Lab 2: Matrix Multiply with Hadoop-Streaming

DS-GA 1004 (Big Data)
February 5, 2018

In this lab, you will implement a matrix multiplication routine using the Hadoop MapReduce framework and test the scalability of your code for various numbers of parallel reducers.

1. Log in (via ssh) to Dumbo (see Lab 1 for instructions)

2. Copy the Lab 2 files from HDFS to your home directory on Dumbo. To do this, type

```
hfs -get /user/ecc290/lab2
```

3. Type cd lab2 to move into the folder you just copied and type ls to see the files.
You should see:

- src: a directory with the following source files:
    - map.py: the Python code skeleton for the map function
    - map.sh: a Bash script to load the necessary libraries for the map.py code
    - reduce.py: the Python code skeleton for the reduce function
    - reduce.sh: a Bash script to load the necessary libraries for the map.py code
- matsmall.txt: an input file with small input matrices for testing your code
- example: a directory with example map and reduce Python codes for the wordcount example from Lab 1. These may be helpful for you to examine as you flesh out the map.py and reduce.py skeletons.

Together: Let’s look at the files in the example folder and go over the structure of the map and reduce functions

1. Introduction to matrix multiplication (see Lab 2 slides)

2. Fill in the skeleton map.py and reduce.py functions in the lab2 folder to implement the matrix multiply (you may do the naive matrix multiply discussed in the slides, or try a blocked version if you are feeling ambitious).

    Together: Let’s look at the format of the input file matsmall.txt and the skeleton map.py and reduce.py files

    You should test your map.py and reduce.py files locally on the small matrix file matsmall.txt before running a Hadoop job.

First make sure you have python 3.4.4 loaded. Type

```
module load python/gnu/3.4.4
```

You can then use the following command to run your code (from the lab2 folder):

```
cat matsmall.txt | python src/map.py 2 3 | sort -n | python src/reduce.py 5
```

This should give the output entries (not necessarily in this order):

```
(0, 0), 90.000000
(0, 1), 100.000000
(0, 2), 110.000000
(1, 0), 240.000000
(1, 1), 275.000000
(1, 2), 310.000000
```

3. Once you have debugged your code for the small example matrix, we will now run Hadoop jobs for a larger test matrix. From the lab2 folder, type (one command):

```
hjs \
  -files /home/netid/lab2/src \
  -mapper src/map.sh \
  -reducer src/reduce.sh \
  -input /user/ecc290/matbig.txt \
  -output /user/netid/mm.out
```

Note: Bash concatenates each line ending with a backslash (“\”) with the next line, provided that the backslash is the last character on the line (e.g., no trailing spaces)

Once the job is complete, type

```
hfs -getmerge mm.out mm.out
```

to move the file to your local dumbo directory. (You can view the output using, e.g., `vi mm.out`)

(!) Note that the input arguments for map and reduce (which give the dimensions of the matrix multiplication problem) are hard-coded for the matbig.txt problem in the map.sh and reduce.sh files. If you want to run the matsmall.txt example using Hadoop, you will need to edit the src/map.sh and src/reduce.sh files.

(!) Remember to type `hfs -rm -r mm.out` in between runs - the Hadoop job will fail if this directory already exists on HDFS.

4. We will now record the time it takes to run the MapReduce job using various numbers of reducers. This can be done by setting command line arguments when you submit the Hadoop job. For example, to run with 4 reducers (all one line):

```
hjs \
  -D mapreduce.job.reduces=2 \
  -files /home/netid/lab2/src \
  -mapper src/map.sh \
  -reducer src/reduce.sh \
  -input /user/ecc290/matbig.txt \
  -output /user/netid/mm.out
```

Gather timing data for the reduce phase using these numbers of reducers: 1,2,4,8,16,32.

Using your favorite plotting software, create a plot of this timing data. Plot number of reducers on the x-axis and time in seconds on the y-axis. (You may use a log scale on the x-axis if desired). Save your plot as an image file (e.g., bmp, png, jpg, pdf, eps) with the name ‘reducescaling’. You can make the plot on your local computer (vs. Dumbo).

Deliverable: (due Monday, 2/12/18 at 6pm) Submit three files to NYU Classes: your map.py file, reduce.py file, and the image file for your plot (e.g., reducescaling.jpg)