MAS 714, Fall 2019

Tutorial 5 / Homework 2

This homework will be graded. Please hand the solutions in before Monday, September 23, 11:30 am.
Solutions can be submitted either via email of in written form.

Problem 1 The Egg-dropping problem is as follows. We have k eggs. We are interested in determining the highest floor b of a building, from which we can drop an egg without breaking. The building has n floors. The eggs are from a hardy species of bird, so any result from 1 to n is possible. All eggs break at the same level b.
The following rules apply:

1. All eggs will break when thrown from a certain (unknown) floor b.
2. All eggs dropped from floors b up to n will break, eggs dropped from 1 up to $b-1$ will not break.
3. Broken eggs cannot be re-used.
4. Eggs that do not break can be re-used (and have the same behavior as all eggs).

We are interested in the number of trials needed to determine b. Note that with one egg, the best way to proceed is to try floors $1, 2, 3, \ldots$ successively until the egg breaks. This strategy would need up to n trials, because b can be anything between 1 and n. With more eggs better strategies exist, for instance with $\log n$ eggs one can use binary search.
We define $T(n, k)$ to be the minimum number of trials needed to find b. Note that the only way to get information about b is to perform trials (i.e., drop an egg from floor i).
Describe a dynamic programming algorithm, that, given n and k, determines the minimum number of trials needed to determine b.
Hint: The recursive formulation should be a minimum over all i and then consider the outcome of the experiment of dropping an egg from floor i. It is OK if the algorithm's running time is polynomial in n, k.
Determine the running time of your algorithm.

Problem 2 The edge connectivity of an undirected graph G is the minimum number of edges that need to be removed from G to disconnect the graph. For instance, a tree has edge connectivity 1.
Describe an efficient algorithm that computes the edge connectivity of a given undirected graph G.
Hint: Construct flow networks from G to solve the problem by finding maximum flows.

Problem 3 **a)** Let G, C, s, t be a flow network with graph G, capacities C, source s and sink t. Suppose we are given a flow $f : E \to \mathbb{R}$, in the form of weights stored with the adjacency list. Describe an efficient algorithm that checks if f is a legal flow, and an efficient algorithm that decides if f is already a maximum flow. The algorithm should run in time $O(m + n)$.
b) Let $G = (L \cup R, E)$ denote a bipartite, undirected, unweighted graph, and M a matching in G. Design an algorithm with time complexity $O(m + n)$ that decides whether M is a maximum matching in G.

Problem 4 We are given a flow network $N = (G, C, s, t)$ and a maximum flow f. N is given as an adjacency list, which also stores f (f is stored only for edges in G, the value of f for non-edges is implicit). Capacities are integers and all $f(u, v)$ are integers.
a) *Show* how to compute a new maximum flow in linear time, if a single edge capacity is increased by 1.
b) *Show* how to compute a new maximum flow, if a single edge capacity is decreased by 1, again in linear time.

Problem 5 Given a weighted, undirected graph (G, W), a minimum weight connecting set is a subset of the edges of minimum total weight, such that the graph stays connected when removing all other edges. This differs from a minimum spanning tree when negative edge weights exist.
Describe an efficient algorithm that finds a minimum weight connecting set for all inputs (G, W).