Programming Assignment 1
A Simple Map-Reduce Program

Due on Tuesday February 19 before midnight

Description

The purpose of this project is to develop a simple Map-Reduce program on Hadoop for graph processing.

This project must be done individually. No copying is permitted. **Note: We will use a system for detecting software plagiarism, called Moss, which is an automatic system for determining the similarity of programs.** That is, your program will be compared with the programs of the other students in class as well as with the programs submitted in previous years. This program will find similarities even if you rename variables, move code, change code structure, etc.

Note that, if you use a Search Engine to find similar programs on the web, we will find these programs too. So don't do it because you will get caught and you will get an F in the course (this is cheating). Don't look for code to use for your project on the web or from other students (current or past). Just do your project alone using the help given in this project description and from your instructor and GTA only.

Platform

You will develop your program on SDSC Comet. Optionally, you may use IntelliJ IDEA or Eclipse to help you develop your program, but you should test your programs on Comet before you submit them.

Setting up your Project

Follow the directions on How to login to Comet at comet.html. Please email the GTA if you need further help. After you login on Comet:

Edit the file `.bashrc` (note: it starts with a dot) using a text editor, such as `nano .bashrc`, and add the following 2 lines at the end (cut-and-paste):

```
alias run='srun --pty -A uot143 --partition=shared --nodes=1 --ntasks-per-node=1 --mem=5G -t 00:05:00 --wait=0 --export=ALL'
export project=/oasis/projects/nsf/uot143/fegaras
```

logout and login again to apply the changes. On Comet, download and untar project1:

```
wget http://lambda.uta.edu/cse6331/project1.tgz
tar xfz project1.tgz
chmod -R g-wrx,o-wrx project1
```

Go to `project1/examples` and look at the Map-Reduce example `src/main/java/Simple.java`. You can compile the Java file using:

```
run simple.build
```

and you can run it in standalone mode using:

```
sbatch simple.local.run
```

The file `simple.local.out` will contain the trace log of the Map-Reduce evaluation while the file `output-simple/part-r-00000` will contain the results. Optionally, you can run Simple.java in distributed mode using:

```
sbatch simple.distr.run
```

Please note that running in distributed mode will waste at least 10 of your SUs.

Project Description

In this project, you are asked to implement a simple graph algorithm that needs two Map-Reduce jobs. A directed graph is represented as a text file where each line represents a graph edge. For example,

```
20,40
```

represents the directed edge from node 20 to node 40. First, for each graph node, you compute the number of node neighbors. Then, you group the nodes by their number of neighbors and for each group you count how many nodes belong to this group. That is, the result will have lines such as:

```
10 30
```

which says that there are 30 nodes that have 10 neighbors. To help you, I am giving you the pseudo code. The first Map-Reduce is:

```
map ( key, line ):
  read 2 long integers from the line into the variables key2 and value2
  emit (key2,value2)

reduce ( key, nodes ):
  count = 0
  for n in nodes
      count++
  emit(key,count)
```

The second Map-Reduce is:

```
map ( node, count ):
  emit(count,1)

reduce ( key, values ):
  sum = 0
  for v in values
      sum += v
  emit(key,sum)
```

You should write the two Map-Reduce job in the Java file `src/main/java/Graph.java`. An empty `src/main/java/Graph.java` has been provided, as well as scripts to build and run this code on Comet. **You should modify the Graph.java only**. In your Java main program, args[0] is the graph file and args[1] is the output directory. The input file format for reading the input graph and the output format for the final result must be text formats, while the format for the intermediate results between the Map-Reduce jobs must be binary formats.

You can compile `Graph.java` on Comet using:

```
run graph.build
```

and you can run Graph.java in standalone mode over a small dataset using:

```
sbatch graph.local.run
```

The results generated by your program will be in the directory `output`. These results should be:

```
2 2
3 2
4 1
5 2
7 1
```

You should develop and run your programs in standalone mode until you get the correct result. After you make sure that your program runs correctly in standalone mode, you run it in distributed mode using:

```
sbatch graph.distr.run
```

This will process the graph on the large dataset `large-graph.txt` and will write the result in the directory `output-distr`. These results should be similar to the results in the file `large-solution.txt`. Note that running in distributed mode will use up at least 10 of your SUs. So do this a couple of times only, after you make sure that your program works correctly in standalone mode.

Optional: Use an IDE to develop your project

If you have a prior good experience with an IDE (IntelliJ IDEA or Eclipse), you may want to develop your program using an IDE and then test it and run it on Comet. Using an IDE is optional; you shouldn't do this if you haven't used an IDE before.

On IntelliJ IDEA, go to New→Project from Existing Sources, then choose your project1 directory, select Maven, and then Finish. To compile the project, go to Run→Edit Configurations, use + to Add New Configuration, select Maven, give it a name, use Working directory: your project1 directory, Command line: install, then Apply.

On Eclipse, you first need to install m2e (Maven on Eclipse), if it's not already installed. Then go to Open File...→Import Project from File System, then choose your project1 directory. To compile your project, right click on the project name at the Package Explorer, select Run As, and then Maven install.

Optional: Use your laptop to develop your project

If you'd prefer, you may use your laptop to develop your program and then test it and run it on Comet. If you have Mac OS or Linux, make sure you have Java and Maven installed. If you have Windows 10, you may install Ubuntu Shell and do: `sudo apt install openjdk-8-jdk-headless maven`.

To install Hadoop and project:

```
cd
wget http://apache.claz.org/hadoop/common/hadoop-2.6.5/hadoop-2.6.5.tar.gz
tar xfz hadoop-2.6.5.tar.gz
wget http://lambda.uta.edu/cse6331/project1.tgz
tar xfz project1.tgz
```

To compile and run project1:

```
cd project1
mvn install
rm -rf output
export JAVA_HOME=/usr
~/hadoop-2.6.5/bin/hadoop jar target/*.jar Graph small-graph.txt output
```

JAVA_HOME must point to your java installation.

Documentation

- The The Map-Reduce API. The API has two variations for most classes: `org.apache.hadoop.mapreduce` and `org.apache.hadoop.mapred`. **You should only use the classes in the package `org.apache.hadoop.mapreduce`**
- The org.apache.hadoop.mapreduce package
- The Job class

What to Submit

You need to submit the following files only:

```
project1/src/main/java/Graph.java
project1/graph.local.out
project1/output-distr/part-r-00000
project1/graph.distr.out
```

Do not submit any other files. Just submit each of these 4 files one-by-one using the following form. These files are automatically uploaded directly into your personal class account for this particular project, so you don't have to include your name or student ID or project number in the file name. You may submit your files as many times as you like, but only the most recently submitted files will be retained and evaluated (newly submitted files replace the old files under the same file name). After you submit the files, please double-check that your submitted files are correct by clicking on the Status link. If you cannot login or have a problem submitting the project using this form, ask the GTA for help.

Submit Programming Assignment #1:

Select a file: Choose File No file chosen

Send File

Last modified: 02/07/2019 by Leonidas Fegaras