CMPSC 497B/CSE 597B HW3. DUE 10/27

Instructions: Debug/develop on the VM. Afterwards use AWS

You cannot share code with other students or view code written by other students. Do not post code online. Violation = Academic integrity violation.

Your submission will be a zip file containing:

- Your code. The controller logic should be in the file BFS.java
- A report containing your pseudocode.
- Your output from AWS

1. Version for CSE 597B and CMPSC 497B Honors Option

Problem 1. *The goal is to implement a variation of the parallel breadth-first search (BFS) algorithm. The command line arguments are the input directory, output directory and a list of nodes (there is no fixed limit on how many nodes can be specified). The last node is the destination node, all of the other nodes are source nodes, and the return value is the length of the shortest (directed) path between any source node and the destination. For example:*

```
hadoop jar bfs.jar org.psuid.BFS inputdir outputdir 33 41 83 90
```

will treat 33, 41, and 83 as the source nodes and 90 as the destination node. The output will be the minimum of:

- *the length of the shortest path from 33 to 90*
- *the length of the shortest path from 41 to 90*
- *the length of the shortest path from 83 to 90*

Note that this is equivalent to first running parellel BFS with source 33 to get its distance to 90, then running parallel BFS from 41 to 90, then running parallel BFS from 83 to 90 then taking the minimum. That approach is too slow and will get 0 points. Instead, it is possible to do this in roughly the same time as a single parallel BFS algorithm. Hint: make sure you completely understand Dijkstras algorithm and parellel BFS.

- *Debug using smallgraph.txt on the VM. In particular, make sure it works for sources 0,7 and destination 8 as well as source 8 and destination 7.*
- *Run it on AWS using the LiveJournal dataset which has the same format as smallgraph.txt (beware of comment lines in the data). Your submitted output data file should use the following sources: 597742, 99603, 102427, 97903, and destination 113251. Hardcoding the node values will result in 0 points.*

2. Version for CSE 497B

Problem 2. *The goal is to implement the parallel breadth-first search algorithm and then to select certain nodes of interest. The command line arguments are the input directory, output directory and a list of nodes (there is no fixed limit on how many nodes can be specified). The first node is the source node, all of the other nodes are the destination nodes, and the return values are: the destination nodes and their distance from the source. For example:*

```
hadoop jar bfs.jar org.psuid.BFS inputdir outputdir 33 41 83 90
```

will use the source node as 33 and will produce a file: 41 distance-from-33 83 distance-from-33 90 distance-from-33

- *Debug using smallgraph.txt on the VM. In particular, make sure it works for sources 0 and destinations 7, 8 as well as source 8 and destination 7.*
- *Run it on AWS using the LiveJournal dataset which has the same format as smallgraph.txt (beware of comment lines in the data). Your submitted output data file should use the following source: 597742 and destinations: 99603, 102427, 97903, 113251. Hardcoding the node values will result in 0 points.*