Exercises Discrete Optimization, Utrecht 2012

(to be handed in one week after each lecture, deadlines are not *too* strict)

2.1 Explain why $O(m \log m) = O(m \log n)$ if n and m are the number of nodes *resp.* edges of a simple graph.

2.2 Solve the problem of finding "min cost spanning trees" of a given graph $G = (E, V)$ when the cost function is

$$c(T) = \sqrt{\sum_{ij \in T} c_{ij}^2}$$

for spanning tree $T \subseteq E$.

2.3 What about the problem of computing a spanning tree with maximum cost (relative to given edge costs $c : E \to \mathbb{R}$)?

2.4 Let T be a tree with edge weights. For two nodes i and j of T, let α_{ij} denote the smallest weight of an edge on the (unique) path from i to j. Describe an algorithm for computing all α_{ij} in $O(n^2)$.
(Here, as usual, n denotes the number of nodes.)

2.5 Let $G = (V, E)$ be a graph with edge weights $c : E \to \mathbb{R}$ and let T^* be a corresponding min cost spanning tree. For fixed $e \in E$, let $[\underline{c}, \overline{c}]$ denote the largest interval such that T^* remains optimal if c_e is changed to any other value $c \in [\underline{c}, \overline{c}]$. Describe an efficient algorithm for computing $[\underline{c}, \overline{c}]$. (Hint: Distinguish between $e \in T^*$ and $e \in E \backslash T^*$.)

2.6 A $1-$ *tree* in $G = (V, E)$ is a subgraph of G consisting of a spanning tree plus one additional edge. Show that, relative to given edge costs $c : E \to \mathbb{R}$, a min cost $1-$ tree can be obtained by first computing a min cost spanning tree and then adding the least cost non-tree edge.

3.1 Verify the details in Example 3.5 (matching matroids). What are the consequences for matching problems with weight functions of type $w_{ij} = p_i + p_j$ for certain $p : V \to \mathbb{R}$?

3.2 Show that if $\mathcal{B} \subseteq 2^S$ is the set of bases of some matroid M, then so is $\mathcal{B}^* = \{S \backslash B | B \in \mathcal{B}\}$. (The corresponding matroid M^* is the *dual* of M.)

3.3 Let $r : 2^S \to \mathbb{Z}_+$ be the rank function of a matroid on $S = \{1, ..., n\}$. Let $c_1 \geq ... \geq c_n \geq 0$. Show that the linear program

$$\max\{c^T x \mid x(R) \leq r(R) \ \forall R \subseteq S, \ x \geq 0\}$$

can be solved "greedily" by starting with $x = 0$, then raising x_1 until some constraint $x(R) \leq r(R)$ gets tight, then raising x_2 until some constraint $x(R') \leq r(R')$ gets tight, *etc.*

3.4 Let $S = \{1, 2, 3, 4\}$ and let M be the matroid whose base set $\mathcal{B}$ consists of all 2-element subsets of S. Is M graphic, linear, or a matching matroid?

4.1 Show that, for arbitrary cost functions, the problem of finding shortest simple $s-t$ paths is as difficult as solving TSP.

4.2 Formulate an (integer) linear program for computing shortest $s-t$ paths. (you may assume that all costs are non-negative.)

4.3 Show by means of an example that Dijkstra may fail to compute a shortest $s-t$ path if negative cost values are allowed.

4.4 Let $G = (V, E)$ be a graph with edge costs $c : E \to \mathbb{R}$. Show that the length of a shortest $s-t$ path is given by

$$\max\{d_t - d_s \mid d_j \leq d_i + c_{ij} \ \forall ij \in E\}.$$

5.1 Given a digraph $G = (V, E)$, define the *arc connectivity* as the minimum number of edges we need to remove such that the resulting graph is *disconnected* in the sense that it contains a pair (i, j) of vertices for which there is no directed $i-j$ path. Similarly, the *node connectivity* is defined to be the minimum number of nodes to be removed in order to obtain a disconnected subdigraph. How can the node *resp.* arc connectivity be computed efficiently?

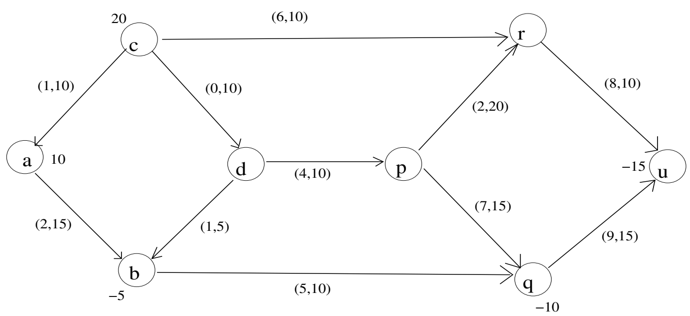
5.2 Let $G = (V, E)$ be a directed graph with edge weights. Show: If $\delta(R_1) \subseteq E$ and $\delta(R_2) \subseteq E$ are min $s-t$ cuts, then so are $\delta(R_1 \cap R_2)$ and $\delta(R_1 \cup R_2)$.

5.3 If augmenting paths with maximum residual capacity are chosen for flow augmentation in each step, may it happen that the increase in flow value in some step is larger than in the preceeding step?

5.4 If augmenting paths are chosen that contain as few reverse arcs as possible, is the corresponding flow augmentation procedure efficient?

6.1 Suppose that we are given a connected digraph G whose underlying graph is eulerian. We seek to reverse some arcs in G such that the resulting digraph has a directed eulerian tour. If c_{uv} is the cost of "reversing" arc uv, *i.e.*, replacing uv by vu, show that finding an optimal (i.e. min cost)set of arcs to be reversed can be formulated as a min cost flow problem.
(Hint: Lower capacity bounds)

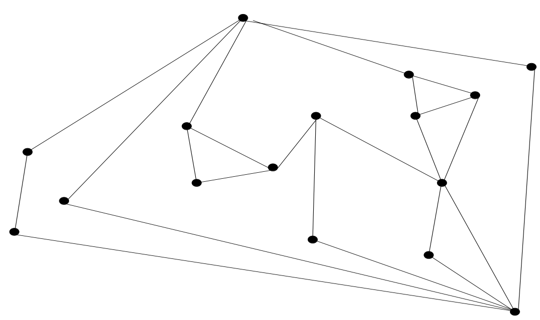
6.2 Consider the network



where each arc is labeled (c_{ij}, u_{ij}) and node balances are as indicated (nodes d, p and r have balance zero). Find a corresponding min cost flow and prove optimality by specifying corresponding node potentials π and reduced costs c^π.

6.3 Search for an efficient algorithm solving the min mean cycle problem, *i.e.*, min $w(C)/|C|$, for given rational edge weights $w : E \to \mathbb{R}$.

7.1 Construct a maximum matching in the graph shown below



7.2 A *line* of a matrix is a row or a column of the matrix. Show that the minimum number of lines containing all the ones of a $(0,1)$-matrix equals the maximum number of 1's, no two of which are in the same line.

7.3 Two people play a game on a graph G by alternately selecting distinct vertices $v_0, v_1, v_2, ...$ such that, for $i > 0$, v_i is adjacent to v_{i-1}. The last player able to select a vertex wins. Show that the first player has a winning strategy if and only if G has no perfect matching.

7.4 ("stable marriages") There are n men and n women. Each man has a certain *ranking*, *i.e.*, an ordering $\pi = (\pi_1, ..., \pi_n)$ of the women, meaning that he prefers woman π_i to woman π_{i+1}. Similarly, every woman has a ranking $\mu = (\mu_1, ..., \mu_n)$ of the men. An assignment ("marriage") of the men to the women is *stable* if there is no pair (i, j) such that man i and woman j prefer each other to their respective partners. Search (the literature) for a procedure to construct stable matchings, describe it shortly and apply it to exhibit a stable matching for $n = 5$ with preference lists as follows for men: (3,5,2,1,4),(4,3,5,1,2), (4,1,3,2,5), (1,3,2,5,4), (4,2,3,1,5) and women: (5,4,3,1,2), (5,1,3,2,4), 5,4,1,3,2), (5,3,1,2,4) and (5,3,2,1,4).

8.1 Prove Prop. 8.1 by LP duality.

8.2 Let $P, Q \subseteq \mathbb{R}^n$ be polytopes. Show that

$$P - Q := \{p - q | p \in P, q \in Q\}$$

is a polytope as well.

8.3 Let $G = (V, E)$ be an (undirected) graph and let $A \in \mathbb{R}^{V \times E}$ be its node-edge incidence matrix. Is A totally unimodular?

8.4 Let $I_1, ..., I_n$ be closed (and bounded) intervals on the real line. Show that the problem of finding a maximum number of pairwise disjoint intervals I_j can be solved by Linear Programming.

9.1 Show how an efficient algorithm A for solving the decision version of VC ("Given graph G and integer k, does there exist a vertex cover of size at most k?") can be used to design an efficient algorithm for solving the minimum vertex cover problem ("Given G, find a min size vertex cover!").

9.2 Explain why LP-feasibility ("Given (A, b), $\exists x$ with $Ax \leq b$?") is in NP $\cap$ co-NP.

9.3 Show that SAT $\preceq$ 3-SAT.

9.4 Give an explicit reduction showing that VC $\preceq$ SAT

10.1 The *bin packing* problem asks to pack n items of size $a_1, ..., a_n \leq 1$ into the least possible number of bins of size 1 each (in such a way that the total size of items packed into a bin does not exceed 1). What about (F)PTAS for solving this problem?

10.2 Consider the *multiobjective* version of TSP: Given a graph G with edge weights $c_e \geq 0$ (cost) and $l_e \geq 0$ (length). Describe a $3-$approximation algorithm, exhibiting a tour whose cost and length are at most 3 times the optimum cost *resp.* length.

10.3 Prove or disprove: Given graph G with edge costs $c_e \geq 0$ and an even size subset W of the nodes, the cost of a min cost perfect matching in $G[W]$ is bounded from above by the cost of a min cost perfect matching in G.

10.4 Consider the directed version of TSP ("Given a digraph G with non-negative edge weights, find a shortest directed tour visiting all nodes"), assuming that the triangle inequality holds. Find a $O(\log n)$ approximation algorithm for this problem.
[Hint: If n, the number of nodes is a power of two, apply successive min cost perfect matching.]