Special practice: Writing a Hadoop Streaming Program

Files and Directories Used in this Exercise

Folder: `~/Desktop/hadoop-streaming`

Source code: `mapper.py`

`reducer.py`

Test data (HDFS): `shakespeare`

In this exercise you will repeat the same task as a previous exercise: writing a program to calculate average word lengths for letters. However, you will write this as a streaming program using a scripting language of your choice rather than using Java.

Your virtual machine has Perl, Python, PHP, and Ruby installed, so you can choose any of these—or even shell scripting—to develop a Streaming solution.

For your Hadoop Streaming program you will not use Eclipse. Launch a text editor to write your Mapper script and your Reducer script. Here are some notes about solving the problem in Hadoop Streaming:

1. **The Mapper Script**

   The Mapper will receive lines of text on `stdin`. Find the words in the lines to produce the intermediate output, and emit intermediate (key, value) pairs by writing strings of the form:

   key <*tab*> *value* <*newline*>

   These strings should be written to `stdout`.

2. **The Reducer Script**

   For the reducer, multiple values with the same key are sent to your script on stdin as successive lines of input. Each line contains a key, a tab, a value, and a newline. All lines with the same key are sent one after another, possibly followed by lines with a different key, until the reducing input is complete.

For example, the reduce script may receive the following:

t *3*
t *4*
w *4*
w *6*

For this input, emit the following to stdout:

t *3.5*
w *5.0*

Observe that the reducer receives a key with each input line, and must "notice" when the key changes on a subsequent line (or when the input is finished) to know when the values for a given key have been exhausted. This is different than the Java version you worked on in the previous exercise

3. **Change into the directory containing your code and run the streaming program:**

```
$ hadoop jar /usr/lib/hadoop-0.20-mapreduce/\
contrib/streaming/hadoop-streaming-<your installed version>.jar \
-files <pathToMapScript>,<pathToReduceScript> \
-input <dataInHDFS> -output <outputDirInHDFS> \
-mapper <mapBasename> -reducer <reduceBasename>
```

Caveats:

Be careful, the first line ends with "/\" There is NO SPACE.

Make sure the -files option is the first "option". Why? Because it is a "*generic*" option. When running a hadoop program, the generic options are parsed before the program specific options.

Finally, remember, you may need to delete any previous output before running your program with `hadoop fs -rm -r oldOutputDir`.)

4. **Review the output in the HDFS directory you specified (outputDirInHDFS).**

Solution in Python

You can find a working solution to this exercise written in Python in your VM under Desktop/streaming-example.

The files are:

- **reducer.py**
- **mapper.py**

To run the solution, change into directory the directory containing `reducer.py` and `mapper.py` and run this command:

```
hadoop jar /usr/lib/hadoop-0.20-mapreduce/\
contrib/streaming/hadoop-streaming-2.6.0-mr1-cdh5.8.0.jar \
-files mapper.py,reducer.py \
-input shakespeare/comedies/* -output avgwordstreaming \
-mapper mapper.py -reducer reducer.py
```

Remember the caveats:

Be careful, the first line ends with "/\" There is NO SPACE.

Make sure the -files option is the first "option".

Remember to delete any old output, Hadoop will refuse to run if the output directory already exists.

Running your own installation of Hadoop?

In general, you can find the current version of your streaming jar by typing:

```
$ ls /usr/lib/hadoop-0.20-mapreduce/contrib/streaming/hadoop-streaming-*
```

This is the end of the Exercise