Assignment 6
CSCI E 63 - Big Data Analytics

Problem 1:
Lecture notes contain script *network-count.py* in both Spark Streaming API and Spark Structured Streaming API. Use Linux *nc* (*NetCat*) utility to demonstrate that scripts work. Run both scripts on your own VM with Spark 2.2 installation. Cloudera VM with Spark 1.6 does not have Spark Structured Streaming API.

Answer:

→ Download network-count.py in both Spark Streaming API and Spark Structured Streaming API

```
[kbhandarkar@localhost Assignment6]$ ls -ltr
total 8
-rw-rw-r--. 1 kbhandarkar kbhandarkar 666 Oct 12 22:47 network-count.py
-rw-rw-r--. 1 kbhandarkar kbhandarkar 688 Oct 12 22:47 network-count-structured.py
```

→ Run in Spark Streaming API and validate using NetCat
Run command: spark-submit --master local[4] network-count.py localhost 9999

```
-------------------------------------------

-------------------------------------------
Time: 2017-10-12 23:08:09
-------------------------------------------

-------------------------------------------
Time: 2017-10-12 23:08:12
-------------------------------------------
(u'this', 1)
(u'let', 1)
(u'use', 1)
(u'us', 1)
(u'for', 1)
(u'testing', 1)

-------------------------------------------
Time: 2017-10-12 23:08:15
-------------------------------------------
(u'and', 1)
(u'some', 1)
(u'testing', 1)
(u'more', 1)

-------------------------------------------
Time: 2017-10-12 23:08:18
-------------------------------------------

-------------------------------------------
Time: 2017-10-12 23:08:21
-------------------------------------------
(u'and', 1)
(u'even', 1)
(u'testing', 1)
(u'more', 1)
```

```
[kbhandarkar@localhost ~]$ nc -lk 9999
let us use this for testing
and some more testing
and even more testing
```

Assignment 6
CSCI E 63 - Big Data Analytics

→ Run in Spark Structured Streaming API and validate using NetCat

Run command: spark-submit --master local[4] network-count-structured.py localhost 9999

```
Batch: 0
-------------------------------------------
+-------+-----+
|   word|count|
+-------+-----+
|testing|    1|
|    and|    1|
|     we|    1|
|   keep|    1|
+-------+-----+

-------------------------------------------
Batch: 1
-------------------------------------------
+-------+-----+
|   word|count|
+-------+-----+
|   more|    2|
|testing|    2|
|   just|    1|
|    and|    2|
|     we|    1|
|   keep|    1|
+-------+-----+

-------------------------------------------
Batch: 2
-------------------------------------------
+-------+-----+
|   word|count|
+-------+-----+
|   more|    2|
|testing|    3|
|   just|    2|
|    and|    2|
|     we|    1|
|   keep|    2|
+-------+-----+
```

```
and we keep testing
just more and more testing
just keep testing
```

→ Understanding the command

Master local[4] - Tells PySpark to start the master with 4 threads. This is to prevent it from shutting down. If VM has 4 CPUs allocated, it would by default start 4 threads.
Localhost and 9999 - The two parameters the scripts expect to be passed.

Karan A. Bhandarkar

Assignment 6
CSCI E 63 - Big Data Analytics

Problem 2:
Expand provide orders.tar.gz file. Also, download shell scrips splitAndSend.original.sh and splitAndSend.sh and the Python script count-buys.py. First run splitAndSend.original.sh and count-buys.py. Record the failure mode of count-buys.py. Simply read the error message produced and tell us what is happening. Then run script splitAndSend.sh and Python program count-buys.py and tell us what the results are. In both cases show use contents of your HDFS directories input, output and staging. The second script splitAndSend.sh is supposed to reduce or eliminate the race condition. You might want to rename HDFS directory output from the first run in order to preserve it's content. In both cases, show the partial contents of your HDFS directories input, output and staging. In the second run, locate an output file named part-00000 that is not empty and show its content to us. Run these experiments on Cloudera VM. You need HDFS for these programs to run.

Answer:

→ Download the required files

```
[cloudera@quickstart Problem2]$ ls -ltr
total 26932
-rw-r--r-- 1 cloudera cloudera 22817161 Feb 24  2017 orders.txt
-rw-rw-r-- 1 cloudera cloudera  4735092 Oct 12 20:57 orders.tar.gz
-rw-rw-r-- 1 cloudera cloudera      326 Oct 12 20:57 splitAndSend.original.sh
-rw-rw-r-- 1 cloudera cloudera      426 Oct 12 20:57 splitAndSend.sh
-rw-rw-r-- 1 cloudera cloudera      988 Oct 12 20:58 count-buys-1.py
drwxrwxr-x 2 cloudera cloudera     4096 Oct 12 21:00 input
drwxrwxr-x 2 cloudera cloudera     4096 Oct 12 21:00 output
```

→ Create input, output and staging directories in the hdfs

Run commands:

```
sudo -u hdfs hadoop fs -mkdir /user/cloudera/input
sudo -u hdfs hadoop fs -mkdir /user/cloudera/output
sudo -u hdfs hadoop fs -mkdir /user/cloudera/staging
```

Validate using command:

```
sudo -u hdfs hdfs dfs -ls -R /user/cloudera
```

```
drwxr-xr-x   - hdfs     cloudera          0 2017-10-13 10:48 /user/cloudera/input
drwxr-xr-x   - hdfs     cloudera          0 2017-10-13 10:52 /user/cloudera/output
drwxr-xr-x   - hdfs     cloudera          0 2017-10-13 10:52 /user/cloudera/staging
```

Assignment 6
CSCI E 63 - Big Data Analytics

→ First run splitAndSend.original.sh and count-buy.py

To run splitAndSend.original.sh, run commands:

chmod +x splitAndSend.original.sh
./splitAndSend.original.sh input

To run count-buy.py, run command:

spark-submit --master local[4] count-buys-1.py

To verify the contents in Input folder, run command:

sudo -u hdfs hdfs dfs -ls -R /user/cloudera/input | head -5

```
[cloudera@quickstart Problem2]$ sudo -u hdfs hdfs dfs -ls -R /user/cloudera/input | head -5
-rw-r--r--   1 cloudera cloudera     437626 2017-10-13 11:04 /user/cloudera/input/chunkaa
-rw-r--r--   1 cloudera cloudera     448647 2017-10-13 11:04 /user/cloudera/input/chunkab
-rw-r--r--   1 cloudera cloudera     448605 2017-10-13 11:04 /user/cloudera/input/chunkac
-rw-r--r--   1 cloudera cloudera     448794 2017-10-13 11:04 /user/cloudera/input/chunkad
-rw-r--r--   1 cloudera cloudera     448624 2017-10-13 11:05 /user/cloudera/input/chunkae
```

→ Error message produced and explanation

Error message:

17/10/13 11:09:36 INFO spark.SparkContext: Created broadcast 4 from textFileStream at NativeMethodAccessorImpl.java:-2
17/10/13 11:09:36 ERROR scheduler.JobScheduler: Error generating jobs for time 1507831776000 ms
org.apache.hadoop.mapreduce.lib.input.InvalidInputException: Input path does not exist: hdfs://quickstart.cloudera:8020/user/cloudera/input/chunkac._COPYING_ at org.apache.hadoop.mapreduce.lib.input.FileInputFormat.singleThreadedListStatus(FileInputFormat.java:323)

Explanation:
Copying directly from local file system to the Hadoop "input" directory ($1 argument of the script) there are cases where Spark tries to read the file right after it's name was changed. It turns out that while the file is being copied, it has an extension that is something like "chunkac._COPYING_" .
When the file is finished copying, the extension is changed. So, if Spark sees the new temp file and then tries to read it just after its name was changed, it causes a File not Found Error and the script exits.

Karan A. Bhandarkar

Assignment 6
CSCI E 63 - Big Data Analytics

→ Contents of staging and output directory

Run commands:

sudo -u hdfs hdfs dfs -ls -R /user/cloudera/staging | head -5

sudo -u hdfs hdfs dfs -ls -R /user/cloudera/output | head -5

```
[cloudera@quickstart Problem2]$ sudo -u hdfs hdfs dfs -ls -R /user/cloudera/output | head -5
drwxr-xr-x   - cloudera cloudera          0 2017-10-13 11:26 /user/cloudera/output/output-1507919175000.txt
-rw-r--r--   1 cloudera cloudera          0 2017-10-13 11:26 /user/cloudera/output/output-1507919175000.txt/_SUCCESS
-rw-r--r--   1 cloudera cloudera          0 2017-10-13 11:26 /user/cloudera/output/output-1507919175000.txt/part-00000
drwxr-xr-x   - cloudera cloudera          0 2017-10-13 11:27 /user/cloudera/output/output-1507919184000.txt
-rw-r--r--   1 cloudera cloudera          0 2017-10-13 11:27 /user/cloudera/output/output-1507919184000.txt/_SUCCESS
[cloudera@quickstart Problem2]$ sudo -u hdfs hdfs dfs -ls -R /user/cloudera/staging | head -5
```

→ Rename the Input and Output directory for second part and recreate the folders

Run commands:

sudo -u hdfs hadoop fs -mv /user/cloudera/input /user/cloudera/input1

sudo -u hdfs hadoop fs -mv /user/cloudera/output /user/cloudera/output1

sudo -u hdfs hadoop fs -mkdir /user/cloudera/input

sudo -u hdfs hadoop fs -mkdir /user/cloudera/output

Verify:

sudo -u hdfs hadoop fs -ls /user/cloudera/

```
drwxr-xr-x   - hdfs     cloudera          0 2017-10-13 13:39 /user/cloudera/input
drwxr-xr-x   - hdfs     cloudera          0 2017-10-13 11:14 /user/cloudera/input1
drwxr-xr-x   - hdfs     cloudera          0 2017-10-13 13:39 /user/cloudera/output
drwxr-xr-x   - hdfs     cloudera          0 2017-10-13 13:06 /user/cloudera/output1
drwxr-xr-x   - hdfs     cloudera          0 2017-10-13 13:35 /user/cloudera/staging
```

→ Now run splitAndSend.sh and count-buy.py

To run splitAndSend.sh, run commands:

chmod +x splitAndSend.sh

./splitAndSend.sh input

To run count-buy.py, run command:

spark-submit --master local[4] count-buys-1.py

→ Check contents of input, staging and output directory

Run commands:

sudo -u hdfs hdfs dfs -ls -R /user/cloudera/input | head -5

sudo -u hdfs hdfs dfs -ls -R /user/cloudera/staging | head -5

sudo -u hdfs hdfs dfs -ls -R /user/cloudera/output | head -5

Karan A. Bhandarkar

Assignment 6
CSCI E 63 - Big Data Analytics

```
[cloudera@quickstart input]$ sudo -u hdfs hdfs dfs -ls -R /user/cloudera/input | head -5
-rw-r--r--   1 cloudera cloudera     437626 2017-10-13 13:43 /user/cloudera/input/chunkaa
-rw-r--r--   1 cloudera cloudera     448647 2017-10-13 13:44 /user/cloudera/input/chunkab
-rw-r--r--   1 cloudera cloudera     448605 2017-10-13 13:45 /user/cloudera/input/chunkac
-rw-r--r--   1 cloudera cloudera     448794 2017-10-13 13:46 /user/cloudera/input/chunkad
-rw-r--r--   1 cloudera cloudera     448624 2017-10-13 13:46 /user/cloudera/input/chunkae
[cloudera@quickstart input]$ sudo -u hdfs hdfs dfs -ls -R /user/cloudera/staging | head -5
[cloudera@quickstart input]$ sudo -u hdfs hdfs dfs -ls -R /user/cloudera/output | head -5
drwxr-xr-x   - cloudera cloudera          0 2017-10-13 13:44 /user/cloudera/output/output-1507927473000.txt
-rw-r--r--   1 cloudera cloudera          0 2017-10-13 13:44 /user/cloudera/output/output-1507927473000.txt/_SUCCESS
-rw-r--r--   1 cloudera cloudera          0 2017-10-13 13:44 /user/cloudera/output/output-1507927473000.txt/part-00000
drwxr-xr-x   - cloudera cloudera          0 2017-10-13 13:44 /user/cloudera/output/output-1507927482000.txt
-rw-r--r--   1 cloudera cloudera          0 2017-10-13 13:44 /user/cloudera/output/output-1507927482000.txt/_SUCCESS
```

This time the race condition is eliminated.

→ Check contents of an output file

Run command:

hadoop fs -cat /user/cloudera/output/output-1507927473000.txt/part-00000 | head -20

```
[cloudera@quickstart input]$ hadoop fs -cat /user/cloudera/output/output-1507927482000.txt/part-00000 | head -20
[cloudera@quickstart input]$ hadoop fs -cat /user/cloudera/output/output-1507927491000.txt/part-00000 | head -20
[cloudera@quickstart input]$ hadoop fs -cat /user/cloudera/output/output-1507927851000.txt/part-00000 | head -20
[cloudera@quickstart input]$ hadoop fs -cat /user/cloudera/output/output-1507927860000.txt/part-00000 | head -20
[cloudera@quickstart input]$ hadoop fs -cat /user/cloudera/output/output-1507927842000.txt/part-00000 | head -20
(False, 4932L)
(True, 5068L)
```

We can see that many of these files are empty.

Karan A. Bhandarkar

Assignment 6
CSCI E 63 - Big Data Analytics

Problem 3:
In the second run of the previous problem you will notice that many of part-00000 files in your output directory are empty. Could you explain why.

Answer:

→ Many of part-00000 files in the output directory are empty, but this behavior is expected.

count-buys.py is expecting a continuous stream of data in the input folder, but that is not the case. Delays in the splitAndSend.sh script processing, including the sleep time, create a lag between stream generation. Since count-buys.py does not see data in that interval, it generates empty part-00000 files.

Karan A. Bhandarkar

Assignment 6
CSCI E 63 - Big Data Analytics

Problem 4:
Could you rewrite `count-buys.sh` in Spark Structured Streaming API. If you do that change script `splitAndSend.sh` to move generated chunks from the local files system directory staging to local file system directory input. Run this experiment on your VM with Spark 2.2.

Answer:

→ Change script `splitAndSend.sh` to move generated chunks from the local files system directory staging to local file system directory input.

Replace the code in the script for a local parameter passed as(Separately submitted):

```
if [ "$2" == "local" ]; then
   mv $f $1/staging
   sleep 3
   mv $1/staging/$f $1/input/
   rm -f $f
```

Create local input, output and staging folders:

```
drwxrwxr-x. 2 kbhandarkar kbhandarkar 6 Oct 14 13:01 output
drwxrwxr-x. 2 kbhandarkar kbhandarkar 6 Oct 14 13:22 staging
drwxrwxr-x. 2 kbhandarkar kbhandarkar 6 Oct 14 13:31 input
```

→ Rewrite count-buys.sh in Spark Structured Streaming API

Code(Separately submitted as count-buys-structured.py):

```
spark = SparkSession.builder.appName("SparkStructuredStreamingCountBuys").getOrCreate()
buyCountSchema = StructType().add("time", TimestampType()) \
        .add("orderId", IntegerType()) \
        .add("clientId", IntegerType()) \
        .add("symbol", StringType()) \
        .add("numberOfStocks", IntegerType()) \
        .add("price", FloatType()) \
        .add("buy", StringType())
data = spark.readStream.schema(buyCountSchema).csv("file:///home/kbhandarkar/PythonProjects/
Assignment6/Problem4/input")
buys = data.groupBy("buy").count()
query = buys.writeStream.outputMode("complete").format("console").start()
query.awaitTermination()
```

Assignment 6
CSCI E 63 - Big Data Analytics

→ Test the two new scripts

To run splitAndSend.sh, run commands:

chmod +x splitAndSend.sh

./splitAndSend.sh Problem4 local

To run count-buys-structured.py, run command:

spark-submit --master local[4] count-buys-structured.py

Console screenprint:

```
[kbhandarkar@localhost Assignment6]$ spark-submit --master local[4] count-buys-structured.py
-------------------------------------------
Batch: 0
-------------------------------------------
+---+-----+
|buy|count|
+---+-----+
|  B|20165|
|  S|19835|
+---+-----+

-------------------------------------------
Batch: 1
-------------------------------------------
+---+-----+
|buy|count|
+---+-----+
|  B|60431|
|  S|59569|
+---+-----+

-------------------------------------------
Batch: 2
-------------------------------------------
+---+-----+
|buy|count|
+---+-----+
|  B|80508|
|  S|79492|
+---+-----+

-------------------------------------------
Batch: 3
-------------------------------------------
+---+-----+
|buy|count|
+---+-----+
|  B|95471|
|  S|94529|
+---+-----+

-------------------------------------------
Batch: 4
-------------------------------------------
+---+------+
|buy| count|
+---+------+
|  B|105564|
|  S|104436|
+---+------+
```

Karan A. Bhandarkar

Assignment 6
CSCI E 63 - Big Data Analytics

Problem 5:
Examine provided Python program stateful_wordcount.py. Make it work as is. If there are errors on the code, fix them. Modify the code so that it outputs the number of words starting with letters a and b. Demonstrate that modified program work. You should provide several both positive and negative examples.

Answer:

→ Run stateful_wordcount.py
Run command:

spark-submit --master local[4] stateful_wordcount.py localhost 9999

```
[cloudera@quickstart Problem5]$ spark-submit --master local[4] stateful_wordcount.py
  File "/home/cloudera/workspace/Assignment6/Problem5/stateful_wordcount.py", line 1
    """
    ^
IndentationError: unexpected indent
```

After removing the indent from the first line:

```
[cloudera@quickstart ~]$ nc -lk 9999
hi there
we are back to using spark 1.6
I prefer spark 2.2
```

```
-------------------------------------------
Time: 2017-10-14 11:14:34
-------------------------------------------
(u'I', 1)
(u'prefer', 1)
(u'are', 1)
(u'2.2', 1)
(u'back', 1)
(u'using', 1)
(u'to', 1)
(u'we', 1)
(u'spark', 2)
(u'1.6', 1)
...

-------------------------------------------
Time: 2017-10-14 11:14:35
-------------------------------------------
(u'I', 1)
(u'prefer', 1)
(u'back', 1)
(u'2.2', 1)
(u'are', 1)
(u'using', 1)
(u'to', 1)
(u'we', 1)
(u'spark', 2)
(u'1.6', 1)
...

-------------------------------------------
Time: 2017-10-14 11:14:36
-------------------------------------------
(u'I', 1)
(u'prefer', 1)
(u'are', 1)
(u'2.2', 1)
(u'back', 1)
(u'using', 1)
(u'to', 1)
(u'we', 1)
(u'spark', 2)
(u'1.6', 1)
...
```

Assignment 6
CSCI E 63 - Big Data Analytics

→ Modify the code so that it outputs the number of words starting with letters a and b.

Add a filter as below(Separately submitted):

```
running_counts = lines.flatMap(lambda line: line.split(" "))\
                 .filter(lambda x:x[0].startswith('a') or x[0].startswith('b'))\
                 .map(lambda word: (word, 1))\
                 .updateStateByKey(updateFunc)
```

→ Demonstrate that modified program work.

```
[cloudera@quickstart ~]$ nc -lk 9999
hi there
we are back to using spark 1.6
I prefer spark 2.2
Now we are testing filters on these
This could be useful
It is amazing
```

```
-------------------------------------------
Time: 2017-10-14 11:22:32
-------------------------------------------
(u'are', 1)

-------------------------------------------
Time: 2017-10-14 11:22:33
-------------------------------------------
(u'be', 1)
(u'are', 1)

-------------------------------------------
Time: 2017-10-14 11:22:34
-------------------------------------------
(u'be', 1)
(u'are', 1)

-------------------------------------------
Time: 2017-10-14 11:22:35
-------------------------------------------
(u'be', 1)
(u'are', 1)

-------------------------------------------
Time: 2017-10-14 11:22:36
-------------------------------------------
(u'be', 1)
(u'are', 1)

-------------------------------------------
Time: 2017-10-14 11:22:37
-------------------------------------------
(u'be', 1)
(u'are', 1)
(u'amazing', 1)

-------------------------------------------
```

Karan A. Bhandarkar