University of Freiburg
Dept. of Computer Science
Prof. Dr. F. Kuhn
P. Bamberger, P. Schneider

Algorithm Theory
Sample Solution Exercise Sheet 7

Due: Tuesday, 7th of December, 2021, 4 pm

Exercise 1: Vertex Cover Variant *(10 Points)*

Given an undirected graph $G = (V, E)$, a subset $U \subseteq V$ of nodes and a capacity function $c : U \to \mathbb{N}$, we want to cover every edge with the nodes in U, where every node $u \in U$ can cover up to $c(u)$ of its incident edges.
Formally, we are interested in the existence of an assignment $f : E \to U$ such that for all $e \in E$ we have $f(e) \in e$ and for all $u \in U$ it holds $|\{e \in E \mid f(e) = u\}| \leq c(u)$.

Devise an efficient algorithm to determine whether or not such an assignment exists and explain its runtime.

Sample Solution

We formulate the problem as a flow problem. We flow-network looks as follows: We have a source node s, a target node t, one node for each $u \in U$ and one node for each $e \in E$. We have the following edges:

- An edge from s to each $u \in U$ with capacity $c(u)$
- For any $e = \{u, v\} \in E$ an edge from u to e and one from v to e with capacity 1 each (or any integer capacity ≥ 1)
- An edge from each $e \in E$ to t with capacity 1

The problem is solvable iff the maximum flow equals $m = |E|$.

The network has integer capacities, the maximum flow is at most m and the network has $O(m)$ edges, so computing a maximum flow with Ford-Fulkerson takes $O(m^2)$.

Exercise 2: Cycle Elimination *(10 Points)*

Let $G = (V, E, c)$ be a directed graph with capacity function $c : E \to \mathbb{N}$ and let $s, t \in V$. We allow G to contain cycles. We now want to build a DAG (directed acyclic graph) $G' = (V, E', c')$ with $E' \subseteq E$ and $c'(e) = c(e)$ for $e \in E'$ (i.e., we obtain G' by deleting edges from G) that has the same minimum s-t cut capacity as G.

Give an efficient algorithm to compute such a graph G', argue that your algorithm is correct and analyze its runtime.

Sample Solution

We compute a maximum s-t flow of G. Assume there is a flow going along a cycle Z. Let $e \in Z$ be the edge with the smallest flow value among all edges in Z. We set the flow on e to 0 and reduce the flow on all other edges in Z by the corresponding amount. This way, we obtain a valid flow in G of the same size without any flow going along cycles. We now obtain G' by deleting all edges from G with a flow value of 0. G' is a DAG with the same maximum s-t flow and hence the same minimum s-t cut capacity as G.

Computing a flow on G takes $O(m \cdot C)$ where C is the size of a minimum cut in G. Finding a "flow cycle" and changing the flow in it takes $O(m)$. After $O(m)$ iterations, all such cycles are eliminated. The total runtime is hence $O(m \cdot C + m^2)$.