ASSIGNMENT 7

This assignment is for students currently enrolled in the Fall 2021 offering of CS 341 only. You may not share or distribute it without explicit written permission from the instructors.

DUE: Tuesday November 30, 11:59 PM. DO NOT COPY. ACKNOWLEDGE YOUR SOURCES.

Please read http://www.student.cs.uwaterloo.ca/~cs341 for general instructions and policies. Note: “Giving” an algorithm means doing the four parts (i)–(iv) as described on the course web page. This goes without saying for all future assignments and exams.

In the lectures, you have seen various polynomial-time algorithms to find shortest (i.e., minimum weight) paths in a directed graphs IF the graph has no negative weight cycles. If a graph HAS a negative weight cycle then going around that cycle more and more times gives paths of smaller and smaller (i.e., more and more negative) weight, so minimum weight paths are not even well-defined. However, it still makes sense to ask for minimum weight simple paths. That’s the topic of this assignment. Recall that “simple” means the path does not repeat vertices. We will consider three versions of the minimum weight simple path (MWSP) problem:

- **DECEDEMWSP problem:**
  Input: A directed graph $G$ with integer weights on the edges, two vertices $s$ and $t$, and an integer $k$.
  Output: Is there a simple path in $G$ from $s$ to $t$ of weight $\leq k$? (The output is YES or NO.)

- **OPTMWSP problem:**
  Input: A directed graph $G$ with integer weights on the edges and two vertices $s$ and $t$.
  Output: The minimum weight of a simple path in $G$ from $s$ to $t$. (The output is a number.)

- **FINDMWSP problem:**
  Input: A directed graph $G$ with integer weights on the edges and two vertices $s$ and $t$.
  Output: A simple path in $G$ from $s$ to $t$ whose weight is minimum. (The output is a path.)

0. [0 marks] Don’t hand this in. Give an example to show that the Bellman-Ford algorithm does not solve OptMWSP, not even for digraphs where there are no edges directed in to $s$ (note that parent pointers will always form a directed tree rooted at $s$ in this case).

1. [10 marks] Give a branch-and-bound algorithm for FindMWSP. Each configuration should include a simple path $s, u_1, u_2, \ldots, u_i$. You may also wish to store the weight of the path. To generate the children of this configuration, consider all edges leaving $u_i$. Clearly describe:
   - how you choose the next configuration to explore
   - how you test for dead-end configurations
• what lower bound you use to discard configurations, and how you compute the lower bound

We are looking for reasonable choices here—there isn’t a unique best answer, and in fact, without implementing and testing (which we are not asking you to do) it would be impossible to know what’s best.

2. [7 marks] Show that the OptMWSP and DecideMWSP problems are equivalent with respect to polynomial-time algorithms. In other words, prove that:

(a) [2 marks] DecideMWSP \leq_P OptMWSP.

*Hint.* This is very easy. Start by assuming you have a polynomial time algorithm for OptMWSP...

Note that you will not be using many-one reductions—they are only appropriate when both problems are decision problems.

(b) [5 marks] OptMWSP \leq_P DecideMWSP.

*Hint.* The obvious approach would be to run the algorithm for DecideMWSP on all possible values of \( k \) to find the smallest one. An overly generous bound on the range of \( k \) is as follows: let \( w = \max \{|w(e)| : e \text{ an edge of } G\} \). Then any simple path in \( G \) has weight in the range from \(-nw\) to \( nw\). This range has \( 2nw + 1 \) integer values, so we do not get a polynomial run time if we try all of them. However, note that \( \log(2nw + 1) \) is \( O(\log n + \log w) \) which is a polynomial in the input size.

**Note:** In fact, FindMWSP is also equivalent to the other two versions with respect to polynomial time, but we’ll leave that to the Challenge Question below.

3. [5 marks] Prove that DecideMWSP \in NP.

Specify the certificate (which must be of polynomial size) and the polynomial-time verification algorithm.

In Assignment 8, you will show that DecideMWSP is NP-complete.

**Challenge Question.** This is for fun and enrichment only. Do not hand in.

1. Show that FindMWSP \leq_P DecideMWSP.