Writing a MapReduce Streaming Program

**In this exercise you will repeat the same task as in the 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

|  |  |
| --- | --- |
| **Exercise Dir** | ~/workspace/averagewordlength |
| **Test Data** | shakespeare |

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.

1. 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:

For this input, emit the following to stdout:

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. Run the streaming program:

$ yarn jar /usr/hdp/2.3.6.0-3796/hadoop-mapreduce/ hadoop-streaming.jar -input shakespeare -output avgword -file python\_sample\_solution/mapper.py -file python\_sample\_solution/reducer.py -mapper mapper.py - reducer reducer.py

|  |  |
| --- | --- |
| t | 3 |
| t | 4 |
| w | 4 |
| w | 6 |

|  |  |
| --- | --- |
| t | 3.5 |
| w | 5.0 |

NOTE: You may need to delete any previous output before running your program with hdfs dfs -rm -r *dataToDelete*

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

**Solution in Python**

You can find a working solution to this exercise written in Python in the directory

~/workspace/averagewordlength/python\_sample\_solution

To run the solution, change directory to ~/workspace/averagewordlength and run this command:

Results should resemble the following output sample:

1 1.15094339623

2 1.07692307692

3 1.0

4 1.5

5 1.5

6 1.75

7 1.0

8 1.66666666667

9 1.0

a 3.27215105713

b 4.44419990449

c 6.2203493377

d 4.32558877171

e 5.32817352253

f 4.88152021426

g 5.16659172662

h 3.97247007948

…

**END**