1. The general network flow problem with bounded capacities (GNFPC). Recall that we consider a directed graph $G = (V, A)$ together with a function $V \ni i \to b_i \in \mathbb{R}$ defining an external supply; we assume $\sum_i b_i = 0$. We are also given a function $c : A \to \mathbb{R}_+$, c_{ij} is the cost in the arc $(i, j) \in A$ and now, additionally, a function $u : A \to \mathbb{R}_+$ determining maximal capacities. Then a flow $f = (f(i, j))_{(i,j)\in A}$ is **feasible** if $0 \leqslant f(i, j) \leqslant u_{ij}$ for every arc $(i, j) \in A$ and $\sum_{j\in \mathrm{In}(i)} f(j, i) + b_i = \sum_{j\in \mathrm{Out}(i)} f(i, j)$ for every vertex i. As before, we want to minimize $\sum_{(i,j)\in A} c_{ij} f(i, j)$ over all feasible flows.

The basic simplex algorithm can be adjusted to cover such a case. Try to analyze the following.

(a) Every basic solution f of GNFPC is associated with a set of edges T forming a spanning tree together with a partition of the remaining arcs $A \setminus T$ into two parts D and U so that $f(i, j) = 0$ for $(i, j) \in D$ and $f(i, j) = u_{ij}$ for $(i, j) \in U$.

(b) Having some basic solution which is feasible, we find p_i's so that the reduced costs $\overline{c_{ij}} = c_{ij} - (p_i - p_j)$ become zero on the arcs belongind to the given tree.

(c) Then the test for optimality is: $\overline{c_{ij}} \geqslant 0$ for all $(i, j) \in D$ and $\overline{c_{ij}} \leqslant 0$ whenever $(ij) \in U$.

(d) If f failed the test for optimality then we pick an arc (i, j) witnessing the fact. Incorporating this arc to the tree we get the unique circle C. We orient C as follows: if $(i, j) \in D$ then (i, j) is a forward arc; otherwise, it becomes backward.

(e) We can determine a parameter $\theta \geqslant 0$ and increase the flow on the forward arcs by θ and decrease it on the backward arcs by θ so that we obtain another basic feasible solution. Then we update T, D, U and repeat.

2. Prove that if GNFPC has at least one feasible solution then such a solution can be found using the Ford-Fulkerson algorithm.

3. The following shows that the problem GNFPC can be seen as a transportation problem TP (we assume that u_{ij} are finite for all arcs). We form TP as follows.

(a) Every arc $(i, j) \in A$ is treated as a provider in TP offering u_{ij}.

(b) Every vertex i becomes a recipient demanding $\sum_{k\in \mathrm{Out}(i)} u_{ik} - b_i$.

(c) For every arc (i, j) (which is now a provider) the transportation cost is zero to i and c_{ij} to j. We think that, in the transportation problem, the cost of sending anything from the 'provider' (i, j) to the 'recipient' k different from i, j is infinite.

Try to find the correspondence between solutions of GNFPC and the resulting transportation problem,

4. **A version of the max flow problem.** Consider a directed graph $G = (V, A)$ with a source $s \in V$ and a target $t \in V$. This time we have a function $u : V \to \mathbb{R}_+$ fixing the capacities of the vertices (no bounds on capacity of the arcs). We want to determine the maximal flow from s to t respecting u, so that

$$\sum_{k \in \mathrm{In}(i)} f(k, i) = \sum_{k \in \mathrm{Out}(i)} f(i, k) \leqslant u(i),$$

for every $i \in V \setminus \{s, t\}$.

Check that such a problem reduces to the classical max flow problem by the following. For every $i \in V \setminus \{s, t\}$ form two vertices i', i'' connected by an arc (i', i'') of capacity $u(i)$ (add all 'obvious' arcs with infinite capacity).

5. Consider a directed graph $G = (V, A)$ with two different vertices $s, t \in V$ fixed. Say that the connectivity of G (between s ant t) is the maximal number of directed paths from s to t going through pairwise disjoint sets of vertices. Say also that the vulnerability in the this context is the minimal number of vertices blocking every way from s to t.

Apply the max flow problem to prove that connectivity is equal to vulnerability.

6. How would you solve the max flow problem on an undirected graph with fixed capacities of the edges?

7. You have a silver wire of length 15 m. You can sell i-meter pieces for c_i per piece, $i = 1, 2, 3, 4$. How do you cut your treasure?