google.txt.gz
Resolving snap.stanford.edu (snap.stanford.edu)... 171.64.75.80
Connecting to snap.stanford.edu (snap.stanford.edu)|171.64.75.80|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 21168784 (20M) [application/x-gzip]
Saving to: 'web-Google.txt.gz'

100%[===================================================================================================================================================>] 21,168,784  5.24MB/s   in 5.7s

2022-03-22 12:01:34 (3.56 MB/s) - 'web-Google.txt.gz' saved [21168784/21168784]

[s1155160788@dicvmd10 Assignment3]$ ls
web-Google.txt.gz
[s1155160788@dicvmd10 Assignment3]$
```

I first wget the dataset we are supposed to apply the Pagerank Spark programs to.

```
[s1155160788@dicvmd10 Assignment3]$ gzip -d web-Google.txt.gz
[s1155160788@dicvmd10 Assignment3]$ ls
web-Google.txt
[s1155160788@dicvmd10 Assignment3]$
```

I extract the gz using gzip, revealing a single text file. I run "cat web-Google.txt | less" to see what we are dealing with.

Screenshot on next page.

```
# Directed graph (each unordered pair of nodes is saved once): web-Google.txt
# Webgraph from the Google programming contest, 2002
# Nodes: 875713 Edges: 5105039
# FromNodeId    ToNodeId
0       11342
0       824020
0       867923
0       891835
11342   0
11342   27469
11342   38716
11342   309564
11342   322178
11342   387543
11342   427436
11342   538214
11342   638706
11342   645018
11342   835220
11342   856657
11342   867923
11342   891835
824020  0
824020  91807
824020  322178
824020  387543
824020  417728
824020  438493
824020  500627
824020  535748
824020  695578
824020  867923
824020  891835
867923  0
867923  11342
867923  136593
867923  414038
867923  500627
867923  523684
867923  760842
867923  815602
867923  835220
867923  846213
867923  857527
867923  891835
891835  0
891835  11342
891835  112028
891835  235849
```

We can see both from the output here as well as a review of the slideshow that this file contains two columns, the first being a node and the second one being an outgoing link that is a

destination endpoint for the node of the first column. This is also detailed as FromNodeId and ToNodeId.

An inspection in Visual Studio Code among uploading the file to my computer reveals that it has 5,105,043 lines, which is too much to test with reliably. It doesn't seem easy to create a small subset of the data that I can easily validate the page ranks of after multiple iterations. I can at least create a small sample around 1000 lines that is computed faster.

```
1022    881590  105322
1023    881590  160306
1024    881590  216203
1025    881590  360442
1026    881590  400675
1027    881590  579432
1028    12  844958
1029    13  104279
```

I can see that one of the nodes ends around line 1027, so I use "cat web-Google.txt | head -n 1027 > test-Google.txt" to create a sample. I then upload both the sample and the full text file to HDFS.

```
[s1155160788@dicvmd10 Assignment3]$ cat web-Google.txt | head -n 1027 > test-Google.txt
[s1155160788@dicvmd10 Assignment3]$ ls
test-Google.txt  web-Google.txt
[s1155160788@dicvmd10 Assignment3]$ hdfs dfs -copyFromLocal test-Google.txt
[s1155160788@dicvmd10 Assignment3]$ hdfs dfs -copyFromLocal web-Google.txt
[s1155160788@dicvmd10 Assignment3]$ hdfs dfs -ls
Found 2 items
-rw-r--r--   3 s1155160788 student      14638 2022-03-22 12:36 test-Google.txt
-rw-r--r--   3 s1155160788 student   75380115 2022-03-22 12:36 web-Google.txt
[s1155160788@dicvmd10 Assignment3]$
```

According to [10], I need to create Scala code (since I want to code this in Scala) and create build / compile / assemble a jar using sbt or maven, then I can call the jar as the target in the spark submit command.

It does not appear that sbt is on the cluster.

I learned about the Visual Studio Code + Metals way of developing Scala code, and was planning to do so on my laptop, however it says that it does not support Scala version 2.11.8 which the cluster has, so instead I downloaded Intellij IDEA, community version.

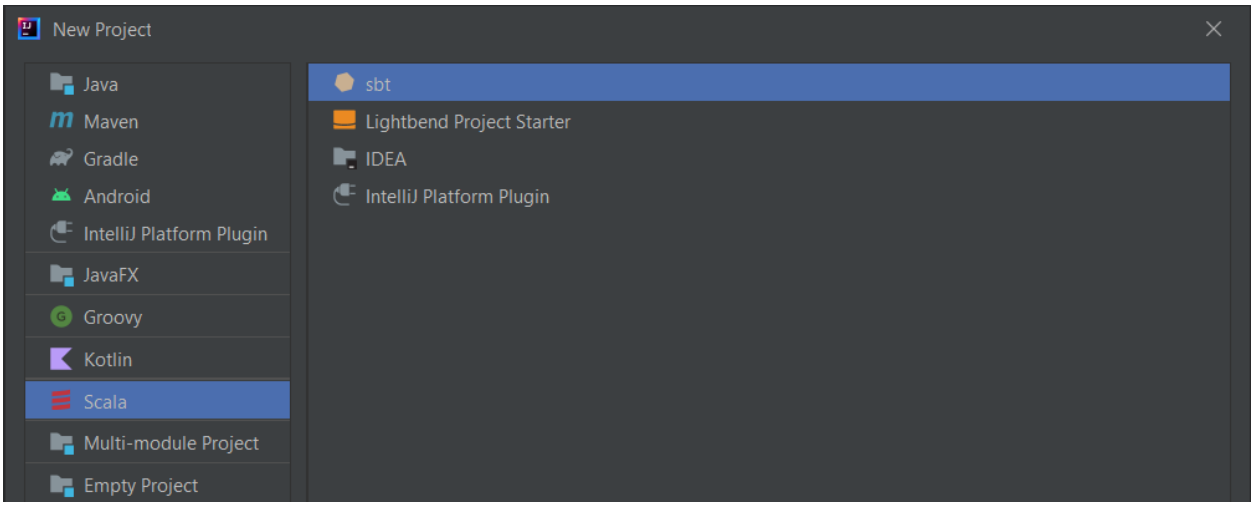
I first download java-1.8.0-openjdk-1.8.0.312-1.b07 from [11] onto my windows computer to match the java version on the IE DIC cluster.

```
C:\Users\Eddie>java -version
openjdk version "1.8.0_312"
OpenJDK Runtime Environment (build 1.8.0_312-b07)
OpenJDK 64-Bit Server VM (build 25.312-b07, mixed mode)
```

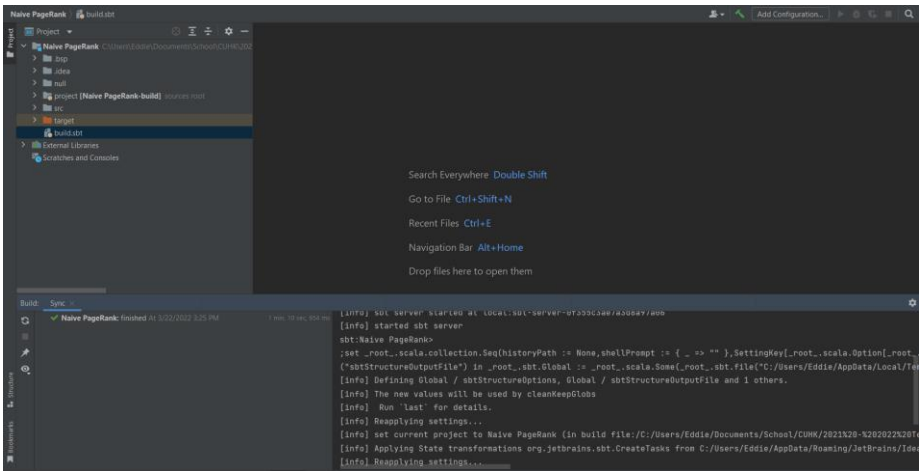
```
[s1155160788@dicvmd10 Assignment3]$ java -version
openjdk version "1.8.0_312"
OpenJDK Runtime Environment (build 1.8.0_312-b07)
OpenJDK 64-Bit Server VM (build 25.312-b07, mixed mode)
```

I can confirm afterwards that the versions are the same.

I download Intellij Idea from [12] and install it, then install the Scala plugin. I refer to [13] and [14] for developing Scala applications for Spark. I download Scala 2.11.8 to match the IE DIC cluster version from [15] and SBT 1.6.2 from [16].

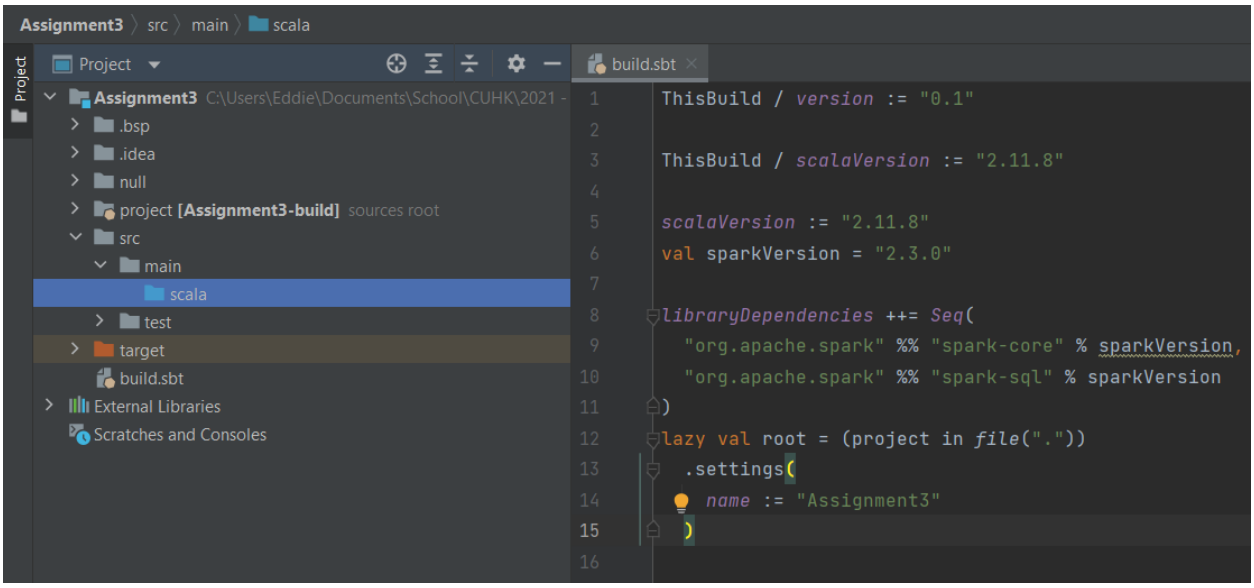


In Intellij, I create a new Scala + sbt project.



After some time, the project is created. I realized that I should probably rename the project to something like Assignment3 and then I can reuse the same project for various scala code under it. I am able to do so by refactoring, editing build.sbt, and then closing and reopening the project.

I add the spark-sql and spark-core dependencies to the build.sbt using [14] as a reference.



Current build.sbt file and project folder structure.

From my understanding in [15], I can now create new packages under /src/main/scala for each question, then within each question, I can create object classes for each subquestion such as Naïve Pagerank and Advanced Pagerank. Ultimately, all these packages under /src/main/scala are part of the "Assignment3" (my project name) module, and I can then compile this into a .jar file that I can upload to the IE DIC cluster for Spark-Submit jobs.

When writing the following code for the Naïve Page Rank, I refer to [17], a naïve pagerank implementation written by Apache on github, as well as [18], our class slideshow on Spark, specifically slides 117-132. I am heavily referencing other slides in [18] for the various transformations and actions that can take place on RDD's.

[18] has most of the needed code for this program, I just need to get up to the point where the file has been read in and assigned to the links .rdd, and all the ranks have been initialized as 1.

I use [17] as an example to create the package, import, object, def main, and SparkSession.builder lines, when combined with my previous knowledge of Spark in Scala.

[17] seems to use Spark Dataframes, but we need to operate at the RDD level, so I use SparkContext and refer to [18] instead.

This program takes one argument, the file path, then converts it to a string, then assigns it to the variable of filePath. Because I am entering the argument, I do not bother with error handling of incorrect number or entry of arguments. I refer to [19] for reading text files in at an RDD level using SparkContext.

I combine Spark context syntax from the slides with the operations performed in [17] to generate links and ranks. I take ranks code directly from [17].

For links, I need to change the split from space separated to tab separated, and I decide to use another .map operation with a trim function to remove leading and trailing whitespace because it is tab separated instead of space separated. distinct, groupByKey and cache are the same as in [17] except I separate everything out into its own line and provide comments for readability and understanding.

Lines 20-27 are taken from Slide 126 of [18].

What we have after this is each node and their final page ranks, all stored in the ranks variable. We need to now sort ranks, then take the top 100 results, and print them out. Because we need to take the top 100, we use take instead of collect. Using [19], I can sort by the 2nd field in reverse order. I mostly copy the final print statement from [17] but also refer to [21] to learn more about string interpolation in the println statements of Scala.

Question 2A Code:

```
build.sbt   NaivePageRank.scala
1   package Question2
2   import org.apache.spark.sql.SparkSession
3
4   object NaivePageRank {
5     def main(args: Array[String]): Unit = {
6       val spark = SparkSession // Create SparkSession
7         .builder
8         .appName( name = "NaivePageRank")
9         .getOrCreate()
10      val sc = spark.sparkContext // Create sparkContext
11      val filePath = args(0).toString // Takes first argument and stores it as file path.
12      val links = sc.textFile(filePath) // Read textfile from file path
13                  .map(line => line.split( regex = "\\t")) // Read each line split with tab separation
14                  .map(parts => (parts(0).trim, parts(1).trim)) // Remove leading whitespace from each part of line
15                  .distinct() // Remove any duplicate neighbors / links
16                  .groupByKey() // Group by each URL / Node / Link.
17                  .cache() // Cache links because we refer to it so much
18      var ranks = links.mapValues(v => 1.0) // Set initial rank for each key in value to 1
19
20      for (i <- 1 ≤ to ≤ 10) {
21        val contribs = links.join(ranks).flatMap{
22          case (url, (links, rank)) =>
23            links.map(dest => (dest, rank/links.size))
24        }
25        ranks = contribs.reduceByKey(_ + _)
26          .mapValues(0.15 + 0.85 * _)
27      }
28      // Sort by pagerank in descending order and take the top 100
29      val top100 = ranks.sortBy(_._2,  ascending = false).take( num = 100)
30      top100.foreach(X=> println(s"${X._1}\t${X._2}"))
```

```
28      // Sort by pagerank in descending order and take the top 100
29      val top100 = ranks.sortBy(_._2, ascending = false).take( num = 100)
30      top100.foreach(X=> println(s"${X._1}\t${X._2}"))
31    }
32  }
```

With the base code completed, I now need to compile it using sbt.

Based on [13], I copy the absolute directory path using Intellij and move it into that directory in cmd run as administrator via cd.

```
C:\Windows\system32>cd C:\Users\Eddie\Documents\School\CUHK\2021 - 2022 Term 2\IEMS 5730 - Big Data Systems and Information Processing\Week 10\Assignment 3\Assignment3

C:\Users\Eddie\Documents\School\CUHK\2021 - 2022 Term 2\IEMS 5730 - Big Data Systems and Information Processing\Week 10\Assignment 3\Assignment3>dir
 Volume in drive C is OS
 Volume Serial Number is 0E25-045C

 Directory of C:\Users\Eddie\Documents\School\CUHK\2021 - 2022 Term 2\IEMS 5730 - Big Data Systems and Information Processing\Week 10\Assignment 3\Assignment3

03/22/2022  04:03 PM    <DIR>          .
03/22/2022  04:03 PM    <DIR>          ..
03/22/2022  03:25 PM    <DIR>          .bsp
03/22/2022  04:03 PM    <DIR>          .idea
03/22/2022  04:03 PM               338 build.sbt
03/22/2022  03:24 PM    <DIR>          null
03/22/2022  03:25 PM    <DIR>          project
03/22/2022  03:24 PM    <DIR>          src
03/22/2022  03:25 PM    <DIR>          target
               1 File(s)            338 bytes
               8 Dir(s)  93,521,420,288 bytes free

C:\Users\Eddie\Documents\School\CUHK\2021 - 2022 Term 2\IEMS 5730 - Big Data Systems and Information Processing\Week 10\Assignment 3\Assignment3>
```

I then run sbt package to compile / package the jar file.

```
C:\Users\Eddie\Documents\School\CUHK\2021 - 2022 Term 2\IEMS 5730 - Big Data Systems and Information Processing\Week 10\Assignment 3\Assignment3>sbt package
[info] welcome to sbt 1.6.2 (ojdkbuild Java 1.8.0_312)
[info] loading project definition from C:\Users\Eddie\Documents\School\CUHK\2021 - 2022 Term 2\IEMS 5730 - Big Data Systems and Information Processing\Week 10\Assignment 3\Assignment3\project
[info] loading settings for project root from build.sbt ...
[info] set current project to Assignment3 (in build file:/C:/Users/Eddie/Documents/School/CUHK/2021%20-%202022%20Term%202/IEMS%205730%20-%20Big%20Data%20Systems%20and%20Information%20Processing/Week%2010/Assignm
ent%203/Assignment3/)
[info] compiling 1 Scala source to C:\Users\Eddie\Documents\School\CUHK\2021 - 2022 Term 2\IEMS 5730 - Big Data Systems and Information Processing\Week 10\Assignment 3\Assignment3\target\scala-2.11\classes ...
[info] Non-compiled module 'compiler-bridge_2.11' for Scala 2.11.8. Compiling...
[info]   Compilation completed in 11.157s.
[success] Total time: 20 s, completed Mar 22, 2022 9:49:49 PM

C:\Users\Eddie\Documents\School\CUHK\2021 - 2022 Term 2\IEMS 5730 - Big Data Systems and Information Processing\Week 10\Assignment 3\Assignment3>
```

« Users › Eddie › Documents › School › CUHK › 2021 - 2022 Term 2 › IEMS 5730 - Big Data Systems and Information Processing › Week 10 › Assignment 3 › Assignment3 › target › scala-2.11

Name	Date modified	Type	Size
classes	3/22/2022 9:49 PM	File folder	
sync	3/22/2022 9:49 PM	File folder	
update	3/22/2022 3:25 PM	File folder	
zinc	3/22/2022 9:49 PM	File folder	
assignment3_2.11-0.1.jar	3/22/2022 9:49 PM	JAR File	13 KB

Afterwards, I can locate the jar file within the /target/scala-2.11 folder within the Assignment3 project folder, and upload it to the cluster using Filezilla.

```
[s1155160788@dicvmd10 Assignment3]$ ls
assignment3_2.11-0.1.jar  test-Google.txt  web-Google.txt
[s1155160788@dicvmd10 Assignment3]$ hdfs dfs -copyFromLocal assignment3_2.11-0.1.jar
[s1155160788@dicvmd10 Assignment3]$ hdfs dfs -ls
Found 3 items
-rw-r--r--   3 s1155160788 student      12815 2022-03-22 21:53 assignment3_2.11-0.1.jar
-rw-r--r--   3 s1155160788 student      14638 2022-03-22 12:36 test-Google.txt
-rw-r--r--   3 s1155160788 student   75380115 2022-03-22 12:36 web-Google.txt
[s1155160788@dicvmd10 Assignment3]$
```

Before I can run the jar file, I will need to find the absolute hdfs path for the text document to pass as an argument to the jar file when invoking the spark cluster. From Assignment 2, I determined that the folder path was /user/s1155160788, but I will also need the port.

```
<property>
  <name>fs.defaultFS</name>
  <value>hdfs://dicvmd2.ie.cuhk.edu.hk:8020</value>
  <final>true</final>
</property>
```

According to [22], I could find the absolute path prefix of the HDFS system by finding the fs.defaultFS property in the core-site.xml folder. Through searching, I determined that the hadoop configuration was stored in /etc/hadoop/2.6.5.0-292/0 and found the full value.

As a string to the final command, I can pass the argument "hdfs://dicvmd2.ie.cuhk.edu.hk:8020/user/s1155160788/test-Google.txt" when testing and "hdfs://dicvmd2.ie.cuhk.edu.hk:8020/user/s1155160788/web-Google.txt" when trying with the full dataset. The jar path is: "hdfs://dicvmd2.ie.cuhk.edu.hk:8020/user/s1155160788/assignment3_2.11-0.1.jar"

Final submission command based on [10]:

"spark-submit \

--class Question2.NaivePageRank \

--master yarn \

--deploy-mode cluster \

hdfs://dicvmd2.ie.cuhk.edu.hk:8020/user/s1155160788/assignment3_2.11-0.1.jar \

hdfs://dicvmd2.ie.cuhk.edu.hk:8020/user/s1155160788/test-Google.txt"

I initially received the following error:

```
Caused by: java.lang.ArrayIndexOutOfBoundsException: 1
        at Question2.NaivePageRank$$anonfun$2.apply(NaivePageRank.scala:14)
        at Question2.NaivePageRank$$anonfun$2.apply(NaivePageRank.scala:14)
```

This was in part because in line 14, I had forgotten to put a space between parts and =>.

Another issue was my assumptions from the text files. If we re-examine the heads of the files, there are four lines that had comments with #. I assumed mistakenly that those lines would be skipped, but it makes sense that these lines would be read as well, the spark application does not parse the text and discriminate between comments or not in a target text file.

Screenshot on next page:

```
# Directed graph (each unordered pair of nodes is saved once): web-Google.txt
# Webgraph from the Google programming contest, 2002
# Nodes: 875713 Edges: 5105039
# FromNodeId    ToNodeId
0       11342
0       824020
0       867923
0       891835
11342   0
11342   27469
11342   38716
11342   309564
```

The issue was that these first four lines had elements without tab separation and I told it to split based on tab separation, so when it tried to access the first part of the split on the line after when trimming, there was probably nothing there, leading to the out of bounds error. Although not directly the same issue, I got inspiration from [23] to resolve the issue.

I modify both the test data and the full dataset to get rid of these first four lines, then recopy them to the HDFS.

```
22/03/22 22:58:48 INFO Client: Application report for application_1644821811052_3210 (state: FINISHED)
22/03/22 22:58:48 INFO Client:
         client token: N/A
         diagnostics: N/A
         ApplicationMaster host: 172.16.5.205
         ApplicationMaster RPC port: 0
         queue: default
         start time: 1647961091832
         final status: SUCCEEDED
         tracking URL: http://dicvmd1.ie.cuhk.edu.hk:8088/proxy/application_1644821811052_3210/
         user: s1155160788
22/03/22 22:58:48 INFO ShutdownHookManager: Shutdown hook called
22/03/22 22:58:48 INFO ShutdownHookManager: Deleting directory /tmp/spark-f33b7503-42e5-43cb-a0e3-a4c0da60a9f4
22/03/22 22:58:48 INFO ShutdownHookManager: Deleting directory /disk1/tmp/spark2/spark-500c402a-0679-4eb3-9549-0b45288f7b90
[s1155160788@dicvmd10 Assignment3]$ ls
assignment3_2.11-0.1.jar  test-Google.txt  web-Google.txt
[s1155160788@dicvmd10 Assignment3]$ hdfs dfs -ls
Found 4 items
drwx------   - s1155160788 student          0 2022-03-22 22:17 .Trash
drwxr-xr-x   - s1155160788 student          0 2022-03-22 22:58 .sparkStaging
-rw-r--r--   3 s1155160788 student      12824 2022-03-22 22:57 assignment3_2.11-0.1.jar
-rw-r--r--   3 s1155160788 student      14449 2022-03-22 22:57 test-Google.txt
```

Running with the test data, it actually succeeds this time. I realize that I didn’t actually tell it to store the data anywhere, just print out.

```
Log Type: stdout
Log Upload Time: Tue Mar 22 22:58:48 +0800 2(
Log Length: 2610
10      1.5756077112123528
181848  1.4864113659539167
393685  1.4502884108276557
310676  1.4502884108276555
556796  1.3992818402405665
226374  0.6391775395788948
514170  0.5925286320238079
877041  0.5775053144738843
134964  0.5585316455227848
19576   0.5574797583393325
43343   0.5432985034351306
358176  0.5394111554092755
487767  0.5394111554092755
400675  0.5206809985874788
332640  0.5198699685119986
579432  0.5157316832728154
360442  0.5157316832728154
216203  0.5126309973530141
160306  0.49233700164640637
42021   0.49233700164640637
597621  0.49106354845502964
85506   0.4837470133857378
194746  0.4837470133857378
644135  0.4686990024616722
430119  0.46744699196268436
30957   0.46744699196268436
565424  0.46063262710564634
462435  0.46063262710564634
581609  0.458266799427946
105322  0.4568225091598429
472889  0.45550208890480337
2       0.45550208890480337
858904  0.45550208890480337
357310  0.45550208890480337
423174  0.45550208890480337
75364   0.44420882373388826
173976  0.4418445831435942
39733   0.4283986915367768
670449  0.42370667199889855
300279  0.4189421649410654
579655  0.41643522324239546
```

It prints successfully to the stdout in the log file of the IE DIC cluster, but I need to route it to a file. I tried various methods of saving to text file within the Scala code, but it doesn't work because we are running a foreach on the top 100 to printout the value in the desired format, but that doesn't actually transform the top100 RDD.

Through [24] and looking at the yarn logs save file, it is possible to save the stdout from the yarn logs to a text file through the following command:

"yarn logs -applicationId (insert application id) -log_files stdout > output.txt"

```
[s1155160788@dicvmd10 Assignment3]$ yarn logs -applicationId application_1644821811052_3211 -log_files stdout > output.txt
22/03/22 23:39:45 INFO client.AHSProxy: Connecting to Application History server at dicvmd2.ie.cuhk.edu.hk/172.16.5.202:10200
22/03/22 23:39:46 INFO client.RequestHedgingRMFailoverProxyProvider: Looking for the active RM in [rm1, rm2]...
22/03/22 23:39:46 INFO client.RequestHedgingRMFailoverProxyProvider: Found active RM [rm2]
22/03/22 23:39:46 INFO zlib.ZlibFactory: Successfully loaded & initialized native-zlib library
22/03/22 23:39:46 INFO compress.CodecPool: Got brand-new decompressor [.deflate]
22/03/22 23:39:46 INFO compress.CodecPool: Got brand-new decompressor [.deflate]
[s1155160788@dicvmd10 Assignment3]$
```

```
[s1155160788@dicvmd10 Assignment3]$ cat output.txt | less
Container: container_e11_1644821811052_3211_01_000001 on dicvmd5.ie.cuhk.edu.hk_45454
LogAggregationType: AGGREGATED
=======================================================================================
LogType:stdout
LogLastModifiedTime:Tue Mar 22 23:08:30 +0800 2022
LogLength:2610
LogContents:
10      1.5756077112123528
181848  1.4864113659539167
393685  1.4502884108276557
310676  1.4502884108276555
556796  1.3992818402405665
226374  0.6391775395788948
514170  0.5925286320238079
877041  0.5775053144738843
134964  0.5585316455227848
19576   0.5574797583393325
43343   0.5432985034351306
358176  0.5394111554092755
```

Output omitted

```
824020  0.19622460449588727
338360  0.19609558682655837
168798  0.19609558682655837
852008  0.19571210744607387
436135  0.19565735705929402
1       0.19565735705929402
599294  0.19324373180703064

End of LogType:stdout
***********************************************************************


End of LogType:stdout
***********************************************************************

(END)
```

The resulting file adds a few lines of information about the log at the beginning and the end, but this can easily be cut out for final output.

My final spark-submit command is the same as the previous one except I use “web-Google.txt” at the end of the argument instead of “test-Google.txt”

Screenshots on next page.

```
[s1155160788@dicvmd10 Assignment3]$ spark-submit \
> --class Question2.NaivePageRank \
> --master yarn \
> --deploy-mode cluster \
> hdfs://dicvmd2.ie.cuhk.edu.hk:8020/user/s1155160788/assignment3_2.11-0.1.jar \
> hdfs://dicvmd2.ie.cuhk.edu.hk:8020/user/s1155160788/web-Google.txt
22/03/22 23:47:26 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Warning: Skip remote jar hdfs://dicvmd2.ie.cuhk.edu.hk:8020/user/s1155160788/assignment3_2.11-0.1.jar.
22/03/22 23:47:26 WARN DomainSocketFactory: The short-circuit local reads feature cannot be used because libhadoop cannot be loaded.
22/03/22 23:47:27 INFO RequestHedgingRMFailoverProxyProvider: Looking for the active RM in [rm1, rm2]...
22/03/22 23:47:27 INFO RequestHedgingRMFailoverProxyProvider: Found active RM [rm2]
22/03/22 23:47:27 INFO Client: Requesting a new application from cluster with 7 NodeManagers
22/03/22 23:47:27 INFO Client: Verifying our application has not requested more than the maximum memory capability of the cluster (188416 MB per container)
22/03/22 23:47:27 INFO Client: Will allocate AM container, with 1408 MB memory including 384 MB overhead
22/03/22 23:47:27 INFO Client: Setting up container launch context for our AM
22/03/22 23:47:27 INFO Client: Setting up the launch environment for our AM container
22/03/22 23:47:27 INFO Client: Preparing resources for our AM container
22/03/22 23:47:28 INFO Client: Use hdfs cache file as spark.yarn.archive for HDP, hdfsCacheFile:hdfs://dicvmd2.ie.cuhk.edu.hk:8020/hdp/apps/2.6.5.0-292/spark2/spark2-hdp-yarn-archive.tar.gz
22/03/22 23:47:28 INFO Client: Source and destination file systems are the same. Not copying hdfs://dicvmd2.ie.cuhk.edu.hk:8020/hdp/apps/2.6.5.0-292/spark2/spark2-hdp-yarn-archive.tar.gz
22/03/22 23:47:28 INFO Client: Source and destination file systems are the same. Not copying hdfs://dicvmd2.ie.cuhk.edu.hk:8020/user/s1155160788/assignment3_2.11-0.1.jar
22/03/22 23:47:28 INFO Client: Uploading resource file:/disk1/tmp/spark2/spark-215e1186-c9f9-4ca9-bac5-843efa460aa5/__spark_conf__1840247375816968094.zip -> hdfs://dicvmd2.ie.cuhk.edu.hk:80
20/user/s1155160788/.sparkStaging/application_1644821811052_3212/__spark_conf__.zip
22/03/22 23:47:28 INFO SecurityManager: Changing view acls to: s1155160788
22/03/22 23:47:28 INFO SecurityManager: Changing modify acls to: s1155160788
22/03/22 23:47:28 INFO SecurityManager: Changing view acls groups to:
22/03/22 23:47:28 INFO SecurityManager: Changing modify acls groups to:
22/03/22 23:47:28 INFO SecurityManager: SecurityManager: authentication disabled; ui acls disabled; users  with view permissions: Set(s1155160788); groups with view permissions: Set(); user
s  with modify permissions: Set(s1155160788); groups with modify permissions: Set()
22/03/22 23:47:28 INFO Client: Submitting application application_1644821811052_3212 to ResourceManager
22/03/22 23:47:28 INFO YarnClientImpl: Submitted application application_1644821811052_3212
22/03/22 23:47:29 INFO Client: Application report for application_1644821811052_3212 (state: ACCEPTED)
22/03/22 23:47:29 INFO Client:
         client token: N/A
         diagnostics: AM container is launched, waiting for AM container to Register with RM
         ApplicationMaster host: N/A
         ApplicationMaster RPC port: -1
         queue: default
         start time: 1647964048380
         final status: UNDEFINED
         tracking URL: http://dicvmd1.ie.cuhk.edu.hk:8088/proxy/application_1644821811052_3212/
         user: s1155160788
22/03/22 23:47:30 INFO Client: Application report for application_1644821811052_3212 (state: ACCEPTED)
22/03/22 23:47:31 INFO Client: Application report for application_1644821811052_3212 (state: ACCEPTED)
22/03/22 23:47:32 INFO Client: Application report for application_1644821811052_3212 (state: RUNNING)
22/03/22 23:47:32 INFO Client:
         client token: N/A
         diagnostics: N/A
         ApplicationMaster host: 172.16.5.205
         ApplicationMaster RPC port: 0
         queue: default
```

```
         ApplicationMaster RPC port: 0
         queue: default
         start time: 1647964048380
         final status: UNDEFINED
         tracking URL: http://dicvmd1.ie.cuhk.edu.hk:8088/proxy/application_1644821811052_3212/
         user: s1155160788
22/03/22 23:47:33 INFO Client: Application report for application_1644821811052_3212 (state: RUNNING)
22/03/22 23:47:34 INFO Client: Application report for application_1644821811052_3212 (state: RUNNING)
22/03/22 23:47:35 INFO Client: Application report for application_1644821811052_3212 (state: RUNNING)
22/03/22 23:47:36 INFO Client: Application report for application_1644821811052_3212 (state: RUNNING)
22/03/22 23:47:37 INFO Client: Application report for application_1644821811052_3212 (state: RUNNING)
22/03/22 23:47:38 INFO Client: Application report for application_1644821811052_3212 (state: RUNNING)
22/03/22 23:47:39 INFO Client: Application report for application_1644821811052_3212 (state: RUNNING)
22/03/22 23:47:40 INFO Client: Application report for application_1644821811052_3212 (state: RUNNING)
22/03/22 23:47:41 INFO Client: Application report for application_1644821811052_3212 (state: RUNNING)
22/03/22 23:47:42 INFO Client: Application report for application_1644821811052_3212 (state: RUNNING)
22/03/22 23:47:43 INFO Client: Application report for application_1644821811052_3212 (state: RUNNING)
22/03/22 23:47:44 INFO Client: Application report for application_1644821811052_3212 (state: RUNNING)
22/03/22 23:47:45 INFO Client: Application report for application_1644821811052_3212 (state: RUNNING)
22/03/22 23:47:46 INFO Client: Application report for application_1644821811052_3212 (state: RUNNING)
22/03/22 23:47:47 INFO Client: Application report for application_1644821811052_3212 (state: RUNNING)
22/03/22 23:47:48 INFO Client: Application report for application_1644821811052_3212 (state: RUNNING)
22/03/22 23:47:49 INFO Client: Application report for application_1644821811052_3212 (state: RUNNING)
22/03/22 23:47:50 INFO Client: Application report for application_1644821811052_3212 (state: RUNNING)
22/03/22 23:47:51 INFO Client: Application report for application_1644821811052_3212 (state: RUNNING)
22/03/22 23:47:52 INFO Client: Application report for application_1644821811052_3212 (state: RUNNING)
22/03/22 23:47:53 INFO Client: Application report for application_1644821811052_3212 (state: RUNNING)
22/03/22 23:47:54 INFO Client: Application report for application_1644821811052_3212 (state: RUNNING)
22/03/22 23:47:55 INFO Client: Application report for application_1644821811052_3212 (state: RUNNING)
22/03/22 23:47:56 INFO Client: Application report for application_1644821811052_3212 (state: RUNNING)
22/03/22 23:47:57 INFO Client: Application report for application_1644821811052_3212 (state: RUNNING)
22/03/22 23:47:58 INFO Client: Application report for application_1644821811052_3212 (state: RUNNING)
22/03/22 23:47:59 INFO Client: Application report for application_1644821811052_3212 (state: RUNNING)
22/03/22 23:48:00 INFO Client: Application report for application_1644821811052_3212 (state: RUNNING)
22/03/22 23:48:01 INFO Client: Application report for application_1644821811052_3212 (state: RUNNING)
22/03/22 23:48:02 INFO Client: Application report for application_1644821811052_3212 (state: RUNNING)
22/03/22 23:48:03 INFO Client: Application report for application_1644821811052_3212 (state: RUNNING)
22/03/22 23:48:04 INFO Client: Application report for application_1644821811052_3212 (state: RUNNING)
22/03/22 23:48:05 INFO Client: Application report for application_1644821811052_3212 (state: RUNNING)
22/03/22 23:48:06 INFO Client: Application report for application_1644821811052_3212 (state: RUNNING)
22/03/22 23:48:07 INFO Client: Application report for application_1644821811052_3212 (state: RUNNING)
22/03/22 23:48:08 INFO Client: Application report for application_1644821811052_3212 (state: RUNNING)
22/03/22 23:48:09 INFO Client: Application report for application_1644821811052_3212 (state: RUNNING)
22/03/22 23:48:10 INFO Client: Application report for application_1644821811052_3212 (state: RUNNING)
22/03/22 23:48:11 INFO Client: Application report for application_1644821811052_3212 (state: RUNNING)
22/03/22 23:48:12 INFO Client: Application report for application_1644821811052_3212 (state: RUNNING)
22/03/22 23:48:13 INFO Client: Application report for application_1644821811052_3212 (state: RUNNING)
22/03/22 23:48:14 INFO Client: Application report for application_1644821811052_3212 (state: RUNNING)
22/03/22 23:48:15 INFO Client: Application report for application_1644821811052_3212 (state: RUNNING)
22/03/22 23:48:16 INFO Client: Application report for application_1644821811052_3212 (state: RUNNING)
```

```
22/03/22 23:49:05 INFO Client: Application report for application_1644821811052_3212 (state: RUNNING)
22/03/22 23:49:06 INFO Client: Application report for application_1644821811052_3212 (state: FINISHED)
22/03/22 23:49:06 INFO Client:
         client token: N/A
         diagnostics: N/A
         ApplicationMaster host: 172.16.5.205
         ApplicationMaster RPC port: 0
         queue: default
         start time: 1647964048380
         final status: SUCCEEDED
         tracking URL: http://dicvmd1.ie.cuhk.edu.hk:8088/proxy/application_1644821811052_3212/
         user: s1155160788
22/03/22 23:49:06 INFO ShutdownHookManager: Shutdown hook called
22/03/22 23:49:06 INFO ShutdownHookManager: Deleting directory /disk1/tmp/spark2/spark-215e1186-c9f9-4ca9-bac5-843efa460aa5
22/03/22 23:49:06 INFO ShutdownHookManager: Deleting directory /tmp/spark-43e32a89-592d-45e8-8c15-2a7b926c6009
[s1155160788@dicvmd10 Assignment3]$
```

Application application_1644821811052_3212

User:	s1155160788
Name:	Question2.NaivePageRank
Application Type:	SPARK
Application Tags:	
Application Priority:	0 (Higher Integer value indicates higher priority)
YarnApplicationState:	FINISHED
Queue:	default
FinalStatus Reported by AM:	SUCCEEDED
Started:	Tue Mar 22 23:47:28 +0800 2022
Elapsed:	1mins, 37sec
Tracking URL:	History
Log Aggregation Status	SUCCEEDED
Diagnostics:	
Unmanaged Application:	false
Application Node Label expression:	<Not set>
AM container Node Label expression:	<DEFAULT_PARTITION>

Question 2 Section A: Final Submission

Screenshots of the final run right before this section. The application succeeds in 1 minute, 37 seconds.

Final build.sbt code on page 15.

Final NaivePageRank.scala code on Page 16 and 17.

I will upload the top 100 nodes as a separate file, and also post it through screenshots here.

Note: The leftmost number is the line number, which is also the ranking. The second number is the node and the third number is the page rank. I did 10 iterations of the Pagerank algorithm, so the top 100 results may be somewhat different from a program that uses more or less iterations.

Question_2A_Top100.txt

C: > Users > Eddie > Documents > School >

```
41909    445.71778596856507
597621   406.6283667503003
504140   399.08930874749063
384666   392.8258437305223
537039   383.90912550319183
486980   382.1656507837419
751384   361.81353653079805
32163    361.42209598491314
163075   357.91973093907933
605856   356.5971554341691
765334   349.9418245162529
558791   343.49196417404994
425770   333.0037797186562
173976   328.69893700049727
908351   314.9342171185024
905628   296.66110982812484
459074   295.295214764298
551829   290.5874145390438
452291   288.1873330782462
687325   285.78728348761473
396321   282.03950023667204
172133   281.7023320213317
407610   280.2651974961968
226374   277.08074096620055
828963   276.7029836052786
7314     266.7792421212717
691633   263.6021134492983
885605   263.29894281517727
213432   256.6545534889912
182121   252.7379501915701
726490   251.10780679100554
804489   249.56347948656162
324502   245.24078532316628
721816   236.99041481067962
277876   233.48250393683696
599130   233.3658457870669
245186   231.75060822375417
```

38	185067	228.87421918562245
39	402132	228.19557839268123
40	54147	227.57684176408924
41	587935	223.4818840993128
42	185821	221.14852171845507
43	448961	219.13169671375866
44	57791	218.60319393508246
45	399699	218.32012210961085
46	621670	217.78711378379302
47	555924	210.7634543262724
48	191990	209.43009916575417
49	434134	207.18961566533076
50	870811	206.64881385804614
51	699754	199.39261525784443
52	323129	197.73674134490048
53	614831	197.29121491144502
54	486434	196.81113759352206
55	468736	195.2614775365472
56	715883	191.6098325833817
57	626893	187.00053551470617
58	818382	186.0798193818107
59	596972	185.11846358075348
60	354640	184.7485757261757
61	183501	182.8509054390332
62	74397	179.4402025053406
63	819223	178.84424802037034
64	846221	177.29519497561154
65	21682	176.79800542211726
66	1536	176.3957091975052
67	637936	171.83906449797615
68	587721	170.46661222862636
69	223236	169.77067366957627
70	666882	168.70393256398188
71	557124	167.850955545494
72	466323	167.35507075759446
73	483532	163.3087056611838
74	138746	162.8488862594346

75	227675	160.30037790267247
76	812640	157.97558366744613
77	187242	157.54813446137192
78	438493	156.0438359269806
79	522190	155.48332945833008
80	369102	155.11774193650606
81	772466	153.92800931898205
82	798584	152.58942793879848
83	427629	146.39551390023817
84	495084	146.2272034799268
85	298511	145.51276521495436
86	133962	142.9368312579908
87	438491	139.8018744350016
88	564521	137.69328651273364
89	36292	136.94078551642664
90	48888	136.83717645193852
91	704396	135.6448926127426
92	887710	133.92928077482165
93	655158	128.52313818101908
94	556129	126.73802544897606
95	456932	126.04758256982129
96	673065	125.5212132499491
97	352975	124.99433737011343
98	539241	124.24430480521207
99	232639	122.10533577340205
100	671193	121.334701716827

Question 2 Section B: Advanced Implementation of Page Rank

As suggested by both the HW3 PDF and [18] slide 128, we need to take advantage of the .partitionBy(new HashPartitioner(number)) line for the links RDD, then Spark will partition the ranks RDD in the same way to optimize the joins and reduce shuffling overhead. It will still have to shuffle during the reduceBy, but not during the join between links and ranks RDD's.

Since we need to try multiple partition numbers, it will be better to create a variable for the number of partitions that pulls another argument so I can just call the same class with different numerical arguments instead of recompiling every time I want to change the number.

I first copy paste the entire NaivePageRank code into a newly created AdvancedPageRank scala object class under the Question 2 package, since I am only adding two lines to it, and the rest is still applicable.

I wanted to use toInt() to convert the second argument to an int as suggested in [25], but that was only introduced in Scala 2.13, and the server uses 2.11.8. I tried to explicitly define the variable as Int as suggested in [26], like “val numPartitions: Int = args(1)” but it knew the argument was a string and that was a mismatch.

When combining both [25] and [26], the compiler seems to be fine with it. “val numPartitions: Int = args(1).toInt”.

[27] confirms that the number given to new Hash partitioner is the number of partitions there will be.

Final code:

build.sbt | AdvancedPageRank.scala

```scala
package Question2
import org.apache.spark.HashPartitioner
import org.apache.spark.sql.SparkSession

object AdvancedPageRank {
  def main(args: Array[String]): Unit = {
    val spark = SparkSession // Create SparkSession
      .builder
      .appName( name = "NaivePageRank")
      .getOrCreate()
    val sc = spark.sparkContext // Create sparkContext
    val filePath = args(0).toString // Takes first argument and stores it as file path.
    val numPartitions: Int = args(1).toInt // Take second argument for number of partitions
    val links = sc.textFile(filePath) // Read textfile from file path
        .map(line => line.split( regex = "\t")) // Read each line split with tab separation
        .map(parts => (parts(0).trim, parts(1).trim)) // Remove leading whitespace from each part of line
        .distinct() // Remove any duplicate neighbors / links
        .groupByKey() // Group by each URL / Node / Link.
        .partitionBy(new HashPartitioner(numPartitions))
        .cache() // Cache links because we refer to it so much
    var ranks = links.mapValues(v => 1.0) // Set initial rank for each key in value to 1

    for (i <- 1 ≤ to ≤ 10) {
      val contribs = links.join(ranks).flatMap{
        case (url, (links, rank)) =>
          links.map(dest => (dest, rank/links.size))
      }
      ranks = contribs.reduceByKey(_ + _)
        .mapValues(0.15 + 0.85 * _)
    }
```

```
30        }
31        // Sort by pagerank in descending order and take the top 100
32        val top100 = ranks.sortBy(_._2, ascending = false).take( num = 100)
33        top100.foreach(X=> println(s"${X._1}\t${X._2}"))
34      }
35    }
```

The new lines are line 13 and line 19. I decide to have cache last so that it is only after all the partitioning.

```
C:\Users\Eddie\Documents\School\CUHK\2021 - 2022 Term 2\IEMS 5730 - Big Data Systems and Information Processing\Week 10\Assignment 3\Assignment3>sbt package
[info] welcome to sbt 1.6.2 (ojdkbuild Java 1.8.0_312)
[info] loading project definition from C:\Users\Eddie\Documents\School\CUHK\2021 - 2022 Term 2\IEMS 5730 - Big Data Systems and Information Processing\Week 10\Assignment 3\Assignment3\project
[info] loading settings for project root from build.sbt ...
[info] set current project to Assignment3 (in build file:/C:/Users/Eddie/Documents/School/CUHK/2021%20-%202022%20Term%202/IEMS%205730%20-%20Big%20Data%20Systems%20and%20Information%20Processing/Week%2010/Assignm
ent%203/Assignment3/)
[info] compiling 1 Scala source to C:\Users\Eddie\Documents\School\CUHK\2021 - 2022 Term 2\IEMS 5730 - Big Data Systems and Information Processing\Week 10\Assignment 3\Assignment3\target\scala-2.11\classes ...
[warn] multiple main classes detected: run 'show discoveredMainClasses' to see the list
[success] Total time: 7 s, completed Mar 25, 2022 12:59:40 PM

C:\Users\Eddie\Documents\School\CUHK\2021 - 2022 Term 2\IEMS 5730 - Big Data Systems and Information Processing\Week 10\Assignment 3\Assignment3>
```

I compile the new jar in SBT again, then transfer it to IE DIC cluster and then HDFS.

Final command:

“spark-submit \

--class Question2.AdvancedPageRank \

--master yarn \

--deploy-mode cluster \

hdfs://dicvmd2.ie.cuhk.edu.hk:8020/user/s1155160788/assignment3_2.11-0.1.jar \

hdfs://dicvmd2.ie.cuhk.edu.hk:8020/user/s1155160788/web-Google.txt \

3”

Instead of 3, I can put any number of partitions I want, 3 is just an example.

The results should be exactly the same as the NaivePageRank, only differing in the time needed to calculate the final result.

Screenshot on next page.

Question 2A Top 100 (10 Iterations).txt -

File Edit Format View Help

```
41909    445.71778596856507
597621   406.6283667503003
504140   399.08930874749063
384666   392.8258437305223
537039   383.90912550319183
486980   382.1656507837419
751384   361.81353653079805
32163    361.42209598491314
163075   357.91973093907933
605856   356.5971554341691
765334   349.9418245162529
558791   343.49196417404994
425770   333.0037797186562
173976   328.69893700049727
908351   314.9342171185024
```

output.txt - Notepad

File Edit Format View Help

```
41909    445.7177859685641
597621   406.6283667503003
504140   399.0893087474894
384666   392.82584373052333
537039   383.9091255031922
486980   382.1656507837405
751384   361.8135365307974
32163    361.42209598491235
163075   357.91973093907893
605856   356.59715543416894
765334   349.94182451625215
558791   343.4919641740506
425770   333.0037797186572
173976   328.69893700049766
908351   314.9342171185032
```

The output is identical so we have confirmed accuracy on the AdvancedPageRank results and can experiment with different runtimes.

After seeing the output of 1 partition being significantly worse than Naïve Page Rank (97 seconds), I reran Naïve Page Rank because I was working on advanced page rank on another day and wanted consistent results. It did take a little longer this time at 109 seconds, probably because other students are now using the IE DIC cluster as well.

Number of Partitions	Runtime
Naïve PageRank	109 Seconds
1	139 seconds
2	110 seconds
3	85 seconds
4	80 seconds
5	90 seconds
6	87 seconds
8	84 seconds
12	83 seconds
20	88 seconds
30	98 seconds
60	92 seconds
120	105 seconds

Explanation of performance:

Most partition arrangements appear to be better than Naïve Page Rank. There is probably less benefit to the advanced implementation than I expected because the performance loss from the shuffling in the naïve implementation may be compensated by the fact I already stored the links RDD to the cache.

1 partition may not be enough to reduce shuffling, because it would be the same as not partitioning either links or ranks, yet the program would still suffer the overhead of running the partition code and creating custom partitions for links and page rdd which have to be matched, which would explain why 1 partition has worse runtime than Naïve Pagerank.

The performance peaks around 4 partitions, where it is about 30% faster than the naïve implementation. At this amount, there is sufficient reduction of shuffling in the joins of the links and ranks rdd's without there being so many partitions that the overhead cuts into the gains.

After 4 partitions, the performance slowly gets worse but there isn't too much variation until after 20 partitions, where a critical point is reached and the overhead from scheduling the partitions greatly eats into the gains from shuffling reduction. By 120 partitions, the performance is nearly as bad as naïve page rank, and if the number were increased enough, the performance would probably become worse than naïve.

Depending on how Spark implements the partition, it is possible that each partition needs to be cached individually after I call the .cache() command after the .partitionBy() command, which would increase the overhead and reduce the performance gains from each partition, as it is probably faster to cache the entire links RDD.

Question 3 Section A: Reading and Truncating Crime Record using Spark SQL

I first start by downloading the data from the specified website. I go to http://opendata.dc.gov/datasets/crime-incidents-in-2013 , then click "View data table", then download, click "Download options" on CSV, then choose "Download file previously generated on Mar 25, 2022, 12:55".

It appears there is 35,927 records with 25 columns, although only 24 of them are filled out.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
1	X	Y	CCN	REPORT_C	SHIFT	METHOD	OFFENSE	BLOCK	XBLOCK	YBLOCK	WARD	ANC	DISTRICT	PSA	NEIGHBOF	BLOCK_GF	CENSUS_T	VOTING_P	LATITUDE	LONGITUD	BID	START_DA	END_DAT
2	-77.0122	38.8714	4104147	2013/04/1	MIDNIGHT	KNIFE	HOMICIDE	1500 - 159	398943.6	133729.3	6	6D	1	105	Cluster 9	006400 2	6400	Precinct 1	38.87139	-77.0122		2004/07/2	2004/07/2
3	-76.9751	38.91729	10028985	2013/02/2	MIDNIGHT	OTHERS	SEX ABUSE	1800 - 229	402158.3	138824.5	5	5C	5	505	Cluster 22	011100 3	11100	Precinct 7	38.91729	-76.9751		2010/03/0	2010/03/0
4	-76.9672	38.87208	10033521	2013/10/1	MIDNIGHT	OTHERS	SEX ABUSE	2800 - 282	402846.6	133805.6	7	7B	6	605	Cluster 34	007709 2	7709	Precinct 1	38.87207	-76.9672		2008/12/3	2008/12/3
5	-77.0139	38.81348	10124918	2013/04/0	MIDNIGHT	OTHERS	SEX ABUSE	1 - 99 BLO	398794	127300.2	8	8D	7	708	Cluster 44	010900 2	10900	Precinct 1	38.81347	-77.0139		2010/08/27	21:00:00
6	-77.0139	38.81348	10124918	2013/04/0	MIDNIGHT	OTHERS	SEX ABUSE	1 - 99 BLO	398794	127300.2	8	8D	7	708	Cluster 44	010900 2	10900	Precinct 1	38.81347	-77.0139		2010/08/27	21:00:00
7	-76.9962	38.92792	11010107	2013/07/3	MIDNIGHT	OTHERS	HOMICIDE	3000 - 309	400331.4	140004.2	5	5E	5	502	Cluster 21	009201 2	9201	Precinct 7	38.92792	-76.9962		2011/01/23	12:50:00
8	-76.9431	38.88369	11045512	2013/01/3	MIDNIGHT	GUN	HOMICIDE	400 - 499	404939.9	135095.9	7	7F	6	603	Cluster 33	007703 1	7703	Precinct 1	38.88369	-76.9431		2011/04/0	2011/04/0
9	-76.9392	38.89228	11250281	2013/07/0	MIDNIGHT	OTHERS	SEX ABUSE	116 - 221	405274.6	136049	7	7D	6	602	Cluster 33	007803 4	7803	Precinct 9	38.89227	-76.9392		2011/05/1	2011/05/1
10	-76.9931	38.92222	12003591	2013/01/0	MIDNIGHT	OTHERS	THEFT/OT	900 - 999	400594.9	139371.6	5	5C	5	505	Cluster 22	009102 4	9102	Precinct 7	38.92222	-76.9931		2013/01/0	2013/01/0
11	-77.0199	38.87744	12139462	2013/11/1	MIDNIGHT	OTHERS	SEX ABUSE	1000 - 119	398274	134400.7	6	6D	1	103	Cluster 9	010201 3	10201	Precinct 1	38.87744	-77.0199	SOUTHWE	2012/09/3	2012/10/0
12	-76.9344	38.88581	12182426	2013/01/0	MIDNIGHT	GUN	ROBBERY	4700 - 489	405689.1	135331.4	7	7E	6	604	Cluster 33	009906 1	9906	Precinct 1	38.8858	-76.9344		2013/01/0	2013/01/0

Since there is no clickable URL when downloading the file, I can't use wget to get it into the IE DIC cluster, so I instead upload it using Filezilla.

```
<U+FEFF>X,Y,CCN,REPORT_DAT,SHIFT,METHOD,OFFENSE,BLOCK,XBLOCK,YBLOCK,WARD,ANC,DISTRICT,PSA,NEIGHBORHOOD_CLUSTER,BLOCK_GROUP,CENSUS_TRACT,VOTING_PRECINCT,LATITUDE,LONGITUDE,BID,START_DATE,END
_DATE,OBJECTID,OCTO_RECORD_ID
-77.0121756892146,38.8713969658963,04104147,2013/04/16 04:00:00+00,MIDNIGHT,KNIFE,HOMICIDE,1500 - 1599 BLOCK OF 1ST STREET SW,398943.608380973,133729.349977531,6,6D,1,105,Cluster 9,006400 2
,006400,Precinct 127,38.8713891837,-77.0121734039,,2004/07/28 00:30:00+00,2004/07/28 00:30:00+00,131970741,
-76.9751148317011,38.9172942137185,10028985,2013/02/27 05:00:00+00,MIDNIGHT,OTHERS,SEX ABUSE,1800 - 2299 BLOCK OF NEW YORK AVENUE NE,402158.31,138824.53,5,5C,5,505,Cluster 22,011100 3,01110
0,Precinct 72,38.9172864211,-76.9751125549,,2010/03/07 07:30:00+00,2010/03/07 07:45:00+00,132008691,
-76.9671993057411,38.8720794353287,10033521,2013/10/10 04:00:00+00,MIDNIGHT,OTHERS,SEX ABUSE,2800 - 2821 BLOCK OF PENNSYLVANIA AVENUE SE,402846.57,133805.55,7,7B,6,605,Cluster 34,007709 2,0
07709,Precinct 111,38.8720716515,-76.9671970341,,2008/12/30 01:00:00+00,2008/12/30 01:00:00+00,132011163,
-77.0138879791717,38.013480469491,10124918,2013/04/09 04:00:00+00,MIDNIGHT,OTHERS,SEX ABUSE,1 - 99 BLOCK OF DC VILLAGE LANE SW,398794.04,127300.24,8,8D,7,708,Cluster 44,010900 2,010900,Prec
inct 126,38.8134726993,-77.0138856966,,2010/08/27 21:00:00+00,,132018556,
```

Looking at a sampling of the data on the cluster, it doesn't appear as neat, but the information is still intact, so I upload it to HDFS.

```
[s1155160788@dicvmd10 Assignment3]$ hdfs dfs -copyFromLocal Crime_Incidents_in_2013.csv
[s1155160788@dicvmd10 Assignment3]$ hdfs dfs -ls
Found 5 items
drwx------   - s1155160788 student          0 2022-03-25 14:00 .Trash
drwxr-xr-x   - s1155160788 student          0 2022-03-25 13:46 .sparkStaging
-rw-r--r--   3 s1155160788 student   10481990 2022-03-25 15:43 Crime_Incidents_in_2013.csv
-rw-r--r--   3 s1155160788 student      14449 2022-03-22 22:57 test-Google.txt
-rw-r--r--   3 s1155160788 student   75379926 2022-03-22 23:04 web-Google.txt
[s1155160788@dicvmd10 Assignment3]$
```

Based on what we learned in Question 2, the file path in HDFS is: "hdfs://dicvmd2.ie.cuhk.edu.hk:8020/user/s1155160788/ Crime_Incidents_in_2013.csv"

Since we are now allowed to use dataframes, I will do so since it is higher level and easier to program in. I refer to our Spark SQL slides [28] when programming the code for this question. The dataframe programming here seems to be done with Spark Context, so I needed to figure out what functions should be imported in Spark 2.0 or higher when sessions is introduced. [29] is a good guide from Apache on Spark SQL and Dataframes in Spark 2.3.0, which the cluster runs on.

I created a new package called Question3, and have decided to have 2 classes overall, since Part B of this question seems to extend the code from Part A. The first class is called Crime2013 and used for Part A and B, while the second class is called Gun2018 and is on Part C. I will compile the jar after finishing Part A to confirm functionality, then recompile after Part B.

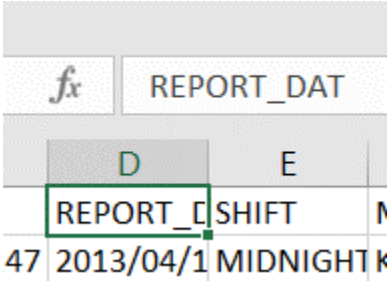
I copy the initial code from Question 2 to create the spark session, and define the filepath, but since we are operating on the dataframe level, we do not need to create a spark context. According to [29] and [30], it may be helpful to add "import org.apache.spark.sql.functions._" for access to built in DataFrame API functions, so I do so. [29] provides an example of how to load a CSV file, so I copy that code, changing the separator from ";" to ",", verified by the previous output of the actual CSV file into the cluster with cat instead of the neat output from Excel on my local machine.

inferSchema, true means that Spark will go through the csv file and infer the schema of each column [31].

Header, true is to be used when the csv file has the column names in the first row, which this file does [31].

I select the 6 columns by using .select method as detailed in [29] under "Untyped Dataset Operations (aka DataFrame Operations)".

Note: Even though the Assignment 3 PDF says to select REPORT_DATE, the column is named REPORT_DAT so I need to use that.



I didn't see any rows with null values, but because I cannot manually check the entire dataset, I comply with the requirement in Assignment 3 PDF to filter out rows with NULL values. Because we now have a subset of the total columns, the six we want, we can just drop any record that has NULL values in ANY column. [32] teaches me how to do this through using .na.drop()

At this point, I am done with the question, but I add in lines to print the schema [29] and show the first 10 rows [33] for debugging purposes to make sure things are formatted correctly. These lines will be removed after I extend the code in Section B.

Initial Code for Section A:

```
package Question3
import org.apache.spark.sql.SparkSession
import org.apache.spark.sql.functions._ // import built in dataframe API functions

object Crime2013 {
  def main (args: Array[String]): Unit = {
    val spark = SparkSession // Create SparkSession
      .builder
      .appName( name = "Crime2013")
      .getOrCreate()
    val filePath = args(0).toString // Takes first argument and stores it as file path.

    val crimesTable = spark.read.format( source = "csv") // read CSV formatted file
      .option("sep", ",") // comma separated
      .option("inferSchema", "true") // infer schema
      .option("header", "true") // csv file has header
      .load(filePath) // load file corresponding to the first argument

    // Select only these 6 columns
    val focusedCrimesTable = crimesTable.select( col = "CCN", cols = "REPORT_DAT", "OFFENSE", "METHOD", "END_DATE", "DISTRICT")
    val filteredCrimesTable = focusedCrimesTable.na.drop() // Drop rows with NULL values in ANY of these columns
    filteredCrimesTable.printSchema() // Print the inferred schema for debugging purposes
    filteredCrimesTable.show( numRows = 10) // Show the first 10 rows for debugging purposes
  }
}
```

```
22/03/27 09:58:58 INFO Client: Application report for application_1644821811052_3828 (state: RUNNING)
22/03/27 09:58:59 INFO Client: Application report for application_1644821811052_3828 (state: FINISHED)
22/03/27 09:58:59 INFO Client:
         client token: N/A
         diagnostics: N/A
         ApplicationMaster host: 172.16.5.204
         ApplicationMaster RPC port: 0
         queue: default
         start time: 1648346300740
         final status: SUCCEEDED
         tracking URL: http://dicvmd1.ie.cuhk.edu.hk:8088/proxy/application_1644821811052_3828/
         user: s1155160788
22/03/27 09:58:59 INFO ShutdownHookManager: Shutdown hook called
22/03/27 09:58:59 INFO ShutdownHookManager: Deleting directory /tmp/spark-5d64cd2f-8bce-4a57-90a3-a2d5ab0830b4
22/03/27 09:58:59 INFO ShutdownHookManager: Deleting directory /disk1/tmp/spark2/spark-e3c789a8-84ea-4bd3-9ece-d8c1831e585a
[s1155160788@dicvmd10 Assignment3]$
```

```
root
 |-- CCN: integer (nullable = true)
 |-- REPORT_DAT: string (nullable = true)
 |-- OFFENSE: string (nullable = true)
 |-- METHOD: string (nullable = true)
 |-- END_DATE: string (nullable = true)
 |-- DISTRICT: integer (nullable = true)

+--------+--------------------+------------+------+--------------------+--------+
|     CCN|          REPORT_DAT|     OFFENSE|METHOD|            END_DATE|DISTRICT|
+--------+--------------------+------------+------+--------------------+--------+
| 4104147|2013/04/16 04:00:...|    HOMICIDE| KNIFE|2004/07/28 00:30:...|       1|
|10028985|2013/02/27 05:00:...|   SEX ABUSE|OTHERS|2010/03/07 07:45:...|       5|
|10033521|2013/10/10 04:00:...|   SEX ABUSE|OTHERS|2008/12/30 01:00:...|       6|
|11045512|2013/01/31 05:00:...|    HOMICIDE|   GUN|2011/04/06 02:32:...|       6|
|11250281|2013/07/08 04:00:...|   SEX ABUSE|OTHERS|2011/05/13 03:15:...|       6|
|12003591|2013/01/09 05:59:...| THEFT/OTHER|OTHERS|2013/01/09 05:59:...|       5|
|12139462|2013/11/13 05:00:...|   SEX ABUSE|OTHERS|2012/10/01 08:00:...|       1|
|12182426|2013/01/01 05:15:...|     ROBBERY|   GUN|2013/01/01 05:15:...|       6|
|12182466|2013/01/01 07:10:...|THEFT F/AUTO|OTHERS|2013/01/01 06:15:...|       7|
|12182502|2013/01/01 08:00:...| THEFT/OTHER|OTHERS|2013/01/01 06:30:...|       1|
+--------+--------------------+------------+------+--------------------+--------+
only showing top 10 rows
```

The above is the stdout log output from the application. I looked at the first X entries of the actual CSV file on Excel on my local machine to verify that the NULL dropping functionality was working properly.

CCN 4104147, 10028985, and 10033521 are properly the first 3 entries of the CSV file, but the output here skips to 11045512 which is 4 entries later. I checked the columns and confirmed that CCN 10124918, 10124918 and 11010107 all have NULL values in the END_DATE field. We can therefore confirm that the NULL dropping functionality is correct.

The inferred schema looks mostly correct, however the REPORT_DAT and END_DATE are noted to be strings, which could cause issues if we were to query the report or end date, however this assignment is focused on queries involving the type of offenses and the time shift they occurred, so it should be fine.

For my final submit command, I changed the class name from Crimes2013 to Crimes2013A because Part B will involve working with a different column subset of the data.

I tried to change .show(10) to .show() but when looking at the output, it only showed the first 20 lines. I refer to [34] for the syntax of outputting the filtered table dataframe to CSV with a header.

Final submission command:

“spark-submit \

--class Question3.Crime2013A \

--master yarn \

--deploy-mode cluster \

hdfs://dicvmd2.ie.cuhk.edu.hk:8020/user/s1155160788/assignment3_2.11-0.1.jar \

hdfs://dicvmd2.ie.cuhk.edu.hk:8020/usfoer/s1155160788/Crime_Incidents_in_2013.csv”

Final Section A Code:

build.sbt | Crime2013A.scala

```scala
package Question3
import org.apache.spark.sql.SparkSession
import org.apache.spark.sql.functions._ // import built in dataframe API functions

object Crime2013A {
  def main (args: Array[String]): Unit = {
    val spark = SparkSession // Create SparkSession
      .builder
      .appName( name = "Crime2013")
      .getOrCreate()
    val filePath = args(0).toString // Takes first argument and stores it as file path.

    val crimesTable = spark.read.format( source = "csv") // read CSV formatted file
      .option("sep", ",") // comma separated
      .option("inferSchema", "true") // infer schema
      .option("header", "true") // csv file has header
      .load(filePath) // load file corresponding to the first argument

    // Select only these 6 columns
    val focusedCrimesTable = crimesTable.select( col = "CCN", cols = "REPORT_DAT", "OFFENSE", "METHOD", "END_DATE", "DISTRICT")
    val filteredCrimesTable = focusedCrimesTable.na.drop() // Drop rows with NULL values in ANY of these columns
    filteredCrimesTable.write.option("header",true).csv( path = "hdfs://dicvmd2.ie.cuhk.edu.hk:8020/user/s1155160788/output.csv")
  }
}
```

```
22/03/27 10:40:15 INFO Client: Application report for application_1644821811052_3836 (state: RUNNING)
22/03/27 10:40:16 INFO Client: Application report for application_1644821811052_3836 (state: RUNNING)
22/03/27 10:40:17 INFO Client: Application report for application_1644821811052_3836 (state: FINISHED)
22/03/27 10:40:17 INFO Client:
         client token: N/A
         diagnostics: N/A
         ApplicationMaster host: 172.16.5.210
         ApplicationMaster RPC port: 0
         queue: default
         start time: 1648348776075
         final status: SUCCEEDED
         tracking URL: http://dicvmd1.ie.cuhk.edu.hk:8088/proxy/application_1644821811052_3836/
         user: s1155160788
22/03/27 10:40:17 INFO ShutdownHookManager: Shutdown hook called
22/03/27 10:40:17 INFO ShutdownHookManager: Deleting directory /disk1/tmp/spark2/spark-d86582f5-221f-401e-b7a4-1d66c7f53d1f
22/03/27 10:40:17 INFO ShutdownHookManager: Deleting directory /tmp/spark-56b3893a-df4d-4411-8772-159747fe121d
[s1155160788@dicvmd10 Assignment3]$
```

```
[s1155160788@dicvmd10 Assignment3]$ hdfs dfs -ls
Found 7 items
drwx------   - s1155160788 student          0 2022-03-27 10:21 .Trash
drwxr-xr-x   - s1155160788 student          0 2022-03-27 10:40 .sparkStaging
-rw-r--r--   3 s1155160788 student   10481990 2022-03-25 15:43 Crime_Incidents_in_2013.csv
-rw-r--r--   3 s1155160788 student      27629 2022-03-27 10:39 assignment3_2.11-0.1.jar
drwxr-xr-x   - s1155160788 student          0 2022-03-27 10:40 output.csv
-rw-r--r--   3 s1155160788 student      14449 2022-03-22 22:57 test-Google.txt
-rw-r--r--   3 s1155160788 student   75379926 2022-03-22 23:04 web-Google.txt
[s1155160788@dicvmd10 Assignment3]$ hdfs dfs -copyToLocal output.csv
[s1155160788@dicvmd10 Assignment3]$
```

output.csv

Filename	Filesize	Filetype	Last modifi...	Permissi...	Owner/Gr...
..					
part-00000-3727955a-...	1,104,628	Microsof...	3/27/2022 ...	-rw-r--r--	s1155160...
part-00001-3727955a-...	1,104,383	Microsof...	3/27/2022 ...	-rw-r--r--	s1155160...
part-00002-3727955a-...	550,214	Microsof...	3/27/2022 ...	-rw-r--r--	s1155160...
_SUCCESS	0	File	3/27/2022 ...	-rw-r--r--	s1155160...

I take the several parts outputted by Spark and concatenate them into one file. I will also be uploading this file with my Blackboard submission.

Screenshot of output:

1	CCN	REPORT_DAT	OFFENSE	METHOD	END_DATE	DISTRICT
2	4104147	2013/04/16 04:00:00+00	HOMICIDE	KNIFE	2004/07/28 00:30:00+00	1
3	10028985	2013/02/27 05:00:00+00	SEX ABUSE	OTHERS	2010/03/07 07:45:00+00	5
4	10033521	2013/10/10 04:00:00+00	SEX ABUSE	OTHERS	2008/12/30 01:00:00+00	6
5	11045512	2013/01/31 05:00:00+00	HOMICIDE	GUN	2011/04/06 02:32:00+00	6
6	11250281	2013/07/08 04:00:00+00	SEX ABUSE	OTHERS	2011/05/13 03:15:00+00	6
7	12003591	2013/01/09 05:59:00+00	THEFT/OTHER	OTHERS	2013/01/09 05:59:00+00	5
8	12139462	2013/11/13 05:00:00+00	SEX ABUSE	OTHERS	2012/10/01 08:00:00+00	1
9	12182426	2013/01/01 05:15:00+00	ROBBERY	GUN	2013/01/01 05:15:00+00	6
10	12182466	2013/01/01 07:10:00+00	THEFT F/AUTO	OTHERS	2013/01/01 06:15:00+00	7
11	12182502	2013/01/01 08:00:00+00	THEFT/OTHER	OTHERS	2013/01/01 06:30:00+00	1
12	12182505	2013/01/01 07:29:00+00	ROBBERY	GUN	2013/01/01 07:26:00+00	4
13	12182530	2013/01/01 08:10:00+00	ASSAULT W/DANGEROUS WEAPON	KNIFE	2013/01/01 08:10:00+00	7
14	12182534	2013/01/01 08:12:00+00	ASSAULT W/DANGEROUS WEAPON	GUN	2013/01/01 08:12:00+00	7
15	12182544	2013/01/01 08:31:00+00	THEFT/OTHER	OTHERS	2013/01/01 08:14:00+00	4
16	12182550	2013/01/01 08:45:00+00	THEFT F/AUTO	OTHERS	2013/01/01 08:30:00+00	5
17	12182554	2013/01/01 08:38:00+00	ROBBERY	OTHERS	2013/01/01 08:38:00+00	3
18	12182577	2013/01/01 09:23:00+00	ASSAULT W/DANGEROUS WEAPON	GUN	2013/01/01 09:23:00+00	7
19	12182587	2013/01/01 08:31:00+00	ASSAULT W/DANGEROUS WEAPON	OTHERS	2013/01/01 07:35:00+00	4
20	12182608	2013/01/01 08:56:00+00	SEX ABUSE	OTHERS	2013/01/01 08:56:00+00	2

Question 3 Section B: Queries on Offenses and Time Slots

I first copy the code from Section A up to and including the point the file is read. I will need to modify the dataframes afterwards in different ways to answer this question.

I add SHIFT to the list of columns that are selected. I refer to [29] when making the following queries, and realize that [29] is just the Spark 2.3 update of the Spark SQL programming guide referenced in the assignment 3 PDF (which was for Spark 1.6).

It seems that using .count() creates a count column, so I will order by that instead of offenses in descending order, referring to [35] to do so.

Final Section B Offense Count Code:

```
build.sbt   Crime2013B.scala   Crime2013A.scala
package Question3
import org.apache.spark.sql.SparkSession
import org.apache.spark.sql.functions._ // import built in dataframe API functions

object Crime2013B {
  def main (args: Array[String]): Unit = {
    val spark = SparkSession // Create SparkSession
      .builder
      .appName( name = "Crime2013")
      .getOrCreate()
    val filePath = args(0).toString // Takes first argument and stores it as file path.

    val crimesTable = spark.read.format( source = "csv") // read CSV formatted file
      .option("sep", ",") // comma separated
      .option("inferSchema", "true") // infer schema
      .option("header", "true") // csv file has header
      .load(filePath) // load file corresponding to the first argument

    // Select only these 7 columns
    val focusedCrimesTable = crimesTable.select( col = "CCN", cols = "REPORT_DAT", "OFFENSE", "METHOD", "END_DATE", "DISTRICT", "SHIFT")
    val filteredCrimesTable = focusedCrimesTable.na.drop() // Drop rows with NULL values in ANY of these columns

    val offenses = filteredCrimesTable.groupBy( col1 = "OFFENSE") // group by offenses
                   .count() // count each type of offenses
                   .orderBy(desc( columnName = "count")) // order by count in descending order
    offenses.show() // display the offenses
  }
}
```

Final submission command:

“spark-submit \

--class Question3.Crime2013B \

--master yarn \

--deploy-mode cluster \

hdfs://dicvmd2.ie.cuhk.edu.hk:8020/user/s1155160788/assignment3_2.11-0.1.jar \

hdfs://dicvmd2.ie.cuhk.edu.hk:8020/user/s1155160788/Crime_Incidents_in_2013.csv”

```
22/03/27 15:45:39 INFO Client: Application report for application_1644821811052_3861 (state: RUNNING)
22/03/27 15:45:40 INFO Client: Application report for application_1644821811052_3861 (state: FINISHED)
22/03/27 15:45:40 INFO Client:
         client token: N/A
         diagnostics: N/A
         ApplicationMaster host: 172.16.5.209
         ApplicationMaster RPC port: 0
         queue: default
         start time: 1648367100251
         final status: SUCCEEDED
         tracking URL: http://dicvmd1.ie.cuhk.edu.hk:8088/proxy/application_1644821811052_3861/
         user: s1155160788
22/03/27 15:45:40 INFO ShutdownHookManager: Shutdown hook called
22/03/27 15:45:40 INFO ShutdownHookManager: Deleting directory /disk1/tmp/spark2/spark-2ae19e2f-7198-4c1b-a4cd-ce627608187f
22/03/27 15:45:40 INFO ShutdownHookManager: Deleting directory /tmp/spark-c1253337-c3a7-4778-9f25-3fd2762164cd
[s1155160788@dicvmd10 Assignment3]$
```

I used “yarn logs -applicationId application_1644821811052_3861 -log_files stdout > output.txt” to extract the logfile.

Screenshot of offense counts below:

```
+--------------------+-----+
|             OFFENSE|count|
+--------------------+-----+
|          THEFT/OTHER|12885|
|         THEFT F/AUTO|10162|
|             ROBBERY| 3982|
|            BURGLARY| 3356|
| MOTOR VEHICLE THEFT| 2660|
|ASSAULT W/DANGERO...| 2397|
|           SEX ABUSE|  297|
|            HOMICIDE|  103|
|               ARSON|   35|
+--------------------+-----+
```

Note: “ASSAULT W/DANGERO...” is actually “ASSAULT W/DANGEROUS WEAPON” but the show() will abridge the name of long columns.

Note 2: The counts may be slightly off because I am still filtering out lines with NULL values in any of the selected columns, as was specified in Section A.

I originally wanted to write to a CSV file like I did in Part 3A, however, after each offense, it repeated the header, as shown in the screenshot below. The reason is because I chose to have a header, and each part of the CSV file has the header, so when I combined the parts together to create the whole CSV file, it also had the header.

1	OFFENSE	count
2	THEFT/OTHER	12885
3	OFFENSE	count
4	THEFT F/AUTO	10162
5	OFFENSE	count
6	ROBBERY	3982
7	OFFENSE	count
8	BURGLARY	3356
9	OFFENSE	count
10	MOTOR VEHICLE THEFT	2660
11	OFFENSE	count
12	ASSAULT W/DANGEROUS WEAPON	2397
13	OFFENSE	count
14	SEX ABUSE	297
15	OFFENSE	count
16	HOMICIDE	103
17	OFFENSE	count
18	ARSON	35

Final Part B Time shift count code:

build.sbt Crime2013B.scala

```scala
package Question3
import org.apache.spark.sql.SparkSession
import org.apache.spark.sql.functions._ // import built in dataframe API functions

object Crime2013B {
  def main (args: Array[String]): Unit = {
    val spark = SparkSession // Create SparkSession
      .builder
      .appName( name = "Crime2013")
      .getOrCreate()
    val filePath = args(0).toString // Takes first argument and stores it as file path.

    val crimesTable = spark.read.format( source = "csv") // read CSV formatted file
      .option("sep", ",") // comma separated
      .option("inferSchema", "true") // infer schema
      .option("header", "true") // csv file has header
      .load(filePath) // load file corresponding to the first argument

    // Select only these 7 columns
    val focusedCrimesTable = crimesTable.select( col = "CCN", cols = "REPORT_DAT", "OFFENSE", "METHOD", "END_DATE", "DISTRICT", "SHIFT")
    val filteredCrimesTable = focusedCrimesTable.na.drop() // Drop rows with NULL values in ANY of these columns

    val offenses = filteredCrimesTable.groupBy( col1 = "OFFENSE") // group by offenses
                  .count() // count each type of offenses
                  .orderBy(desc( columnName = "count")) // order by count in descending order
    // Removed show command for offenses because I am now showing time shift only.

    val timeShift = filteredCrimesTable.groupBy( col1 = "SHIFT") // group by time shift crime occurred on
                  .count() // count each time shift
                  .orderBy(desc( columnName = "count")) // order by count in descending order
```

```
    val offenses = filteredCrimesTable.groupBy( col1 = "OFFENSE") // group by offenses
                   .count() // count each type of offenses
                   .orderBy(desc( columnName = "count")) // order by count in descending order
    // Removed show command for offenses because I am now showing time shift only.

    val timeShift = filteredCrimesTable.groupBy( col1 = "SHIFT") // group by time shift crime occurred on
                   .count() // count each time shift
                   .orderBy(desc( columnName = "count")) // order by count in descending order
    timeShift.show() // Show the most frequently occurring time shift and the count.
  }
}
```

```
22/03/27 16:06:12 INFO Client: Application report for application_1644821811052_3870 (state: RUNNING)
22/03/27 16:06:13 INFO Client: Application report for application_1644821811052_3870 (state: RUNNING)
22/03/27 16:06:14 INFO Client: Application report for application_1644821811052_3870 (state: RUNNING)
22/03/27 16:06:15 INFO Client: Application report for application_1644821811052_3870 (state: FINISHED)
22/03/27 16:06:15 INFO Client:
         client token: N/A
         diagnostics: N/A
         ApplicationMaster host: 172.16.5.206
         ApplicationMaster RPC port: 0
         queue: default
         start time: 1648368335072
         final status: SUCCEEDED
         tracking URL: http://dicvmd1.ie.cuhk.edu.hk:8088/proxy/application_1644821811052_3870/
         user: s1155160788
22/03/27 16:06:15 INFO ShutdownHookManager: Shutdown hook called
22/03/27 16:06:15 INFO ShutdownHookManager: Deleting directory /tmp/spark-9421f87b-1632-45be-8394-8908ec4e74e2
22/03/27 16:06:15 INFO ShutdownHookManager: Deleting directory /disk1/tmp/spark2/spark-08ba48fa-709d-4d88-a6a4-6fcebb1ed93a
[s1155160788@dicvmd10 Assignment3]$
[s1155160788@dicvmd10 Assignment3]$
```

I use “yarn logs -applicationId application_1644821811052_3870 -log_files stdout > output.txt” to extract the stdout log.

```
+--------+-----+
|   SHIFT|count|
+--------+-----+
| EVENING|15073|
|     DAY|14295|
|MIDNIGHT| 6509|
+--------+-----+
```

From this, we can see that the shift where most crimes occur is evening. I confirmed with Chun Fai LEE on Blackboard forums that it is ok to leave the calculation at this, and that we don’t need to calculate the time shift that occurs the most for every type of offense.

Question 3 Section C: Merge tables and compute percentage of gun offense in each year

First I go to http://opendata.dc.gov/datasets?q=crime%20incidents%20 and download each of the “Crime Incidents in 20XX” from 2010 to 2018, CSV format, then transfer it with Filezilla to the server under a new directory.

Assignment3
Question 3C Data

Filename	Filesize	Filetype	Last modifi...	Permissi...	Owner/Gr...
..					
Crime_Incidents_in_2014.csv	11,222,432	Microsof...	3/27/2022 ...	-rw-rw-r--	s1155160...
Crime_Incidents_in_2015.csv	10,856,577	Microsof...	3/27/2022 ...	-rw-rw-r--	s1155160...
Crime_Incidents_in_2016.csv	10,827,232	Microsof...	3/27/2022 ...	-rw-rw-r--	s1155160...
Crime_Incidents_in_2013.csv	10,481,990	Microsof...	3/27/2022 ...	-rw-rw-r--	s1155160...
Crime_Incidents_in_2012.csv	10,306,462	Microsof...	3/27/2022 ...	-rw-rw-r--	s1155160...
Crime_Incidents_in_2018.csv	9,698,590	Microsof...	3/27/2022 ...	-rw-rw-r--	s1155160...
Crime_Incidents_in_2017.csv	9,624,019	Microsof...	3/27/2022 ...	-rw-rw-r--	s1155160...
Crime_Incidents_in_2011.csv	9,610,935	Microsof...	3/27/2022 ...	-rw-rw-r--	s1155160...
Crime_Incidents_in_2010.csv	9,208,076	Microsof...	3/27/2022 ...	-rw-rw-r--	s1155160...

I first copy the files from the Cluster into the HDFS under a new directory called Q3C.

```
[s1155160788@dicvmd10 Question 3C Data]$ ls
Crime_Incidents_in_2010.csv  Crime_Incidents_in_2012.csv  Crime_Incidents_in_2014.csv  Crime_Incidents_in_2016.csv  Crime_Incidents_in_2018.csv
Crime_Incidents_in_2011.csv  Crime_Incidents_in_2013.csv  Crime_Incidents_in_2015.csv  Crime_Incidents_in_2017.csv
[s1155160788@dicvmd10 Question 3C Data]$ hdfs dfs -ls
Found 5 items
drwx------   - s1155160788 student          0 2022-03-27 15:30 .Trash
drwxr-xr-x   - s1155160788 student          0 2022-03-27 16:06 .sparkStaging
-rw-r--r--   3 s1155160788 student   10481990 2022-03-25 15:43 Crime_Incidents_in_2013.csv
-rw-r--r--   3 s1155160788 student      14449 2022-03-22 22:57 test-Google.txt
-rw-r--r--   3 s1155160788 student   75379926 2022-03-22 23:04 web-Google.txt
[s1155160788@dicvmd10 Question 3C Data]$ hdfs dfs -mkdir Q3C
[s1155160788@dicvmd10 Question 3C Data]$ hdfs dfs -copyFromLocal Crime_Incidents_in* Q3C
[s1155160788@dicvmd10 Question 3C Data]$ hdfs dfs -ls Q3C
Found 9 items
-rw-r--r--   3 s1155160788 student    9208076 2022-03-27 17:59 Q3C/Crime_Incidents_in_2010.csv
-rw-r--r--   3 s1155160788 student    9610935 2022-03-27 17:59 Q3C/Crime_Incidents_in_2011.csv
-rw-r--r--   3 s1155160788 student   10306462 2022-03-27 17:59 Q3C/Crime_Incidents_in_2012.csv
-rw-r--r--   3 s1155160788 student   10481990 2022-03-27 17:59 Q3C/Crime_Incidents_in_2013.csv
-rw-r--r--   3 s1155160788 student   11222432 2022-03-27 17:59 Q3C/Crime_Incidents_in_2014.csv
-rw-r--r--   3 s1155160788 student   10856577 2022-03-27 17:59 Q3C/Crime_Incidents_in_2015.csv
-rw-r--r--   3 s1155160788 student   10827232 2022-03-27 17:59 Q3C/Crime_Incidents_in_2016.csv
-rw-r--r--   3 s1155160788 student    9624019 2022-03-27 17:59 Q3C/Crime_Incidents_in_2017.csv
-rw-r--r--   3 s1155160788 student    9698590 2022-03-27 17:59 Q3C/Crime_Incidents_in_2018.csv
[s1155160788@dicvmd10 Question 3C Data]$
```

Based on [36] and [37], I formulate an approach where I will specify "hdfs://dicvmd2.ie.cuhk.edu.hk:8020/user/s1155160788/Q3C/" as the file path, then add "Crimes_Incidents_in_*.csv" to the load command to read in all the files at once.

After reading in the data, I select only REPORT_DAT and METHOD, where the method is gun [38] (DataFrame filter () with SQL Expression section).

Because I was uncertain if this would work, I ran a write CSV statement similar to question 3A to see if it would actually combine.

Screenshot on next page:

2011/11/10 02:58:00+00	GUN
2011/12/23 20:18:00+00	GUN
2011/10/04 19:18:00+00	GUN
REPORT_DAT	METHOD
2010/01/21 05:00:00+00	GUN
2010/01/13 05:00:00+00	GUN
2010/01/04 00:45:00+00	GUN
2010/01/04 04:40:00+00	GUN
2010/01/04 08:00:00+00	GUN
2010/01/14 01:40:00+00	GUN
2010/01/14 05:42:00+00	GUN
2010/01/14 17:20:00+00	GUN

It does seem to combine with the header inserted at the top of each year, and there does not seem to be a particular order in which the files are read, but the resulting file is quite large and I have located entries from every year 2010 – 2018, so I assume the merger went well.

```
root
 |-- REPORT_DAT: string (nullable = true)
 |-- METHOD: string (nullable = true)
```

I also run a version with printSchema on the resulting filtered table and see that the REPORT_DAT is still being interpreted as a string.

```
val yearsTable = filteredCrimesTable.withColumn( colName = "Year", year(to_date(col( colName = "REPORT_DAT"), fmt = "yyyy/MM/dd")))
```

This line deserves some explanation. I was having trouble with extracting the year out of the string, because as far as I know, there is no method built into string to extract the year. [39] and [40] suggested using .withColumn() to create a column named Year that takes the string and uses another function called to_timestamp to convert each row of the REPORT_DAT column from string to timestamp, then apply the year() function on the timestamp to extract the year.

Unfortunately, the timestamp seems to be fairly stringent, and even needs milliseconds, which the string does not have. Based on [41] and [42], I used to_date() instead of to_timestamp() with formatting specified from [42] to convert the initial date section only (Year/Month/Day) to a date, then extract the year from that.

I ran it with show() and print schema() and the output seems as desired.

```
root
 |-- REPORT_DAT: string (nullable = true)
 |-- METHOD: string (nullable = true)
 |-- Year: integer (nullable = true)

+--------------------+------+----+
|          REPORT_DAT|METHOD|Year|
+--------------------+------+----+
|2014/05/07 04:00:...|   GUN|2014|
|2014/05/06 04:00:...|   GUN|2014|
|2014/01/16 04:01:...|   GUN|2014|
|2014/01/16 10:21:...|   GUN|2014|
|2014/01/08 15:57:...|   GUN|2014|
|2014/01/08 19:22:...|   GUN|2014|
|2014/01/08 23:30:...|   GUN|2014|
```

For the calculation of the percentages, I first group by year, and then use .agg(count) to create a count of the gun crimes for each year, referring to [43]. I then create a new column called Percentage using withColumn again and calculate the percentage by taking the number of gun cases, multiplying by 100 [44] and dividing by the total crimes each year, accomplished by using .sum and .over() [43]. Over looks at the total under METHOD rather than the subset from the where method = gun filter.

Using .show(false), I can output the final results without truncation [45].

Final code screenshot on next page:

```scala
1  package Question3
2  import org.apache.spark.sql.SparkSession
3  import org.apache.spark.sql.functions._ // import built in dataframe API functions
4
5  object GunControl {
6    def main (args: Array[String]): Unit = {
7      val spark = SparkSession // Create SparkSession
8        .builder
9        .appName( name = "GunControl")
10       .getOrCreate()
11     val filePath = args(0).toString // Takes first argument and stores it as file path.
12
13     val crimesTable = spark.read.format( source = "csv") // read CSV formatted file
14       .option("sep", ",") // comma separated
15       .option("inferSchema", "true") // infer schema
16       .option("header", "true") // csv file has header
17       .load(filePath + "Crime_Incidents_in_*.csv") // load file corresponding to the first argument
18
19     // Select only REPORT_DAT and records where method of crime is gun
20     val focusedCrimesTable = crimesTable.select( col = "REPORT_DAT", cols = "METHOD")
21                            .where( conditionExpr = "METHOD == 'GUN'")
22     val filteredCrimesTable = focusedCrimesTable.na.drop() // Drop rows with NULL values in ANY of these columns
23
24     // Extract year from the date in each column, make it its own column.
25     val yearsTable = filteredCrimesTable.withColumn( colName = "Year", year(to_date(col( colName = "REPORT_DAT"), fmt = "yyyy/MM/dd")))
26                   .groupBy( col1 = "Year") // group by Year
27                   .agg(count( columnName = "METHOD").alias( alias = "Count")) // Sum up number of gun crimes per year
28                   .withColumn( colName = "Percentage", col( colName = "Count").multiply( other = 100) / sum( columnName = "Count").over())
29     // Output final results.
```

```scala
    // Output final results.
    val yearsTableOutput = yearsTable.select( col = "Year", cols = "Percentage")
                                     .orderBy(asc( columnName = "Year"))
    yearsTableOutput.show( truncate = false)
  }
}
```

I initially had “org.apache.spark.sql.AnalysisException: cannot resolve given input columns” because I had the .alias() command outside of .agg instead of being within, so the aggregation was not aliased to Count, and then in the .withColumn line, it would not find any Count count column to perform operation, summing, or division operations with, leading to the errors. After fixing it to be the current line 27 in the screenshot above, the errors went away.

Final submission command:

“spark-submit \

--class Question3.GunControl \

--master yarn \

--deploy-mode cluster \

hdfs://dicvmd2.ie.cuhk.edu.hk:8020/user/s1155160788/assignment3_2.11-0.1.jar \

hdfs://dicvmd2.ie.cuhk.edu.hk:8020/user/s1155160788/Q3C/”

```
22/03/28 12:58:48 INFO Client: Application report for application_1644821811052_4117 (state: RUNNING)
22/03/28 12:58:49 INFO Client: Application report for application_1644821811052_4117 (state: RUNNING)
22/03/28 12:58:50 INFO Client: Application report for application_1644821811052_4117 (state: RUNNING)
22/03/28 12:58:51 INFO Client: Application report for application_1644821811052_4117 (state: FINISHED)
22/03/28 12:58:51 INFO Client:
         client token: N/A
         diagnostics: N/A
         ApplicationMaster host: 172.16.5.204
         ApplicationMaster RPC port: 0
         queue: default
         start time: 1648443489110
         final status: SUCCEEDED
         tracking URL: http://dicvmd1.ie.cuhk.edu.hk:8088/proxy/application_1644821811052_4117/
         user: s1155160788
22/03/28 12:58:51 INFO ShutdownHookManager: Shutdown hook called
22/03/28 12:58:51 INFO ShutdownHookManager: Deleting directory /tmp/spark-f11f53c6-0299-44a8-b511-428905fcd6be
22/03/28 12:58:51 INFO ShutdownHookManager: Deleting directory /disk1/tmp/spark2/spark-d9b47883-7c7d-4725-a0c6-75031b0757a6
```

I used “yarn logs -applicationId application_1644821811052_4117 -log_files stdout > output.txt” to extract the stdout log and uploaded the text file to my local machine.

Screenshot of final output:

```
+----+------------------+
|Year|Percentage        |
+----+------------------+
|2010|11.437559687657997|
|2011|10.448851188135498|
|2012|12.448738834896917|
|2013|12.375709229818549|
|2014|11.033088028762428|
|2015|12.285826638952868|
|2016|11.960002247064772|
|2017|8.909611819560698 |
|2018|9.100612325150273 |
+----+------------------+
```

On the left is the year, and on the right is the percentage of all the crimes that are gun offenses.

Obama signed a number of executive actions for gun control in January 2016 [46]. We can see the impact of these executive actions as the percentage of crimes involving guns went down from 10.5-12.5% to 8.9-9.1%, significantly less.

References

[1] https://spark.apache.org/docs/latest/spark-standalone.html

[2] https://spark.apache.org/downloads.html

[3] https://www.tutorialspoint.com/apache_spark/apache_spark_installation.htm

[4] https://medium.com/beeranddiapers/installing-scala-on-ubuntu-7a4b14d26bdc

[5] https://stackoverflow.com/questions/70390397/what-is-difference-between-spark-slave-and-spark-worker

[6] https://spark.apache.org/docs/latest/submitting-applications.html

[7] https://www.linode.com/docs/guides/install-configure-run-spark-on-top-of-hadoop-yarn-cluster/

[8] https://stackoverflow.com/questions/37757733/install-spark-on-yarn-cluster

[9] https://spark.apache.org/docs/latest/configuration.html

[10] https://spark.apache.org/docs/2.3.0/submitting-applications.html

[11] https://github.com/ojdkbuild/ojdkbuild/releases/tag/java-1.8.0-openjdk-1.8.0.312-1.b07

[12] https://www.jetbrains.com/idea/download/#section=windows

[13] https://kaizen.itversity.com/setup-development-environment-intellij-and-scala-big-data-hadoop-and-spark/

[14] https://www.learningjournal.guru/courses/spark/spark-foundation-training/how-to-create-and-submit-spark-applications/

[15] https://www.scala-lang.org/download/2.11.8.html

[16] https://www.scala-sbt.org/download.html

[17] https://github.com/apache/spark/blob/master/examples/src/main/scala/org/apache/spark/examples/SparkPageRank.scala

[18] https://mobitec.ie.cuhk.edu.hk/iems5730/static_files/slides/SparkPart1IERG4330IEMS5730Spring2022.pdf

[19] https://sparkbyexamples.com/spark/spark-read-text-file-rdd-dataframe/

[20] https://stackoverflow.com/questions/41918826/how-to-order-my-tuple-of-spark-results-descending-order-using-value

[21] https://docs.scala-lang.org/overviews/core/string-interpolation.html

[22] https://stackoverflow.com/questions/27478096/cannot-read-a-file-from-hdfs-using-spark

[23] https://stackoverflow.com/questions/46759790/out-of-bound-exception-in-spark-using-scala

[24] https://stackoverflow.com/questions/23058663/where-are-logs-in-spark-on-yarn

[25] https://stackoverflow.com/questions/23811425/scala-convert-string-to-int-or-none

[26] https://docs.scala-lang.org/overviews/scala-book/two-types-variables.html

[27] https://stackoverflow.com/questions/31424396/how-does-hashpartitioner-work

[28] https://mobitec.ie.cuhk.edu.hk/iems5730/static_files/slides/SparkSQLIERG4330IEMS5730Spring2022.pdf

[29] https://spark.apache.org/docs/2.3.0/sql-programming-guide.html

[30] https://sparkbyexamples.com/spark/spark-sql-functions/

[31] https://stackoverflow.com/questions/56927329/spark-option-inferschema-vs-header-true

[32] https://sparkbyexamples.com/spark/spark-dataframe-drop-rows-with-null-values/

[33] https://sparkbyexamples.com/spark/show-top-n-rows-in-spark-pyspark/

[34] https://sparkbyexamples.com/spark/spark-write-dataframe-to-csv-file/

[35] https://stackoverflow.com/questions/30332619/how-to-sort-by-column-in-descending-order-in-spark-sql

[36] https://stackoverflow.com/questions/69350586/pyspark-read-multiple-csv-files-at-once

[37] https://dev.to/sridharanprasanna/using-wildcards-for-folder-path-with-spark-dataframe-load-4jo7

[38] https://sparkbyexamples.com/spark/spark-dataframe-where-filter/

[39] https://stackoverflow.com/questions/57100885/how-to-extract-year-from-a-date-string

[40] https://sparkbyexamples.com/spark/spark-dataframe-withcolumn/

[41] https://sparkbyexamples.com/spark/convert-string-to-date-format-spark-sql/

[42] https://spark.apache.org/docs/latest/sql-ref-datetime-pattern.html

[43] https://stackoverflow.com/questions/46863360/spark-count-percentage-percentages-of-a-column-values

[44] https://stackoverflow.com/questions/46609524/multiplying-a-column-in-a-spark-dataframe-by-a-constant-value

[45] https://stackoverflow.com/questions/33742895/how-to-show-full-column-content-in-a-spark-dataframe

[46] https://obamawhitehouse.archives.gov/the-press-office/2016/01/04/fact-sheet-new-executive-actions-reduce-gun-violence-and-make-our