Algorithms, Fall 2022/23, Homework 6

Due Saturday, December 3, 2022, 11:59pm

Problem 1

Given is a weighted undirected graph $G = (V, E)$ and a subset of its edges $F \subseteq E$. An *F-containing spanning tree of G* is a spanning tree that contains all edges from F (the spanning tree might contain other edges as well). Design an algorithm that finds the cost of a minimum-cost F-containing spanning tree of G. Your algorithm should run in time $O(m \log n)$ or $O(n^2)$.

Problem 2

Given is a weighted directed graph $G = (V, E, w)$ with exactly one negative-weight edge. Design an $O(n^2)$ algorithm that determines whether G contains a negative-weight cycle.

Problem 3

The Edmonds-Karp algorithm refines the idea of Ford and Fulkerson in the following way: in every iteration, the algorithm chooses the augmenting path that uses the smallest number of edges (if there are more such paths, it chooses one arbitrarily). Find a graph for which in some iteration the Edmonds-Karp algorithm has to choose a path that uses a backward edge. Run the algorithm on your graph – more precisely, for every iteration (a) draw the current residual graph, (b) show the augmenting path taken by the algorithm, and (c) draw a copy of the original graphs with the flow after adding the augmenting path.

Do not make your graph overly complicated.

Problem 4

Suppose you are running a daycare center and you need to get the children dressed to take them outside for a walk. Every child needs a hat, a pair of mittens, and a winter jacket. The trouble is that all the clothes got mixed up and the children do not remember (or do not want to tell you) which clothes are whose. Every child has certain preferences for which clothes they are willing to put on. For example, Johnny is ok with the green or the red hat, the flowery mittens, and either the long blue jacket, or the purple jacket, or the jacket with yellow stripes. You record the preferences for all the children and now you are trying to figure out if you can get every child dressed properly for the walk.

Formally, suppose there are n children, a hats, b pairs of mittens, and c jackets (there might be extra or missing clothing items so it might happen that we do not have exactly n hats, mittens, or jackets). For every child we have a list of acceptable hats, mittens, and jackets. Design an $O(n^2(a + b + c))$ algorithm that determines whether it is possible to get every child dressed with a hat, a pair of mittens, and a winter jacket that the child finds acceptable. Of course, no two children can share the same item of clothing.

Problem 5

This problem is about reducing one problem to another problem. Suppose we have two problems P and Q. And suppose we have an algorithm $\mathcal{A}_Q$ that solves problem Q. Can we then use the algorithm $\mathcal{A}_Q$ to solve problem P, with only little extra work? (In this context by "little" we mean that the number of extra steps is polynomially bounded.) If yes, we say that we reduced problem P to problem Q.

We will work with the following problems:

Problem P – Hamiltonian path: Given is an unweighted undirected graph G and two vertices s and t. Does there exist a path from s to t that goes through every vertex exactly once?

Problem Q_1 – Longest path: Given is an unweighted undirected graph G and two vertices s and t. Find the length of the longest path from s to t. The path can go through every vertex at most once.

Problem Q_2 – Shortest path with negative weights: Given is a weighted undirected graph G with arbitrary weights (positive, negative, or zeros) and two vertices s and t. Find the length of the shortest path from s to t. The path can go through every vertex at most once.

Here are your tasks:

a) Reduce problem P to problem Q_1. In particular, for a graph G and vertices s, t for which you want to find a Hamiltonian path, create an input graph G_1 and its vertices s_1 and t_1 that you'll use as the input for algorithm $\mathcal{A}_{Q_1}$. The algorithm will output ℓ, the length of the longest path from s_1 to t_1. Use ℓ to determine whether G has an s-t Hamiltonian path.

- Given G, s, and t, describe how to construct G_1, s_1, and t_1.
- Given ℓ for your G_1, s_1, and t_1, describe how to determine whether G has an s-t Hamiltonian path.

b) Reduce problem P to problem Q_2.

- Given G, s, and t, describe how to construct G_2, s_2, and t_2, the input for algorithm $\mathcal{A}_{Q_2}$. Make sure to specify the edge weights for G_2.
- Given ℓ, the length of the shortest s_2-t_2 path in your G_2, describe how to determine whether G has an s-t Hamiltonian path.

Note: you cannot modify the algorithms $\mathcal{A}_{Q_1}$ or $\mathcal{A}_{Q_2}$ – you do not know how they work! Imagine them being a part of a library for which you cannot see the source code. We often refer to such functions as *black boxes*.