CS440 - Assignment 7

Hadoop

February 28, 2018

The assignment is to be turned in before Midnight (by 11:59pm) on March 6th, 2018. You should turn in the solutions to this assignment as a ZIP file through the TEACH website.

1 Hadoop (8 points)

Objective

In this assignment you are going to implement a Hadoop program that computes the page visit counts of Wikipedia pages.

Setting Up Hadoop Environment

In this section, we are going to setup the Hadoop environment. First you need to login[1] to the Hadoop server using your engineering account:

```
ssh hadoop-master.engr.oregonstate.edu
```

Next you need to set the JAVA_HOME and HADOOP_HOME variables. These variables help Linux shell to locate the java and hadoop binaries when you are using hadoop commands. To set these variables, you can follow either of following methods[2]:

Method 1

Once you are logged in to the Hadoop server, open the shell startup configuration file *∼/.cshrc* with an editor and add the following lines to the end of the file:

```
setenv JAVA_HOME "/usr/lib/jvm/java-1.8.0"
setenv PATH "$JAVA_HOME/bin:$PATH"
setenv CLASSPATH ".:$JAVA_HOME/jre/lib:$JAVA_HOME/lib:$JAVA_HOME/lib/tools.jar"

setenv HADOOP_HOME /opt/hadoop/hadoop
setenv HADOOP_COMMON_HOME $HADOOP_HOME
setenv HADOOP_HDFS_HOME $HADOOP_HOME
setenv HADOOP_MAPRED_HOME $HADOOP_HOME
setenv HADOOP_YARN_HOME $HADOOP_HOME
setenv HADOOP_OPTS "-Djava.library.path=$HADOOP_HOME/lib/native"
setenv HADOOP_COMMON_LIB_NATIVE_DIR $HADOOP_HOME/lib/native
setenv PATH $HADOOP_HOME/sbin:$HADOOP_HOME/bin:$PATH
setenv HADOOP_CLASSPATH "${JAVA_HOME}/lib/tools.jar"
```

[1] To access the Hadoop server you need to be on campus or use the school's vpn.

[2] Method 2 is provided as an alternative if you could not have success to set variables manually following the first method.

After this, you need to reload the startup file:
`source ~/.cshrc`

Method 2

Once you are logged in to the Hadoop server via `ssh` command given in the first paragraph, run the following command which sets all the required variables from `hadoop_bashrc`[3]:

```
source /scratch/hadoop_bashrc
```

Working with HDFS

You have a folder on HDFS server at `/user/cs540/<your onid user name>`. Note that files on HDFS can be manipulated only using the special commands that are given below. Throughout the assignment, you should just use this directory when you need to work with HDFS. You can upload files or write output of your jobs to your directory. Note that this directory is not being backed up. Following is a list of commands that you can use to interact with HDFS:

- View list of files and folders: `hdfs dfs -ls <path>`
- Upload a file to HDFS:
  `hdfs dfs -put <file on engr account> <path to your directory on HDFS>`
- Download a file from HDFS:
  `hdfs dfs -get <path to your directory on HDFS>/<file_name>`
- View file on HDFS: `hdfs dfs -cat <path to your directory on HDFS>/<file_name>`
- Make a new directory: `hdfs dfs -mkdir <path your directory on HDFS>/<folder_name>`
- Remove a file: `hdfs dfs -rm <path to your directory on HDFS>/<file_name>`
- Remove a directory: `hdfs dfs -rm -r <path to your directory on HDFS>/<folder_name>`

Problem Explanation

Your task is to compute the page visit counts of Wikipedia pages over a period of time. We have provided an input file *input.dat* that contains page titles and visit counts for each page. Each row of the file has page title, page visit count, content size and validity(T/F) information. These values are separated by space. There can be multiple rows for a single page. Following is a fragment of sample input file:

```
Ace_of_Swords 2 17276 T
Law_school 29 539143 T
Ace_of_Swords 3 17705 T
```

You should write a program that computes the total page visits for each page. You should only consider valid rows i.e. whose validity is T. For example, the total page visits for *Ace_of_Swords* will be 5. The output file should contain page title and the total count separated by a single space. Following is the output for the sample input:

```
Law_school 29
Ace_of_Swords 5
```

Note that the rows of the output file do not need to be in the same order as the input file.

[3]These changes are limited to your current terminal session, so if you close your terminal you should rerun this command to have all the required variables set.

Implementation

You are given a skeleton code *PageCount.java* and you need to complete the *map* and *reduce* functions in this file.

Mapper

The *map* function is called once for each key/value pair in the input split. The *value* argument of *map* function is of type *Text* and contains the text of the input file. You can call *toString*() on it to get a String object of the value [4]. You should use *value* to produce the needed key-value pairs. After that you can call $context.write(new\_key, new\_value)$ to send the key values to the reducer.

Reducer

The *reduce* function is called once for each key received from the Mapper. You can iterate over the values of that key to do your computations. In the end of the method you can call $context.write(another\_key, another\_value)$ to write the results of reduce to an output file.

Compiling and Running

Once you have finished editing the Skeleton, follow these commands to run your hadoop job:

a. To compile your code and create a jar file:

```
hadoop com.sun.tools.javac.Main PageCount.java
jar cf pc.jar PageCount*.class
```

b. Upload the input file *input.dat* to your HDFS folder:

```
hdfs dfs -put input.dat <path to your folder>/
```

c. Run the application:

```
hadoop jar pc.jar PageCount <path to your folder>/input <path to your folder>/output
```

Note that the output directory should not exist before running the above command. Otherwise you will get an error.

d. You can view the output file:

```
hdfs dfs -cat <path to your folder>/output/part-r-00000
```

Depending on the number of reducers you may get more than one output files. Use the *list* command mentioned in the previous section to go through different output files.

Hints:

- You can put your code in a directory under your engineering account's home folder. This way, after logging in to the Hadoop server, you can access your code in your engineering home folder and you can edit and compile the code there. However, the input file should be uploaded to the HDFS using the given commands.

[4] For more information on the *Text* class visit this page.

- After running your job you can view its status using a web interface. To do this, login to Hadoop server. If you are a mac user, you need to add $-X$ to ssh command to enable $X11$ services. After you logged in, type $firefox$ in the terminal. This should open a firefox window. Then go to "http://hadoop-master.engr.oregonstate.edu:8088/" in the firefox. You should be able to see information on your hadoop jobs.

Implementation notes for C++ developers

You are provided with an skeleton code containing *mapper* and *reducer* cpp and header files. You need to complete this code and run it on hadoop server. Once you have completed the codes, compile it on hadoop server. Make sure you enable execution permission for your executables using *chmod* command. Then use the following the instructions:

a. Upload the input file *input.dat* to your HDFS folder:

```
hdfs dfs -put input.dat <path to your folder>/
```

b. Run the application:

```
hadoop jar /opt/hadoop/hadoop-2.7.3/share/hadoop/tools/lib/hadoop-streaming-2.7.3.jar
 -input <your folder on hdfs>/<input file name>
 -output <your folder on hdfs>/<outputdirectory>
 -file ./mapper -mapper ./mapper -file ./reducer -reducer ./reducer
```

Note that the output directory should not exist before running the above command. Otherwise you will get an error.

c. You can view the output file:

```
hdfs dfs -cat <path to your folder>/output/part-r-00000
```

Depending on the number of reducers you may get more than one output files. Use the *list* command mentioned in the previous section to go through different output files.

Deliverables

You should submit a zip file containing completed PageCount.java (or C++ files), your output file and a pdf file containing answer to the Recovery question.