CSC 221 Spring 2020 Due 11:00pm 4/29/2020 Project 4

Graph Exercises

Overview

In this project you will be computing some basic properties of a graph. The graph you will be working with is constructed from links embedded in web pages from a small university that we're going to call "hogsnorts.edu", even though it's data from an actual university about the size of Wake Forest – but it's not from our campus. The basic understanding is simply that a hyperlink on a web page can be represented as an edge in a directed graph and that abstraction is very useful for certain types of calculations.

Instructions

- Go to https://classroom.github.com/a/zC7u4b7b and accept the assignment to get a URL for your private repository. This URL will be something like https://github.com/WFU-CSC221/project4-username. Copy the URL. You'll need it in the next step.
- Close your IntelliJ / IDE windows. Create a new IntelliJ project by selecting New Project and "Get from Version Control." On the next screen choose "Repository URL' and NOT "GitHub". Paste in the URL obtained in the previous step and Clone. Accept all the defaults.
- You should be able to compile and run the code provided without modifications at this point.

Files

- The data files are named **URLs.dat, URLsTest.dat, links.dat, and linksTest.dat**. The files with "Test" in the names are recommended for testing your code on small graphs that you can easily draw on paper. **URLs.dat and links.dat** are the real data files.
- ***URLs*** and ***URLsTest*** have the same format. Each line contains a vertex identifier and a label for that vertex. Note that in the code provided, a constant of 1 has been subtracted from each identifier so that the graph vertices are numbered starting at 0.
- **links** and **linksTest** represent the hyperlinks associated with the vertices listed in the corresponding URL file. The first line contains the number of Vertices (V) and Edges (E). Each subsequent line contains 2 integers representing a hyperlink from the first web URL (by number) **to** the second web page (by number). Again, in the code provided a constant 1 has been subtracted from each number because array indices start at 0.
- The code provided reads in two data files and constructs a Graph object. Note that the AdjacencyList for the graph as well as the list of URLs are "public" in the Graph class as a convenience.

Your Tasks

- Write code to answer the following questions. Your responses can be submitted by adding a text file name "Results" to your Project and then committing it and pushing it when your work has been completed.
    - What is the maximum out-degree of any vertex and what is its URL name

(assume the answer is unique)?

- What is the maximum in-degree of any vertex and what is its URL name (assume the answer is unique)?
- Is the graph strongly connected? How did you determine your response?
- Is the graph weakly connected? That is, is the underlying undirected graph connected? How did you determine your response?
- What is the "diameter" of the underlying undirected graph? The diameter of a graph is the largest of all the shortest paths between any two vertices. Use Dijkstra's method incorporating a Priority Queue (min-heap) to calculate the shortest path from each vertex to every other vertex. Identify the largest of these shortest path values. [All edges in the undirected graph have a cost of 1.] Use data structures in the Standard Library when possible. Don't reinvent the wheel.

Submission and Evaluation

- Submit your code through GitHub classroom. The deadline is 11pm on 4/29/2020.
- Be sure to identify yourself in the Honor pledge. It is sometimes difficult to identify you from your GitHub user name. If you "borrow" code from a web site, include an appropriate attribution.
- Be sure to include your "Results" file.
- Your program will be evaluated not only on correctness but also on general structure and readability. Use meaningful variable names and method names. Define your own classes if it helps to clarify the code. Don't comment every line of your program but use comments to explain your logic.