CS 6363.500 Homework 4

Due Tuesday April 9th on eLearning (grace period ends April 11th at 10am)

March 27, 2019

Please answer each of the following questions.

1. The following two problems can be solved using relatively simple reductions, so we will put a higher emphasis on the proofs of correctness when scoring.

   (a) Describe and analyze an algorithm to compute the *maximum*-weight spanning tree of a given connected edge-weighted graph (edge weights may be positive, negative, or zero).

   (b) A *feedback edge set* of an undirected graph G is a subset F of the edges such that every cycle in G contains at least one edge in F. In other words, removing every edge in F makes the graph G acyclic. Describe and analyze a fast algorithm to compute the minimum weight feedback edge set of a given edge-weighted graph (edge weights may be positive, negative, or zero).

2. Suppose we are given a directed graph G with weighted edges and two vertices s and t. Describe and analyze a *fast* algorithm to find the shortest path from s to t when exactly one edge of G has negative weight. *[Hint: Modify Dijkstra's algorithm. Or don't. We'll put higher emphasis on the running time analysis if you don't.]*

3. After moving to a new city, you decide to choose a walking route from your home to your new office. To get a good daily workout, your route must consist of an uphill path (for exercise) followed by a downhill path (to cool down), or just an uphill path, or just a downhill path. (You'll walk the same path home, so you'll get exercise one way or the other.) But you also want the *shortest* path that satisfies these conditions so that you actually get to work on time.

   Your input consists of an undirected graph G whose vertices represent intersections and whose edges represent road segments, along with a start vertex s and a target vertex t. Every vertex v has an associated value $h(v)$, which is the height of that intersection above sea level, and each edge uv has an associated value $\ell(uv)$, which is the length of that road segment.

   (a) Describe and analyze an algorithm to find the shortest uphill-downhill walk from s to t. Assume all vertex heights are distinct.

   (b) Now suppose we allow some or all vertex heights to be equal. Describe and analyze an algorithm to find the shortest "uphill then downhill" walk from s to t; you may use flat edges in both the "uphill" and "downhill" portions of your walk.

4. The Floyd-Warshall algorithm as described in class returns incorrect results if the graph contains a negative cycle. However, it can be modified to return correct shortest-*walk* distances, even in the presence of negative cycles. Specifically, for all vertices u and v:

   - If u cannot reach v, the algorithm should return $dist[u, v] = \infty$.
   - If u can reach a negative cycle that can reach v, the algorithm should return $dist[u, v] = -\infty$.
   - Otherwise, there is a shortest walk from u to v that happens to be a path, so the algorithm should return its length.

   Describe how to modify Floyd-Warshall to return the correct shortest-walk distances, even if the graph has negative cycles. *[Hint: Go back through the process of deriving Floyd-Warshall using recurrences and dynamic programming.]*

5. You may want to start on this problem after April 2nd's lecture, but part (b) at least should be doable with what you'll see March 28th.

   (a) Suppose you are given a directed graph $G = (V, E)$, two vertices s and t, a capacity function $c : E \to \mathbb{R}^+$, and a second function $f : E \to \mathbb{R}$. Describe a *fast* algorithm to determine whether f is a maximum (s, t)-flow in G.

   (b) Let (S, T) and (S', T') be minimum (s, t)-cuts in some flow network G. Prove that $(S \cap S', T \cup T')$ and $(S \cup S', T \cap T')$ are also minimum (s, t)-cuts in G.