University of Freiburg
Dept. of Computer Science
Prof. Dr. F. Kuhn
Marc Fuchs

Algorithm Theory
Sample Solution Exercise Sheet 9

Due: Friday, 19th of December 2025, 10:00 am

Exercise 1: Fully Saturated Edges? *(14 Points)*

Let $G = (V, E)$ be an $s - t$ flow network with integer capacity's $c_e > 0$ on each edge $e \in E$. We say that edge $e \in E$ is *fully saturated* if the flow uses its full capacity in every maximum $s - t$ flow in G.

(a) Assume we are given a maximum flow f^* of G, describe an algorithm with time complexity $\mathcal{O}(|E|)$ that computes a new maximum flow of G when the capacity of a specific edge $\{u, v\} \in E$ with $c_e \geq 1$ decreases by one unit. *(3 Points)*

(b) Prove that any minimum cut of G exclusively contains fully saturated edges. *(2 Points)*

(c) Prove that an edge $e \in E$ is fully saturated *if and only if* decreasing the capacity of e by 1 decreases the maximum flow value in G. *(5 Points)*
Remark: Notice that in general even if the maximum flow value and the capacities are integers, the flow values per edges don't necessarily need to be integers. However, the values of (min) cuts are always integers. It may make sense to consider cuts for the backward direction and use the result of (b).

(d) Devise an algorithm that computes the number of fully saturated edges in the flow network G and analyze its running time in dependency of E and the value of the maximum flow $|f^*|$. *(4 Points)*
Remark: Use the statement of (c) and the algorithm from (a) as a black box.

Sample Solution

(a) If the previous maximum flow use (at most) the full capacity minus 1 of $\{u, v\}$, the capacity change does not influence the validity of f^* as the maximum flow. Otherwise, we modify the flow as follows:

Since the flow over $\{u, v\}$ cannot be more than $\{u, v\}$'s new capacity, we reduce the flow over $\{u, v\}$ by one unit. Now the flow entering u is one unit more than the flow leaving it and the flow entering v is one unit less than the flow leaving it.

We search in the residual graph for a path from u to v without using the edge $\{u, v\}$. This can be done in time $\mathcal{O}(|V| + |E|)$. If there is such a path we augment the path, and f^* remains the maximum flow even after the capacity change. Otherwise, we must reduce the flow from both s to u and from v to t by one unit. We do so by finding two paths in the residual graph, one from u to s, and one from t to v, along each of which the flow can increase by one unit. There must be such paths, because we had a flow from s to t via $\{u, v\}$. Augmenting these paths restores the condition $f^*_{in}(u) = f^*_{out}(u)$ and $f^*_{in}(v) = f^*_{out}(v)$. Then the value of the new max flow would be $|f^*| - 1$.
Here we perform a few searches in the residual graph, each takes time at most $\mathcal{O}(|V| + |E|)$ (using e.g. BFS). The statement follows since flow networks are always connected and therefore $|V| \in \mathcal{O}(|E|)$.
Remark: If the capacity of the edge is one in G, the edge basically vanished, however, this algorithm still works.

(b) For contradiction assume there is a minimum cut (A, B) of G such that some edge $e = \{u, v\}$ crossing that cut, say $u \in A$ and $v \in B$, is not fully saturated. Let f be the max flow where e is not using its full capacity. By definition of a max flow we do not have an augmenting path and therefore $|f| = c(A, B)$. Since e does not use full capacity in f we contrarily have $|f| < c(A, B)$.

(c) Let k be the value of the maximum flow, i.e., the capacity of a minimum cut in G. Let G_0 be the new flow network where the capacity of e is reduced by one. We argue by contradiction for both directions.

$\Longrightarrow$ Suppose decreasing the capacity does not decrease the size of the max flow. Then there is a flow in this new network G_0 of value k that does not use the full capacity of e in G. This corresponds to a flow of size k in G with the same property, thus there is a max flow that does not use the full capacity of e (and thus by definition e is not fully saturated). Contradiction.

$\Longleftarrow$ Suppose e is not saturated in every maximum $s-$t flow. If such a flow (where e is not saturated) is an integer flow we are done: This flow would have value k in G_0, contradicting the assumption. But we can not assume an integer flow, so let's do something else instead:

Indeed, if e is not saturated in some maximum flow, then e does not occur in any min cut (see previous task). Since every cut has an integer capacity, we can assume that (A, B) is the cut including e with smallest capacity (among all cuts that include e in G) say k_e. Since the min cut in G has value k and our cut (A, B) is by assumption not a min cut, we have $k_e > k$. That cut (A, B) in G_0 has capacity $k_e - 1 \geq k$ that is not smaller than the min cut of size k. Hence, the size of the min cut and thus also the max flow does not decrease. Contradiction.

(d) Create a counter and initially set it to 0 (that counter will keep track of the fully saturated edges in G). Run Ford-Fulkerson and save the max flow value $|f^*|$ (note that Ford-Fulkerson will compute integer flows if all capacities are integers, so we can assume that), then iterate through every edge once and check: If $f^*(e) < c_e$, don't do anything and move on to the next edge (since e can't be fully saturated)... If $f^*(e) = c(e)$ that edge e is a potential candidate for a fully saturated edge. To verify if it really is, run the algorithm from part (a) of the exercise and if the new max flow decreases, we increment the counter by 1 (regarding part (c) this decrease implies that e is fully saturated) and continue until we have checked all edges. The runtime of Ford-Fulkerson is $\mathcal{O}(|E| \cdot |f^*|)$, each of the subsequent $|E|$ checks takes $\mathcal{O}(|E|)$ time, leading to an overall runtime of $\mathcal{O}(|E| \cdot (|f^*| + |E|))$.

Exercise 2: Work Schedule *(6 Points)*

Assume you want to design a work schedule for a hospital for the next n days. The hospital employs k doctors. On day $1 \leq i \leq n$, **exactly** p_i doctors need to be present in the hospital as less doctors can not provide an optimal health service and more doctors would be too expensive. Each doctor $1 \leq j \leq k$ provides a set $L_j \subseteq \{1, \ldots, n\}$ of days on which they are willing to work.

(a) Describe a polynomial-time algorithm that either: *(3 Points)*

- Returns a list $L'_j \subseteq L_j$ of working days for each doctor j such that on day i, exactly p_i doctors are present; or
- Reports that there is no such set of lists that fulfills the given constraints.

(b) The hospital finds that the doctors tend to submit lists that are too restrictive, and consequently, it often happens that there is no feasible working schedule. Thus, the hospital relaxes the requirements in the following way:

Each doctor j can be forced to work on up to 5 days which are *not* in their list L_j.

Give a polynomial-time algorithm to solve this problem. The algorithm should either: *(3 Points)*

- Return a list L'_j of working days for each doctor j with $|L'_j \setminus L_j| \leq 5$ such that on day i, exactly p_i doctors are present; or
- Report that there is no such set of lists that fulfills the given constraints.

Sample Solution

(a) We translate the problem into a flow network construction:

- Create a source node s and a sink node t.
- For each doctor $j \in \{1, \ldots, k\}$, create a node α_j. Add an edge from s to α_j with capacity $|L_j|$ (note: any capacity $\geq |L_j|$ would also suffice)
- Draw an edge with capacity 1 from α_j to β_i if and only if $i \in L_j$. This represents doctor j's willingness to work on day i.
- Finally, draw an edge with capacity p_i from β_i to t to ensure the staffing requirement for day i is met.

We compute a maximum flow with integer flow values.

Result: There is a feasible work schedule if and only if the maximum flow value of the network equals $\sum_{i=1}^{n} p_i$.

Schedule Construction: Doctor j is assigned to work on day i if there is a flow (of value 1) from α_j to β_i.

(b) We extend the network constructed in part (a) to accommodate the relaxation:

- Keep the structure from (a) (nodes s, t, α_j, β_i and their respective edges).
- For each doctor j, introduce an additional node α'_j.
- Add an edge from s to α'_j with capacity 5. This capacity limits the number of "forced" days for doctor j to at most 5.
- Add an edge from α'_j to β_i with capacity 1 if $i \notin L_j$. This allows doctor j to be assigned to day i even if it is not in their preferred list L_j.

We compute a maximum flow with integer flow values.

Result: There is a feasible work schedule if and only if the maximum flow value of the network equals $\sum_{i=1}^{n} p_i$.

Schedule Construction: Doctor j has to work on day i if there is a flow of value 1 from either α_j or α'_j to β_i.