STSCI 5065 - Spring 2019 HOMEWORK 2

This assignment intends to give you an idea about the HDFS, Ambari and hands on experience with Hortonworks Hadoop Sandbox and Hadoop streaming.

Logistics:

- This assignment needs to be completed individually and is of total 100 marks.
- The assignment is due by March 11, 2019 at 11:59 PM.
- You will submit a zip file named as "**homework2_submission.zip**" containing the below
  - **solutions.pdf** - This contains all your answers including screenshots with your **NetID** and **Name** in the top right corner.
  - **screenshots** - Attach the screenshots in **solutions.pdf** below the answer. Do not attach them separately as they will not be accepted that way.
  - **README.txt (optional)** - It consists of anything which you want us to be aware of while grading and name/link of the sources from where you may have referred while solving the assignment.
  - **feedback.txt (optional)** - Any feedback you want to provide about the assignment.
- For this assignment you should work on Hortonworks Hadoop Sandbox 2.5 or 2.6.1.

Cornell's academic integrity policy is enforced. Anyone caught cheating or plagiarizing will be handled according to Cornell's Code of Academic Integrity.

Please follow the instructions stated in the questions and above; otherwise, you may be subject to unwanted penalties.

Question 1 (total 18 points)

Let's assume that there exists a 5 GB file. For all below sub-parts, assume that transfer rate is 50 MB/s and seek time is 3 milliseconds. Remember from class, seek time is the time to find the start position of Each block to read the data from. For simplicity, assume that 1GB = 1000 MB, 1 MB = 1000 KB and so on, and the complete file is stored on just one system.

a) If the file is stored on a simple file system with a block size of 100 KB, how much time does it take to read the complete file? (5 points)
b) If the file is stored on a Hadoop file system with a block size of 100 MB, how much time does it take to read the complete file? (5 points)
c) What is the percentage gain of time in the second case as compared to the first case? (5 points)
d) Give reasons and situations where block size in Question 1a is better than block size in Question 1b, and vice versa. (3 points)

Question 2 (total 12 points)

In this question we will be working with the command line interface on the Hortonworks Sandbox and use commands for working in the Hadoop ecosystem. Remember the username to login is "root" and password is "Hadoop"(for the first time). For each of the below sub-question, write the command you executed to perform the task.

a) Create an HDFS directory called **"course-data"** at the root of the Hadoop filesystem.
b) Create a normal directory **"data-set"** at the root of the normal Linux file system.
c) Download the files from the following locations in the directory data-set and unzip it. It may take some time to download them.
   http://data.gdeltproject.org/events/1990.zip
   http://data.gdeltproject.org/events/1991.zip
d) After extraction of the above files, copy the files into Hadoop filesystem.
e) Write the command to validate if the files are actually copied correctly in the system.
f) Write the command to examine the file storage statistics of your Hadoop filesystem for the directory course-data. How many block(s) is/are allocated for it? Attach a screenshot of the command's output.

Question 3 (total 4 points)

Remember in Question 2, we copied two files in Hadoop file system. For the below questions, you will use file browser, via Apache Ambari, to view the files and perform operation.

a) Attach the screenshot of the file browser showing the two files copied in Hadoop filesystem. The screenshot should show the file size and permissions as well.

b) Delete the file "1990.csv" using the file browser. Attach the screenshot of the file browser after the operation. Please write down the Hadoop command that performs the same operation.

Question 4 (total 66 points)

In this question, you will practice data analysis using MapReduce in Hadoop with the streaming method. You will modify the mapper and reducer programs written in Python to make them more general and robust and to add more functionality so that they do more than just counting the word frequencies. In Hortonworks Sandbox, in CentOS (through localhost:4200), create a new directory called **HW2Q4** at the root, and save all the related files in this directory.

You are required to use vi to code your Python map and reduce programs. In your code, you must use comments to include your name at the beginning of each program file and to explain what your code blocks do in the program, such as the meaning of the variables you define and the task that your "for loop" performs, etc. These comments are very important for the grader to understand your code. **If the comments are missing (or too simple), up to 10 points may be taken away.** You are required to list all the Linux or Hadoop commands and steps you use to perform the tasks below. Your Python map and reduce programs should be saved to separate files with a **.py** extension.

a) **(30 points)** The map and the reduce programs (**WDmapper.py** and **WDreducer.py**) introduced in the class produced the desired results when we tested them with some simple and short

texts; however, when we apply these to complicated texts, you will find problems that some words are pre-fixed and/or post-fixed with punctuation marks, parentheses and quotes, etc. and the same words are treated as different words.

Modify **WDmapper.py** and **WDreducer.py** to fix the above problems. The input text file to use is the **shakespeare.txt** file on Blackboard. First, test your WDmapper.py and WD reducer.py files with Linux shell scripting by forming a pipeline; attach a screenshot of this result. Second, run your mapper and reducer in HDFS; download the reducer output file, **part-00000**, from the HDFS system to your host OS (Windows OS or Mac OS) and then rename it to "**Q4a_Reducer-output.txt.**" Paste this in your solution file. Additionally, submit **WDmapper.py, WDreducer.py** and **Q4a_Reducer-output.txt** files.

b) **(26 points)** Further modify **WDmapper.py** and **WDreducer.py** and rename them to **WDmapper1.py** and **WDreducer1.py** respectively so that you only output the following (you should not output the whole word list and word frequencies).
   1) The total number of the lines of shakespeare.txt.
   2) The 100 most frequently used words in shakespeare.txt, including the words and their counts. (Hint: you should consider using the Python lambda operator/function and the sort() function)
   3) The total number of the words in shakespeare.txt.
   4) The number of unique words in the shakespeare.txt.

   Here the same words with different cases are treated as different words, e.g., "University" and "university" are different words. Again, your output words should only contain the words themselves, i.e., no punctuation marks, quotes, etc. at the beginning and/or the end (same below). Download the reducer output file, **part-00000**, from the HDFS system to your host OS and then rename it to "**Q4b_Reducer-output.txt**.". Paste this in your solution file. Additionally, submit **WDmapper1.py**, **WDreducer1.py** and **Q4b_Reducer-output.txt** files.

c) **(10 points)** If the case difference of the words is ignored, modify your above Python programs and rename them to **WDmapper2.py** and **WDreducer2.py**. Repeat similar sub-questions as in b):
   1) The total number of the lines of shakespeare.txt.
   2) The 100 most frequently used words in shakespeare.txt, including the words and their counts.
   3) The total number of the words of shakespeare.txt. Here you use the same case for all the words in the text.
   4) The number of unique words in the shakespeare.txt.

   Download the reducer output file, **part-00000**, from the HDFS system to your host OS and then rename it to "**Q4c_Reducer-output.txt**." Paste the output in your solution file. Additionally, submit **WDmapper2.py**, **WDreducer2.py** and **Q4c_Reducer-output.txt** files.

As an example, your output should be something as shown on the next page.

```
There are 2621 lines in the text.

The 30 most frequently used words are:

('AND', 490)
('THE', 431)
('TO', 408)
('MY', 394)
('OF', 370)
('I', 344)
('IN', 323)
('THAT', 322)
('THY', 287)
('THOU', 235)
('WITH', 181)
('FOR', 171)
('IS', 168)
('NOT', 167)
('ME', 164)
('A', 163)
('BUT', 163)
('LOVE', 162)
('THEE', 162)
('SO', 145)
('BE', 142)
('AS', 121)
('ALL', 117)
('YOU', 112)
('IT', 111)
('WHICH', 108)
('HIS', 107)
('WHEN', 106)
('THIS', 105)
('YOUR', 100)

There are  17741 words in the text.

If the case difference is ignored, there are 3363 unique words in the text.
```