Electrical and Computer Engineering, Purdue University Northwest
Big Data (ECE59500/ECE49500)
Hands-on Assignment 2

Task 1 [20 points] Hadoop installation and set-up. You **must have** to add screenshots in the report for the output of underlined text in orange color to get full score for Task 1.

1. Open the terminal
2. Install Java: `sudo apt-get install openjdk-8-jdk`
   You can confirm the installation by typing the command**:** `java -version`
3. Install SSH server: `sudo apt-get install openssh-server`
4. Download Hadoop 2.10.1 from the following link:
   https://mirrors.koehn.com/apache/hadoop/common/hadoop-2.10.1/hadoop-2.10.1.tar.gz
5. Go to the Downloads directory: `cd Downloads`
6. Extract Hadoop files from the downloaded file: `tar xzvf hadoop-2.10.1.tar.gz`
7. Move extracted Hadoop directory to /usr/local directory:
   `sudo mv hadoop-2.10.1 /usr/local/hadoop`
   Change ownership of `/usr/local/hadoop` directory:
   `sudo chown –R bigdata:bigdata /usr/local/hadoop`
   Note: **`bigdata`** is a user here. It may be different in your case. Use command **`whoami`** to find the user name.
   Use the required command that will show the directory with its new owner.
8. Find Java installation directory: `readlink -f /usr/bin/java | sed "s:/bin/java::"`
   The command output will be the directory for Java.
9. Go to home directory: `cd`
10. Open .bashrc file: `gedit .bashrc`
    Add following lines:
    ```
    export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64/jre
    export HADOOP_HOME=/usr/local/hadoop
    export PATH=$PATH:$HADOOP_HOME/bin:$HADOOP_HOME/sbin
    ```
    Note: Value for `JAVA_HOME` is the output of the previous command for finding Java directory
    Close .bashrc file and execute command: `. .bashrc`
    You can verify the changes with the command: `echo $JAVA_HOME`
11. Open /usr/local/hadoop/etc/hadoop/hadoop-env.sh:
    `gedit /usr/local/hadoop/etc/hadoop/hadoop-env.sh`
    Replace `export JAVA_HOME=${JAVA_HOME}` with
    `export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64/jre`
    Also add this: `export HADOOP_OPTS=-Djava.net.preferIPv4Stack=true`
    Close the file
12. Create a directory /app/hadoop/tmp: `sudo mkdir -p /app/hadoop/tmp`
    Change ownership: `sudo chown -R bigdata:bigdata /app/hadoop/tmp`
    Use the required command that will show the directory with its new owner
13. Open /usr/local/hadoop/etc/hadoop/core-site.xml:
    `gedit /usr/local/hadoop/etc/hadoop/core-site.xml`

Add these lines inside *<configuration> </configuration>* tags

```
<property>
<name>hadoop.tmp.dir</name>
<value>/app/hadoop/tmp</value>
</property>
<property>
<name>fs.default.name</name>
<value>hdfs://localhost:9000</value>
</property>
```

Close the file

Use a command that will show the output of this file in terminal.

14. Open /usr/local/hadoop/etc/hadoop/hdfs-site.xml:
    `gedit /usr/local/hadoop/etc/hadoop/hdfs-site.xml`
    Add these lines inside *<configuration> </configuration>* tags

```
<property>
<name>dfs.replication</name>
<value>1</value>
</property>
```

    Close the file.
    Use a command that will show the output of this file in terminal.

15. Open /usr/local/hadoop/etc/hadoop/mapred-site.xml:
    `gedit /usr/local/hadoop/etc/hadoop/mapred-site.xml`
    Add these lines inside *<configuration> </configuration>* tags

```
<property>
<name>mapreduce.framework.name</name>
<value>yarn</value>
</property>
```

    Note: mapred-site.xml is not available. Rename mapred-site.xml.template file to mapred-site.xml file: `mv /usr/local/hadoop/etc/hadoop/mapred-site.xml.template /usr/local/hadoop/etc/hadoop/mapred-site.xml`
    Close the file.
    Use a command that will show the output of this file in terminal.

16. Open /usr/local/hadoop/etc/hadoop/yarn-site.xml:
    gedit /usr/local/hadoop/etc/hadoop/yarn-site.xml
    Add these lines inside *<configuration> </configuration>* tags

```
<property>
<name>yarn.nodemanager.aux-services</name>
<value>mapreduce_shuffle</value>
</property>
```

    Close the file.
    Use a command that will show the output of this file in terminal.

17. Perform following commands to set-up SSH public key authentication
    `ssh-keygen -t rsa -P ""`
    `cat /home/bigdata/.ssh/id_rsa.pub >> /home/bigdata/.ssh/authorized_keys`
    Note: **`bigdata`** is a user here. It may be different in your case.
18. Format HDFS: **`hadoop namenode -format`**
19. Start service: **`start-all.sh`**
20. Execute command to check services are running: **`jps`**
21. Once your job is done, you can stop services: **`stop-all.sh`**
    Do not execute this command now, first do something in HDFS cluster as described below.

HDFS commands:

1. Go to your home directory: **`cd`**
2. Examine files in HDFS cluster: **`hdfs dfs –ls /`**
   You will see nothing in the output as you have not created anything yet.
3. Create a directory in HDFS cluster: `hdfs dfs -mkdir /user`
4. Examine files again in HDFS cluster: `hdfs dfs –ls /`
   You will see the listing of `/user` directory in the output
5. Create a directory inside `/user` directory using your user name:
   `hdfs dfs -mkdir /user/bigdata`
6. Examine that directory in HDFS cluster: `hdfs dfs -ls /user/`
7. Create a text file, say **demo.txt**, use gedit and write something in it and save it.
8. Upload the file in the cluster**:**
   `hdfs dfs -copyFromLocal demo.txt /user/bigdata/`
9. List the file in the cluster: `hdfs dfs -ls /user/bigdata/`
10. See the content of file: `hdfs dfs -cat /user/bigdata/demo.txt`

Task 2 [60 points] Map-Reduce Programming

Note: Find the instructions for running the programs at the end. For full score, your report should have the code and its output for verification of each problem. Also, you need to separately upload mapper and reducer code files so that the instructor may run them if needed.

A. [**15 points**] Write Mapper and Reducer programs that read **passage.txt** file from HDFS cluster and find top-10 most frequent words and their frequencies. In the text file, many words may appear in different forms, e.g. The, the, they should be considered same word. In addition, some words may have double quote, single quote, or period character either in prefix or suffix (e.g. book., “book”), your programs should be able to remove them and then consider the remaining characters as a word.

B. [**20 points**] (**Optional** for ECE 49500 students) Write Mapper and Reducer programs that read **purchases.txt** (**File source**: Udacity’s Hadoop course) file from HDFS cluster and finds the standard deviation of stores’ sales in each city. To calculate the standard deviation refer Welford's online algorithm. You may find Python code snippet in the link, adapt it for your reducer program.
The required items in the algorithm are as follows:

$$\overline{x_n} = \overline{x_{n-1}} + \frac{x_n - \overline{x_{n-1}}}{n}$$
$$M_{2,n} = M_{2,n-1} + (x_n - \overline{x_{n-1}})(x_n - \overline{x_n})$$
$$\sigma^2{}_n = \frac{M_{2,n}}{n}$$

Here, x_n denotes the sample mean of the first n samples and $\sigma^2{}_n$ their population variance. Once you calculate the popular variance for each city and then you can take the square root of it, which will give you the standard deviation.

C. **[15 points]** Write Mapper and Reducer programs that read a web server log file **http_log.txt** and find number of **unique files** request sent by each client. You may refer the link to understand the fields in the log file. The file request could be found inside the double quotes after `GET` in the line.

D. [**10 points**] Write Mapper and Reducer programs that read iris flower dataset **iris.txt** and find the average values for each attribute for each flower class. The description of dataset is available here.

Instructions

You can create a folder where you can keep your Map Reduce code. Use naming format **mapper_*lastname_letter*.py** for Mapper code and **reducer_*lastname_letter*.py** for reducer code for each problem. For example, mapper_niyaz_A.py and reducer_niyaz_A.py will be the files' names for Problem A. Once you finish mapper and reducer code, you can verify their correctness by using them for a small file similar to the problem data file. You can create it by examining the file content for the given problem or copy-pasting first 10-20 lines from the data file.

Mapper verification

```
cat problem_file_small.txt | python your_mapper_code_file
```

Check whether you are getting the same output that you expect. If you do not then you need to fix the mapper code and execute the command again. It is assumed that you are executing the command from the folder where you have put Map Reduce code and small data file.

Reducer code verification

```
cat problem_file_small.txt | python your_mapper_code_file | sort | python
your_reducer_code_file
```

Check whether you are getting the same output that you expect. If you do not then you need to fix the reducer code and execute the command again. It is assumed that you are executing the command from the folder where you have put Map Reduce code and small data file.

If your mapper and reducer work fine, then move the actual data file in the cluster and run Mapper and Reducer code on it. Since your code have been written in Python, you need to use Hadoop

streaming. You can find the jar file for it inside **/usr/local/hadoop/share/hadoop/tools/lib/** folder. To avoid the long typing, you can copy the jar file in /usr/local/hadoop folder.

```
cp /usr/local/hadoop/share/hadoop/tools/lib/hadoop-streaming-2.10.1.jar
/usr/local/hadoop/
```

Now execute commands similar to the example command given below:

```
hadoop jar /usr/local/hadoop/hadoop-streaming-2.9.1.jar
-mapper /path/to/your/MapReduceCode_folder/your_mapper_code_file
-reducer /path/to/your/MapReduceCode_folder/your_reducer_code_file \
-input /path/to/folder/for/your/data/file/in/cluster/ -output
/path/to/your/output/folder/in/cluster
```

The output file will be created in the output folder that you have provided in the command. The output file will have a prefix part. You can see the content of the file using `hdfs dfs -cat` command or you can download the file in the local file system using `hdfs dfs -copyToLocal` command.

Note: Do not use a name for the output folder if a folder with that name already exists in HDFS cluster. Either remove the existing folder or use other folder name.

Task 3 [20 points] (**Optional** for ECE 49500 students) Review the article titled as "*The history of Hadoop*" available here and summarize your key findings in 200-300 words.