Advanced Algorithms. Assignment 2

Exercise 3.
Suppose that we are given a directed graph $G = (V, E)$ with a source s and a sink t, where every edge e has a (finite, non-negative) lower capacity l_e and a (finite or infinite) upper capacity c_e, and we want to find a flow f in G that respects these capacity constraints, with $val(f) = k$ (or show that no such flow exists), where the number $k \geq 0$ is also given. All numbers (capacities, flow values) are assumed to be integers.
We assume that directed edges neither enter s nor leave t, and we define $k_0 := \max\{\sum_{e=(s,v)} l_e, \sum_{e=(v,t)} l_e\}$. Obviously, $k \geq k_0$ is a necessary condition for the existence of a flow.
Note that, unlike the ordinary Maximum Flow problem, the value k of the flow is prescribed here, and certain edges must (at least) carry some prescribed positive amounts of flow. Despite these differences, we would like to solve this problem variant via a reduction to Maximum Flow. For the following, recall the reductions LZ-red and CF-red from Lecture Notes 5.

3.1. Let $G(k)$ be the graph obtained from G by introducing the demands $d(s) := -k$, $d(t) := +k$, and $d(v) := 0$ for all other nodes v. Explain in a few lines why the following equivalance is true: G has a flow f with $val(f) = k$ if and only if $G(k)$ has a circulation.

3.2. Let $H(k)$ be the graph obtained from $G(k)$ by LZ-red, where $k \geq k_0$. Explain in a few lines why the following statements about $H(k)$ are true:
The $d(v)$ for $v \neq s, t$ do not depend on k.
We still have $d(s) \leq 0$ and $d(t) \geq 0$.

3.3. Let $J(k)$ be the graph obtained from $H(k)$ by CF-red, where $k \geq k_0$. We denote the new source and sink by s' and t', respectively. (Note that s and t are retained as usual nodes.) Show that G has a flow f with $val(f) = k$ if and only if the maximum $s' - t'$ flow in $J(k)$ saturates all the edges at s' and t'. (Thus, computing this flow and checking saturation solves the problem introduced above.) Advice: Consider the whole chain of reductions and use the already known facts about LZ-red and CF-red from Lecture Notes 5. You should not need to write much.

Exercise 4.
We further elaborate on the problem from Exercise 3. There, a desired flow value k was already given. Now, only G is given, and we want to find the smallest value k_1 such that G has a flow f with $val(f) = k_1$, or recognize that G has no feasible flow at all.

4.1. Suppose that G has a flow, and we run the above method, but with a too small value $k < k_1$. Let g be the maximum flow we have computed in $J(k)$. Due to 3.3 we recognize that g does not saturate all the edges at s' and t'. Let us further use this flow g anyway, but with an incremented value of k. Your task: Explain why the residual graph $J(k+1)_g$ has an augmenting path.

Remark and hint: 4.1 is the only more tricky part. Consider $J(k_1)_g$ and use "Ford-Fulkerson theory" from the course. – However, if you cannot manage 4.1, you can skip it, and use the claim of 4.1 to continue with the rest.

4.2. The result of 4.1 hints to an algorithm: We start with $k := k_0$ and check whether the maximum flow in $J(k)$ saturates all edges at s' and t'. If so, we stop and output $k_1 := k$. If not, we set $k := k + 1$ and search for augmenting paths in $J(k)$. If none exists, we stop and report that G has no flow. If augmenting paths exist, we augment the flow in $J(k)$ as much as possible (as the Ford-Fulkerson algorithm would do). – Argue why this algorithm yields the correct output.

4.3. In the Path Edge Cover problem, we are given a directed acyclic graph A with two distinguished nodes s and t. We wish to find a minimum number of directed $s - t$ paths that cover all edges, that is, every edge must be in at least one of the selected paths. (If no solution exists at all, this shall be recognized, too.) Clearly, this is a special case of the Set Cover problem. Show that Path Edge Cover is also a special case of our minimum flow problem. More precisely: Turn A in polynomial time into a graph G with appropriate lower and upper edge capacities, and briefly show equivalence of the problems. Do not forget that an equivalence has two directions.

4.4. Combine the results from 4.2 and 4.3 to show that Path Edge Cover is solvable in polynomial time. In particular, state and motivate some polynomial time bound, as a function of the graph size.