CS/ECE 374 Fall 2018
Homework 8 Problem 2

Anqi Yao (anqiyao2@illinois.edu)
Zhe Zhang (zzhan157@illinois.edu)
Ray Ying (xinruiy2@illinois.edu)

2. Let $G = (V, E)$ be a directed graph with edge lengths that can be negative. Let $\ell(e)$ denote the length of edge $e \in E$ and assume it is an integer. Assume you have a shortest path tree T rooted at a source node s that contains all the nodes in V. You also have the distance values $d(s, u)$ for each $u \in V$ in an array (thus, you can access the distance from s to u in $O(1)$ time). Note that the existence of T implies that G does not have a negative length cycle.

- Let $e = (p, q)$ be an edge of G that is *not* in T. Show how to compute in $O(1)$ time the smallest integer amount by which we can decrease $\ell(e)$ before T is not a valid shortest path tree in G. Briefly justify the correctness of your solution.
- Let $e = (p, q)$ be an edge in the tree T. Show how to compute in $O(m + n)$ time the smallest integer amount by which we can increase $\ell(e)$ such that T is no longer a valid shortest path tree. Your algorithm should output ∞ if no amount of increase will change the shortest path tree. Briefly justify the correctness of your solution.

Solution:

- Let $e = (p, q)$ be an edge of G that is *not* in T. To find the smallest decrease of $\ell(e)$ that will make the shortest path tree T invalid, the main idea is to find the smallest decrease in $\ell(e)$ where taking this edge (with such weight decreased) from p to q will result a smaller distance between s and q, compared to the original distance.

  The smallest integer to return is $\ell(e) - (d(s, q) - d(s, p)) + 1$.

  Justification:
  First, $d(s, q) - d(s, p)$ is the current shortest distance from p to q via T, and can be evaluated with $O(1)$.
  Second, $\ell(e)$ is the length of edge e and can be obtained with $O(1)$.
  Thus by decrease $\ell(e) - (d(s, q) - d(s, p))$, e will now have the length same to the shortest distance from p to q via T.
  Then we decrease $\ell(e)$ further by 1, $\ell(e)$ is now smaller than the shortest distance from p to q via T, and thus is a shorter path from p to q. In this case, T is violated.
  Therefore, the smallest integer to return is $\ell(e) - (d(s, q) - d(s, p)) + 1$.

  Running Time:
  Since $d(s, u)$ and $\ell(e)$ take linear time, the algorithm will take $O(1)$ time.

- Let $e = (p, q)$ be an edge of G that is in T. The main idea is to increase $\ell(p, q)$ by x such that continuing taking the original path will result a larger distance from s to q and $q's$ descendants than taking another path.

The smallest integer to return is $x = MIN\{\ell(u,v)-(d(s,v)-d(s,u))+1\}$, where (u,v) are all edges in G such that v is a descendant of q in T but u is not.
If no such edges exist, return ∞.

Justification:
$MIN\{\ell(u,v)-(d(s,v)-d(s,u))+1\}$ will give the shortest path from any other non-descendant of q to descendants of q plus 1. Increase $\ell(e)$ by such value will promise v(descendant of q) to have the largest distance from p among from any u(non-descendant of q). Thus this value is correct if such edges (u,v) exist. If no such edge exist, it means that any increase in $\ell(p,q)$ will not affect T and thus we output ∞.

Running Time:
The worst case is to compute $O(n)$) descendants of q and then to look through all edges, $O(m)$. Thus in total, this algorithm takes $O(m+n)$ time complexity.

■