## Monitoring Hadoop: Practical exercise

You will be working in pairs and using a preconfigured Linux VM (virtual machine) with Hadoop (v1.20), PIG and HIVE already installed.

Python and Java are available on the Linux box.

A number of text editors are available, including nano, vi and vim.

You will be working completely at the command line so you will need to access the VM via SSH using an SSH client application (such as PuTTY if you are using Windows).

The VMs are at:

shu-hadoopn.cloudapp.net

There are two preconfigured users: hduser and azureuser

To perform the Hadoop exercises, you will need to login as hduser, so:

ssh hduser@shu-hadoopn.cloudapp.net

(Don't forget to replace n with the number for your group).

Hadoop Web UIs are also available (network firewalls permitting):

http://shu-hadoopn.cloudapp.net:50070/ – web UI of the NameNode daemon http://shu-hadoopn.cloudapp.net:50030/ – web UI of the JobTracker daemon http://shu-hadoopn.cloudapp.net:50060/ – web UI of the TaskTracker daemon

### Part 1: Getting ready

1. Some sample data has been made available to you in the gutenberg directory (this is located at /home/hduser/gutenberg. It has already been loaded into the VM's HDFS.
2. Write a simple Java mapper and reducer to do a wordcount on the contents of these three files.
3. Use the Linux time command to time how long the wordcount takes. See the guidance notes below for an example of how to do this.
4. Repeat exercise 2 but write the mapper and reducer in Python and use Hadoop streaming to run the wordcount.
5. Again, use the time command to time how long this takes.

#### Guidance Notes

* You will find some sample Python scripts to use in the python-streaming directory. This is located at /home/hduser/python-streaming
* If you are unfamiliar with how to run Hadoop jobs from the command line, follow Michael Noll's tutorials:
* Running a MapReduce job:   
  http://www.michael-noll.com/tutorials/running-hadoop-on-ubuntu-linux-single-node-cluster/#running-a-mapreduce-job
* Running a Hadoop streaming Python job:   
  http://www.michael-noll.com/tutorials/writing-an-hadoop-mapreduce-program-in-python/#running-the-python-code-on-hadoop
* The Linux time command: http://unix.stackexchange.com/questions/10745/how-do-i-time-a-specific-command
* Basic usage is: time <command> and you will get something back like (it's the real time you are interested in:
* real 0m0.178s  
  user 0m0.003s  
  sys 0m0.005s

### Part 2: Comparing Native Java, Python streaming, PIG and HIVE

For these exercises, you have been given some data about airline departures from airports in the USA for 2008. The data is in the airline\_data directory (/home/hduser/airline\_data ) and is described in the README which can be read using a text editor such as nano.

1. Put the data into a suitable directory in the HDFS. Instructions on how to do this are in the Michael Noll tutorial mentioned in the previous section.
2. Write a native Java mapper and reducer to calculate the average delay time across all flights.
3. Run this job and time how long it takes using the Linux time command.
4. Write and execute a Python streaming mapper and reducer to calculate the average delay. Time how long it takes.
5. Write a PIG script to calculate the average delay. Use your preferred method to run and time the script processing time (you can do this either interactively or by running the script).
6. Write and execute a HIVE query to calculate the average delay. Use your preferred method to run and time the script processing time.

#### Guidance Notes

Some Web resources that might help:

**Java MapReduce calculate averages:**   
Example code to demonstrate an approach: https://gist.github.com/rishav-rohit/8133433

**Python MapReduce calculate averages:**   
This one is for you to work out!

**PIG calculate averages:** http://stackoverflow.com/questions/15212985/calculate-average-using-pig

**HIVE calculate averages:**http://stackoverflow.com/questions/23804281/hive-grouping-and-calculating-average-by-calculating-distinct

### Part 3: Finishing off

Some points to consider to feedback at the end of the session:

1. What factors can be changed to improve the execution time of a job?
2. How can you determine which factors to change?
3. What is meant by total time to solution? How can this concept help determine the most appropriate solution?