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

Algorithm Theory
Sample Solution Exercise Sheet 7

Due: Tuesday, 22nd of December 2020, 4 pm

Exercise 1: Global Min-Cut *(8 Points)*

Given a simple, undirected graph $G = (V, E)$ and a subset $A \subset V$ with $A \neq \emptyset$ and $A \neq V$, we call the number of edges between A and $V \setminus A$ the *size* of the cut $(A, V \setminus A)$. A min-cut of G is a cut of minimum size.
Give an algorithm that computes the size of a min-cut of G in time $O(|E| \cdot |V|^2)$. Explain why the algorithm is correct and analyze the runtime.

Sample Solution

Pick an arbitrary source node s. For each $v \in V \setminus \{s\}$, we consider the flow network which we obtain by directing each edge incident to s away from s and each edge incident to v towards v. All other edges are replaced by two directed edges. Each edge is assigned capacity 1. We denote this network by (G, s, v). We compute the size of a minimum (s, v)-cut in (G, s, v) using Ford-Fulkerson for all $v \in V \setminus \{s\}$ and finally take the minimum of all computed min-cuts.
Correctness: Let $\mathcal{A}$ be the size of a global min-cut and $\mathcal{B}$ the output of our algorithm. We show $\mathcal{A} = \mathcal{B}$. We assume $|V| \geq 2$, otherwise the statement is trivial.
Let $v \neq s$ and $(S, V \setminus S)$ be a (s, v)-cut in (G, s, v). The capacity of $(S, V \setminus S)$ equals the number of (undirected) edges between S and $V \setminus S$ in G. Hence the size of a global minimum cut in G is at most the size of a minimum (s, v)-cut in (G, s, v) for any $v \neq s$. This implies $\mathcal{A} \leq \mathcal{B}$.
On the other hand, let $(A, V \setminus A)$ be a global minimum cut in G. If $s \in A$, at some point our algorithm computes a minimum (s, v)-cut $(S, V \setminus S)$ in (G, s, v) for some $v \notin A$. For this choice of v, the global min-cut $(A, V \setminus A)$ induces an (s, v)-cut in (G, s, v) with capacity $\mathcal{A}$ and thus we have $\mathcal{A} \geq c(S, V \setminus S)$. Hence we obtain $\mathcal{A} \geq \mathcal{B}$. If $s \notin A$, the proof goes along similar lines.
Runtime: We run Ford Fulkerson $|V|$ times on a graph with maximum flow at most $\Delta \leq |V|$ (as all capacities are 1). One execution thus takes $O(|E| \cdot |V|)$ and therefore the whole algorithm takes $O(|E| \cdot |V|^2)$.

Exercise 2: Scheduling *(12 Points)*

Assume there are n students $s_1, \ldots, s_n$. Each student has finished some individual project and now has to present the results to some professors. There are k professors $p_1, \ldots, p_k$. Each professor p_i hands in a list $L_i \subseteq \{s_1, \ldots, s_n\}$ of students for whose projects he/she is an expert. Each professor p_i is willing to attend at most a_i presentations.
The exam regulations require that at each presentation, x professors that are experts on the topic are present.

(a) Describe a polynomial-time algorithm that computes an assignment of the professors to the student's presentations such that the given constraints are fulfilled, or reports that no such assignment exists. *(6 Points)*

(b) As there is a shortage of professors, the university loosens the requirements such that among the x professors that need to be present at each presentation, at least y need to be an expert on the topic, for some $y < x$. Describe how to construct a feasible schedule in this case. *(6 Points)*

Sample Solution

(a) We build a flow network with source node s, sink t, nodes $s_1, \dots, s_n$ for the students and nodes $p_1, \dots, p_k$ for the professors. For each $i \in \{1, \dots, k\}$, we add the edge (s, pi) with capacity a_i. For each $j \in \{1, \dots, n\}$, we add the edge (s_j, t) with capacity x. Finally, for each $i \in \{1, \dots, k\}$ and $j \in \{1, \dots, n\}$, we add the edge (p_i, s_j) with capacity 1 if $s_j \in L_i$. We compute a maximum flow in this network. A feasible schedule exists if and only if the maximum flow equals $n \cdot x$. In this case, we find a schedule by assigning professor p_i to student s_j if the edge (p_i, s_j) exists and has a flow value of 1.

(b) We take the flow network from (a) with the following adjustments. We add extra nodes $s'_1, \dots, s'_n$ for the students and edges (p_i, s'_j) with capacity 1 if $s_j \notin L_i$. Further, we add edges (s'_j, s_j) for each $j \in \{1, \dots, n\}$ with capacity $x - y$. Then we proceed as in (a).