Recitation Two: Hadoop Examples
Cloud Computing 2017

Answers to R2 should be emailed to the course email (ece.cc579@gmail.com) before 7pm on Friday Sep 29th. Your full names should be in the email. The recitation should be done in groups of two.

R1: Hadoop-Wordcount Example in Java and Python:

Download the files in the below link:

https://drive.google.com/open?id=0B1an055wg0hqM3VVWjVGdzBxTjg

a) Compile and execute the wordcount code in the wordcount/java folder using the Makefile. Show the the output to the instructor.

b) Go to the wordcount/streaming directory. Write the wordcount map and reduce codes in python (may.py and reduce.py). Also edit the Makefile in the same directory to compile your code. Show the output and code to the instructor.

R2: Histogram of Data in Python:

In this exercise we will use Hadoop to build a histogram of input data. Histogram is a graphical representation of frequencies of samples. The provided example reads an input and finds the min, max, and mean of the input first. It then uses the below equations to tag each input with a bin in the histogram.

$$binWidth = (x_{max} - x_{min})/numberofbins$$

$$binNumber = (x_i - x_{min})/binWidth$$

$$binCenter = binNumber * binWidth + x_{min} + binWidth/2$$

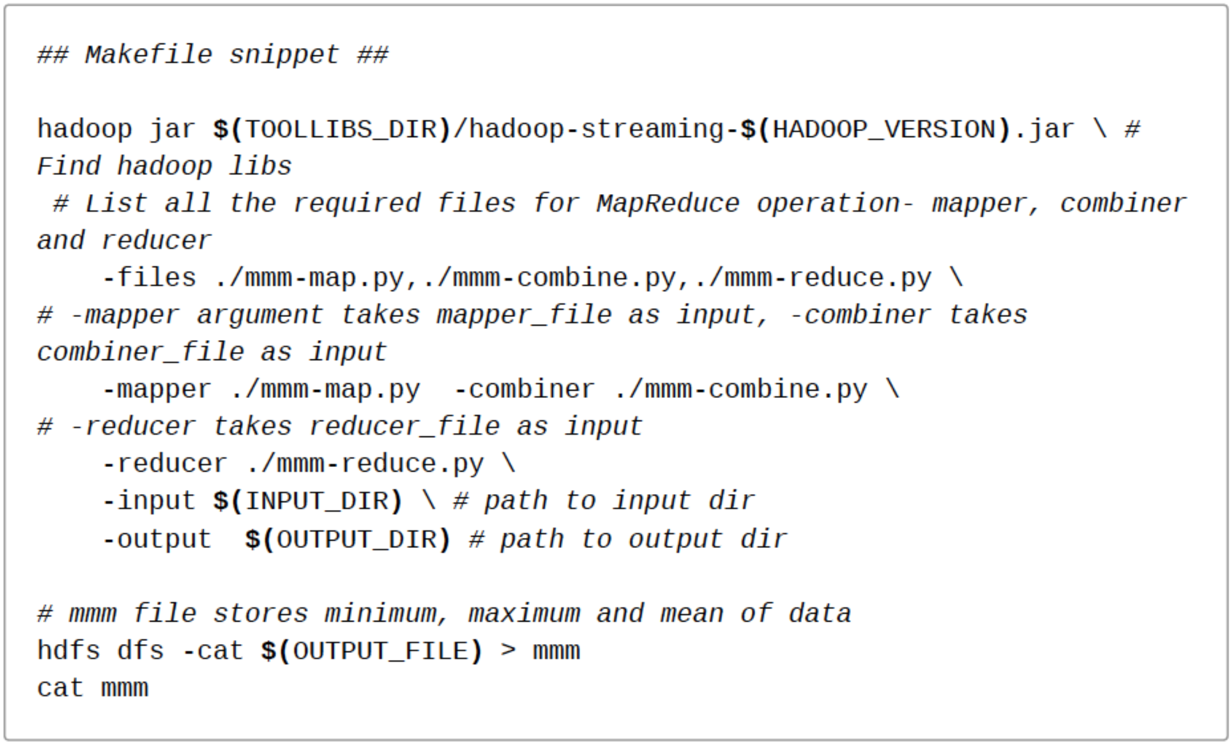
Download the code from:
https://github.com/ishank26/ECE579.git

or clone using -

```
local$ git  clone https://github.com/ishank26/ECE579.git
```

To update your local clone with latest changes in repo-
local$ git pull https://github.com/ishank26/ECE579.git

a) Read the provided code. What does the below part of the makefile do? What is the output mmm? What does mmm-map do? What does the for-loop in mmm-combiner do? Can the mmm-combiner code be replaced with the code in mmm-reducer? Why?



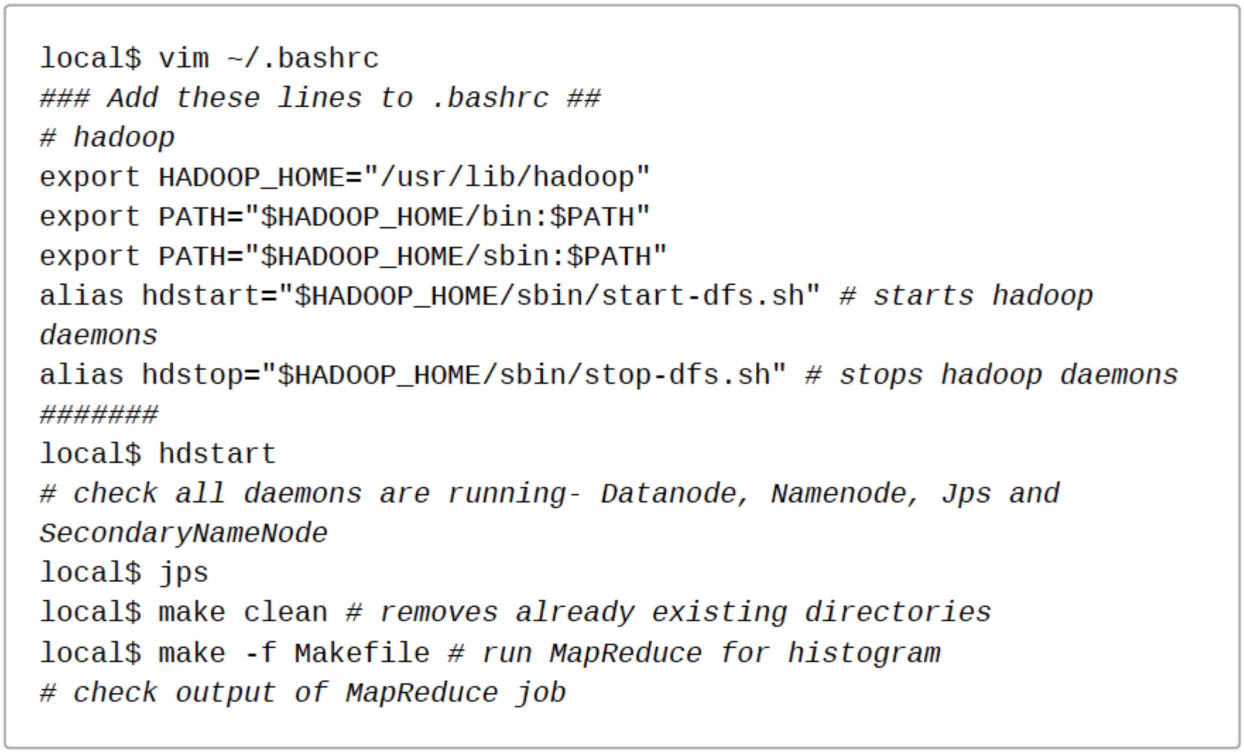
```
## Makefile snippet ##

hadoop jar $(TOOLLIBS_DIR)/hadoop-streaming-$(HADOOP_VERSION).jar \ #
Find hadoop libs
 # List all the required files for MapReduce operation- mapper, combiner
and reducer
    -files ./mmm-map.py,./mmm-combine.py,./mmm-reduce.py \
# -mapper argument takes mapper_file as input, -combiner takes
combiner_file as input
    -mapper ./mmm-map.py  -combiner ./mmm-combine.py \
# -reducer takes reducer_file as input
    -reducer ./mmm-reduce.py \
    -input $(INPUT_DIR) \ # path to input dir
    -output  $(OUTPUT_DIR) # path to output dir

# mmm file stores minimum, maximum and mean of data
hdfs dfs -cat $(OUTPUT_FILE) > mmm
cat mmm
```

b) Now that we have found the mean, min, and max of data we want to build a histogram of the original inputs. Complete the hist-map and hist-combine-reduce files to show the number of total data points in each bincenter.

Hints for running the code: in your console follow the below:



```
local$ vim ~/.bashrc
### Add these lines to .bashrc ##
# hadoop
export HADOOP_HOME="/usr/lib/hadoop"
export PATH="$HADOOP_HOME/bin:$PATH"
export PATH="$HADOOP_HOME/sbin:$PATH"
alias hdstart="$HADOOP_HOME/sbin/start-dfs.sh" # starts hadoop
daemons
alias hdstop="$HADOOP_HOME/sbin/stop-dfs.sh" # stops hadoop daemons
#######
local$ hdstart
# check all daemons are running- Datanode, Namenode, Jps and
SecondaryNameNode
local$ jps
local$ make clean # removes already existing directories
local$ make -f Makefile # run MapReduce for histogram
# check output of MapReduce job
```