CSE 6140 / CX 4140 Assignment 1
Help Protect the City

Due Date: September 13, 2018 at 6pm Eastern time

1 Background

Let $G = (V, E)$ be a simple, unweighted, undirected graph. The distance between two vertices $u, v \in V$, given by $d(u, v)$, is defined as the length of the shortest path between u and v in the graph G. Note that $d(u, u) = 0$.

The *farness* of a vertex u in a graph G is defined as:

$$\text{far}(u) = \sum_{v \in V} d(u, v)$$

The *closeness centrality* of u is then defined as:

$$C_C(u) = \frac{1}{\text{far}(u)}$$

2 Problem

This year during Dragon Con, Batgirl made a special visit to Klaus to visit your professor. She has an urgent problem and needs theoretical and computational help before September 13. She received a tip-off that the Riddler is going to perform a major heist in Gotham City at some point during the following week—Batgirl is planning on preventing it and saving the city.

More specifically, Batgirl does not know when or where exactly the heist will occur, so she wants to find a well located position in the city. She plans on remaining at that position and since it is well located, at the first sign of the heist she can quickly move to the correct location and prevent it. There are a number of points of interest that she has identified that the heist may occur at. Closeness centrality seems like an important tool as she is looking for a ranking of locations that are the "least far" from every point a heist may occur at.

Your TAs have been working hard helping Batman connect his cell phone audio player to bluetooth on his batmobile, so we are turning to you for help with this problem.

You should approach the problem in the following way. You will be given a connected graph G (for Batgirl's problem, this will consist of the intersections as vertices and roads as edges throughout

Gotham City) and a set of vertices H (the intersections the heist may occur at). Let the *heist-closeness centrality* of u be given by the following:

$$C_{HC}(H, u) = \frac{1}{\sum_{v \in H} d(u, v)} \tag{1}$$

Your job is to design, analyze, and implement an algorithm to efficiently compute the heist-closeness centrality for each vertex in a graph.

Concretely, you need to do the following:

1. Design an algorithm to compute the heist-closeness centrality, as defined by (1). Write the high-level idea, then concrete details and pseudo-code for the algorithm.
2. Prove the feasibility of the algorithm. That is, prove that the algorithm will correctly compute the heist-closeness centrality for each vertex.
3. Compute and prove the run-time and space complexity of the algorithm. Identify if different data structures will impact the complexity.
4. Using one of the provided templates, implement your new algorithm and ensure it runs correctly on the sample graphs provided.
5. Describe your implementation in writing and perform an evaluation that includes a plot containing the run-time as the number of edges increase. Describe at a high level how different input graph topologies may impact your run-time.

3 I/O File Formats

There are three file formats used for this assignment which are described in this section.

3.1 Graph File Format

The input graph files are in a format that is easy to construct a CSR from. The file extension used is `.graph`. The format is as follows:

```
n m
s1 d1
s2 d2
...
sm dm
```

Here, `n` is $n = |V|$ as expected. Due to the design of CSR construction in the templates, `m` is actually $2m = 2|E|$, with each edge essentially considered as two directed edges with each direction included in the input file. The values should be formatted as ASCII integers.

The edges must be sorted numerically by source.

3.2 Heist Nodes File Format

The file extension `.h` is used for heist nodes. The heist nodes are provided in the following format:

```
k
v1
v2
...
vk
```

Here, `k` is the number of heist nodes, or $k = |H|$. Each following `vi` is a given vertex ID that is a heist node. The values should be formatted as ASCII integers.

3.3 Output File Format

The output file format has an extension of `.hc` and is as follows:

```
n
v1-hcc
v2-hcc
...
vn-hcc
```

Here, `n` is $n = |V|$, as in the input graph format. Each following line corresponds to the respective vertex's computed C_{HC} value. The values should be formatted as an ASCII integer and doubles respectively.

4 Submission Instructions

Please follow these instructions carefully.

- You should submit two files:
  1. A PDF containing your typed written assignment: the algorithm, the feasibility proof, the run-time and space complexity analysis and proof, and the implementation report and evaluation.
  2. A single source code file (C++, Python, or Java) that implements your algorithm and compiles or runs in the templates provided at `https://github.gatech.edu/ucatalyurek7/cse6140-Fall18.git`. This file will be used to replace `hcc.cpp`, `HCC.java`, or `hcc.py` appropriately when testing
- Your code should contain no dependencies beyond what the templates contain (standard libraries are okay) and should compile or run on a vanilla installation of Ubuntu 18.04 with only `build-essential`, `default-jdk`, `python3`, and `python2.7` installed. If you have any doubt about whether it will compile or run, please reach out to the TAs before the deadline.