ETH Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Institute for Operations Research
ETH Zurich HG G21-22

Prof. Dr. Rico Zenklusen and Assistants
Contact: `math.opt@ifor.math.ethz.ch`

IFOR Institute for Operations Research
Department of Mathematics

Fall 2019

Mathematical Optimization – Problem set 8

https://moodle-app2.let.ethz.ch/course/view.php?id=4844

Problem 1: Flow decomposition

Let $G = (V, A)$ be a digraph with arc capacities $u\colon A \to \mathbb{Z}_{\geq 0}$, and let $s, t \in V$ be distinct. Given an s-t flow $f\colon A \to \mathbb{Z}_{\geq 0}$, prove that one can efficiently find s-t paths $P_1, \dots, P_k$ in G and values $\gamma_1, \dots, \gamma_k \in \mathbb{Z}_{\geq 0}$ as well as cycles $C_1, \dots, C_\ell$ in G and values $\eta_1, \dots, \eta_\ell \in \mathbb{Z}_{\geq 0}$ such that $k + l \leq |A|$ and

$$f = \sum_{i=1}^{k} \gamma_i \chi^{P_i} + \sum_{i=1}^{\ell} \eta_i \chi^{C_i} \ ,$$

where, for $S \subseteq A$ and $a \in A$, $\chi^S \in \{0,1\}^A$ is defined by $\chi^S(a) = 1$ for $a \in S$ and $\chi^S(a) = 0$ for $a \notin S$.

Problem 2: Improving over Edmonds-Karp: Blocking flows and Dinic's algorithm

Let $G = (V, A)$ be a directed graph with edge capacities $u\colon A \to \mathbb{Z}_{\geq 0}$, and let $s, t \in V$ be distinct vertices of G. In class, we've seen the Edmonds-Karp algorithm for finding a maximum s-t flow in time $O(m^2 n)$, where $m = |E|$ and $n = |V|$, and we assume $n = O(m)$ without loss of generality. Recall that the Edmonds-Karp algorithm is a variation of the algorithm of Ford and Fulkerson, where we always augment along shortest paths. The analysis showed that there are $O(n)$ augmentation phases, each comprising consecutive augmentations with paths of the same length, and we have seen that each phase can be realized in running time $O(m^2)$.

In this problem, we study a more efficient realization of the $O(n)$ many augmentation phases. Assume that we are given an s-t flow $f\colon A \to \mathbb{Z}_{\geq 0}$ in G. As an alternative to augmenting along paths in the residual graph $G_f = (V, B)$, we can also augment along an s-t flow $f_0\colon B \to \mathbb{Z}_{\geq 0}$ in the network G_f with capacities u_f, with the resulting augmentation of f along f_0 being the s-t flow f' in G defined by

$$f'(a) = f(a) + f_0(a) - f_0(a^R) \quad \text{for all } a \in A.$$

(a) Prove that the augmentation f' is an s-t flow in G with capacities u. Furthermore, show that $\nu(f') = \nu(f) + \nu(f_0)$.

We will see how to find suitable flows f_0, so-called *blocking flows*, such that augmenting along one f_0 essentially replaces all augmentations in a phase of the Edmonds-Karp algorithm. An s-t flow f in a capacitated graph G is called a *blocking flow* if every s-t path in G has an edge saturated by f.

(b) Show that every maximum s-t flow is a blocking flow.

(c) Give an example of a blocking flow that is not a maximum s-t flow.

Recall that the Edmonds-Karp algorithm augments along shortest s-t paths in (V, U_f), where $U_f \coloneqq \{b \in B\colon u_f(b) > 0\}$. Consider the subgraph of (V, U_f) that consists precisely of all vertices and edges that lie on shortest s-t paths in (V, U_f). We call this subgraph the *s-t layered subgraph* of (V, U_f). Indeed, the vertex set of the layered graph can be split into layers, each consisting of vertices with equal distance from s, and the graph only contains edges connecting consecutive layers.

(d) Show every shortest s-t path in (V, U_f) is an s-t path in the s-t layered subgraph of (V, U_f), and vice versa.

With the definitions given above, we are ready to state Dinic's algorithm.

Algorithm 1 (Dinic's algorithm)

Input: Digraph $G = (V, A)$ with capacities $u\colon A \to \mathbb{Z}$, distinct $s, t \in V$.
Output: A maximum s-t flow $f\colon A \to \mathbb{Z}_{\geq 0}$ in G.

1. **Initialization:**
   $f(a) = 0$ for all $a \in A$.
2. **while** $(d_{(V,U_f)}(s,t) < \infty)$ **do:**
   Find a blocking s-t flow f_0 in the s-t layered subgraph of (V, U_f).
   Augment f along f_0.
3. **return** f.

As indicated earlier, we want to prove that augmentations along blocking flows mimic a full phase of the Edmonds-Karp algorithm. This is implied by the following two properties.

(e) Given a flow f, consider a blocking s-t flow f_0 in the s-t layered subgraph of (V, U_f). Prove that there exist paths $(P_i)_{i\in[k]}$ and values $(\gamma_i)_{i\in[k]}$ such that the following holds true.
   - For every $i \in [k]$, P_i is a shortest s-t path in (V, U_f).
   - Consecutively augmenting f along P_i with augmentation volume γ_i for all $i \in [k]$ results in the same flow as augmenting f along f_0.

   Hint: Use a flow decomposition of f_0 (see Problem 1 of this problem set).

(f) Let f be an s-t flow in G, let f_0 be a blocking flow in the s-t layered subgraph of (V, U_f), and let f' be the augmentation of f along f_0. Prove that the distance of s and t in $(V, U_{f'})$ is strictly larger than in (V, U_f).

   Hint: Use point (e) to interpret the augmentation along f_0 as consecutive augmentations along shortest s-t paths in (V, U_f), and exploit that if an arc is used in an augmenting path in a phase of the Edmonds-Karp algorithm, then in the same phase, its reverse arc will not be used (see the proof of Theorem 4.34 in the script).

(g) Conclude that Dinic's algorithm is correct, i.e., that it terminates and returns a maximum s-t flow in G.

It remains to discuss the running time of Dinic's algorithm. To this end, recall that without loss of generality, we assumed $n = O(m)$.

(h) Show that the s-t layered subgraph of (V, U_f) can be constructed in time $O(m)$.

(i) Assume that you are given access to an algorithm that finds a blocking flow in a layered graph with at most m edges and at most n vertices in running time $\beta(m, n)$. Prove that using this procedure, Dinic's algorithm can be implemented with running time $O(n(\beta(m, n) + m))$.

(j) Show that there is an algorithm for finding blocking flows in s-t layered graphs with running time $O(mn)$, and conclude that Dinic's algorithm can be implemented with running time $O(mn^2)$.

Remark: Note that the running time bound achieved in (j) is indeed better than the one proved in class for the Edmonds-Karp algorithm. Using more involved data structures (so-called dynamic trees*), it turns out that the the running time needed for finding a blocking flow can be improved to $O(m \log n)$, thus implying a bound of $O(mn \log n)$ when implementing Dinic's algorithm with dynamic trees.*

Programming exercises

(a) Work through the notebook `08_extremalMinCuts.ipynb`, where you prove uniqueness of inclusion-wise maximal and minimal minimum s-t cuts, characterize them in terms of components of the residual graph, and and implement algorithms for finding these cuts.

(b) Work through the notebook `08_winningPossibilities.ipynb`, where you implement an algorithm for deciding whether a handball team still has the chance to become the sole leader at the end of the season, and test this algorithm on real-world data.