Matthias Krause
Alexander Moch

Mannheim, 2020-05-19

CS 550 Algorithmics, Spring 2020

Additional Exercise Sheet

Exercise 1.1:
For the following two-dimensional optimization problems: solve the problems graphically and determine how many extremal points exist.

1. **maximize** x_2
   subject to $x_1 + x_2 \leq 5$
   $-x_1 + x_2 \leq 3$

2. **minimize** $-x_1 + x_2$
   subject to $x_1 \leq 7$
   $x_2 \leq 4$
   $x_2 \geq 0$

3. **maximize** $13x_1 - 27x_2$
   subject to $8x_1 + 12x_2 \leq 19$
   $2x_1 + 3x_2 \geq 3$

4. **minimize** $x_1 + x_2$
   subject to $x_1^2 + x_2^2 \leq 1$

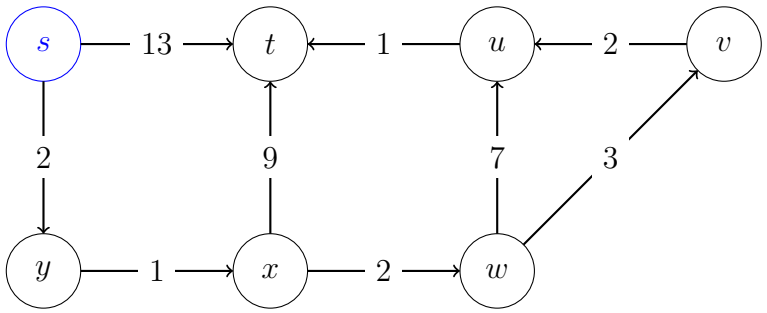
Exercise 1.2: (3 - 4 minutes per subtask)
In this exercise, we consider the simplex algorithm **as introduced in the lecture notes**. Convert the following linear programs to normal form, create the simplex tableaus (you may need the auxiliary tableau) and solve them using the simplex method.

1. **minimize** $-2x_1 + 5x_2$
   subject to $4x_2 \geq 1$
   $2x_1 + 1x_2 \geq 2$
   $-2x_1 - 3x_2 \geq 4$
   $x_1 \ , \ x_2 \geq 0$

2. **maximize** $1x_1 - 2x_2$
   subject to $2x_1 - 1x_2 \geq -4$
   $-1x_1 + 4x_2 \leq 8$
   $1x_1 + 1x_2 \leq 0$
   $x_1 \geq 0$

3. **maximize** $3x_1 + 2x_2 + 1x_3$
   subject to $1x_2 - 1x_3 = 0$
   $1x_1 + 2x_2 + 3x_3 \leq 5$
   $x_1 \ , \ x_2 \ , \ x_3 \geq 0$

Exercise 1.3: (8 - 10 minutes)
Given the following graph $G = (V, E)$ with the weight function ω and the source node s.



Draw the graph $G_\pi = (V_\pi, E_\pi)$ including the values $v.d$ for all $v \in V_\pi$ before iteration 1 and after each iteration i, $0 < i < 7$, during the first phase of `BellmanFord`(G, ω, s) under the condition that in each iteration the edges are processed in lexicographic order, i.e.

$$(s,t),(s,y),(u,t),(v,u),(w,u),(w,v),(x,t),(x,w),(y,x).$$

You may use the table below if it helps you.

	I		R_1^*		R_2^*		R_3^*		R_4^*		R_5^*		R_6^*	
	$v.d$	$v.\pi$	$v.d$	$v.\pi$	$v.d$	$v.\pi$	$v.d$	$v.\pi$	$v.d$	$v.\pi$	$v.d$	$v.\pi$	$v.d$	$v.\pi$
s														
t														
u														
v														
w														
x														
y														

Exercise 1.4:
Let $G = (V, E, c)$ be a flow network with capacity function $c : E \longrightarrow \mathbb{R}^{\geq 0}$ and source $s \in V$ and sink $t \in V$, $s \neq t$. The set of nodes is $V = \{s, t, a, b, c, d, e\}$. The set of edges E and their corresponding capacity $c(e)$ can be found in the following table:

E	(s,a)	(s,c)	(a,b)	(a,t)	(b,c)	(b,t)	(c,a)	(c,d)	(d,b)	(d,e)	(e,b)
$c(e)$	5	9	5	3	3	20	4	7	2	4	3

a) Perform the Edmonds-Karp algorithm on the flow network G and draw the residual network after every iteration.

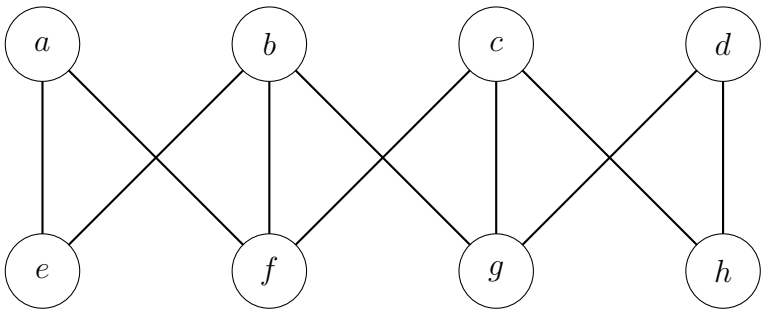
b) Find a minimal cut in the flow network depicted in exercise part (a). **Prove your claim!**

Exercise 1.5: (4 - 5 minutes)
Consider the complete bipartite graph $K_{n,n} = (V, W, E)$ with

$$\begin{aligned} V &= \{v_1, \ldots, v_n\}, \\ W &= \{w_1, \ldots, w_n\}, \\ E &= \{\{v_i, w_j\} \mid i = 1, \ldots, n,\ j = 1, \ldots, n\}. \end{aligned}$$

How many different maximal matchings does $K_{n,n}$ have?

Exercise 1.6: (2 - 3 minutes per subtask)
Consider the following bipartite graph $G = (V, W, E)$ with $V = \{a, b, c, d\}$, $W = \{e, d, f, g\}$ and the edges corresponding to the illustration:



a) Draw the flow network corresponding to G.

b) Draw the residual network and an augmenting path corresponding to the (non-maximal) matching $M = \{\{a, f\}, \{b, g\}, \{c, h\}\}$. Pick the augmenting path which is lexicographically smallest.

Exercise 1.7: (4 - 5 minutes)
The problem Vertex Cover decides if for given $G = (V, E)$ and $k \in \mathbb{N}$, the undirected graph G has a vertex cover of size at most k. $V' \subseteq V$ is called vertex cover in G if $v \in V'$ or $w \in V'$ for all edges $e = \{v, w\} \in E$.

Describe a polynomial reduction from 3-SAT to Vertex Cover. You may use results from the lecture and the tutorials.

Exercise 1.8:
Given a sequence of natural numbers $a_1, \ldots, a_n$. Describe a polynomial time algorithm which decides if there is some subset $I \in \{1, \ldots, n\}$ such that $\sum_{i \in I} a_i = n$.

Exercise 1.9: (1 - 2 minutes for each definition and 2 - 3 minutes for each Π_i)
Given a computational problem Π, we learned the following attributes characterizing the complexity of Π:

- $\Pi \in \mathcal{PTIME}$,
- $\Pi \in \mathcal{NP}$,
- Π is $\mathcal{NP}$-hard,
- $\Pi \in \mathcal{P}$,
- $\Pi \in \mathcal{NPC}$,
- Π is strongly $\mathcal{NP}$-hard.

a) State the definitions of these attributes.

b) Assuming $\mathcal{P} \neq \mathcal{NP}$, decide which of the attributes given above are true for the following computational problems Π_1 to Π_{14}:

- Π_1: **Input:** A directed graph $G = (V, E)$, a number $k \in \mathbb{N}$ and nodes $s, t \in V$. **Output:** TRUE if there is a directed path with less than k edges from s to t. FALSE otherwise.
- Π_2: **Input:** An undirected bipartite graph $G = (V, W, E)$. **Output:** A maximal matching in G.
- Π_3: **Input:** A boolean formula $M = M_1 \vee \cdots \vee M_k$ in disjunctive normal form over the set of variables $\mathcal{X}_n = \{X_1, \ldots, X_n\}$. **Output:** TRUE if M is satisfiable. FALSE otherwise.
- Π_4: **Input:** An undirected graph $G = (V, E)$ where $|V| = 3 \cdot m$ for some $m \in \mathbb{N}$. **Output:** TRUE if G has a clique $V' \subseteq V$ with $|V'| =$ m. FALSE otherwise.
- Π_5: **Input:** A boolean formula $C = C_1 \wedge \cdots \wedge C_k$ in conjunctive normal form over $\mathcal{X}_n = \{X_1, \ldots, X_n\}$ where each clause C_i contains at most 4 literals. **Output:** TRUE if C is satisfiable. FALSE otherwise.
- Π_6: **Input:** A sequence of natural numbers $a_1, \ldots, a_n$ and another number $A \in \mathbb{N}$. **Output:** TRUE if there is a subset $I \subseteq \{1, \ldots, n\}$ with $\sum_{i \in I} a_i = A$. FALSE otherwise.
- Π_7: **Input:** A CNF-Formula $C = C_1 \wedge \cdots \wedge C_m$ over $\mathcal{X}_n = \{X_1, \ldots, X_n\}$. **Output:** TRUE if there is an assignment $b \in \{0, 1\}^n$ that does <u>not</u> satisfy C. FALSE otherwise.
- Π_8: **Input:** A DNF-Formula $M = M_1 \vee \cdots \vee M_m$ over $\mathcal{X}_n = \{X_1, \ldots, X_n\}$. **Output:** TRUE if there is an assignment $b \in \{0, 1\}^n$ that does <u>not</u> satisfy M. FALSE otherwise.
- Π_9: **Input:** A sequence of natural numbers $a_1, \ldots, a_n$ where $1 \leq a_i \leq n$ for all $i \in \{1, \ldots, n\}$. **Output:** TRUE if there is a subset $I \subseteq \{1, \ldots, n\}$ with $\sum_{i \in I} a_i = \sum_{i \notin I} a_i$. FALSE otherwise.
- Π_{10}: **Input:** A distance matrix for n cities, $D = (d_{ij})_{i,j=1}^{n}$, $d_{ij} \in \mathbb{N}$ **Output:** A shortest roundtrip π through the cities, i.e. a permutation $\pi \in \sigma_n$ such that $d_{\pi(1),\pi(2)} + d_{\pi(2),\pi(3)} + \cdots + d_{\pi(n),\pi(1)}$ is minimal.
- Π_{11}: **Input:** A CNF-Formula $C = C_1 \wedge \cdots \wedge C_m$ over $\mathcal{X}_n = \{X_1, \ldots, X_n\}$. **Output:** An assignment $b \in \{0, 1\}^n$ which fulfills a maximal number of clauses.

- Π_{12}: **Input:** A directed graph $G = (V, E)$, a number $k \in \mathbb{N}$ and $s, t \in V$.
  Output: TRUE if there is a directed path from s to t with at least k edges. FALSE otherwise.

- Π_{13}: **Input:** An undirected graph $G = (V, E)$.
  Output: TRUE if G has a clique of size $|V| - 2$. FALSE otherwise.

- Π_{14}: **Input:** A distance matrix for n cities, $D = (d_{ij})_{i,j=1}^{n}$, $d_{ij} \in \{1, 2\}^n$
  Output: A shortest roundtrip π through the cities, i.e. a permutation $\pi \in \sigma_n$ such that $d_{\pi(1),\pi(2)} + d_{\pi(2),\pi(3)} + \cdots + d_{\pi(n),\pi(1)}$ is minimal.

Example:

- Π_0: **Input:** A weighted graph $G = (V, E, w)$ with $w : E \to \mathbb{Z}$ and a node $s \in V$.
  Output: A shortest path tree $G_\Pi = (V_\Pi, E_\Pi)$ with root s.
  Solution:
  $\Pi_0 \in \mathcal{PTIME}$ *as the Bellman-Ford algorithm can compute it in time* $\mathcal{O}(|E| \cdot |V|)$.
  $\Pi_0 \notin \mathcal{P}, \mathcal{NP}, \mathcal{NPC}$ *as* Π_0 *is not a decision problem.*
  Π_0 *is not (strongly)* $\mathcal{NP}$*-hard since* $\Pi_0 \in \mathcal{PTIME}$.

Exercise 1.10: (4 - 5 minutes)
Consider problem Π_{12} from Exercise 1.9. Describe an efficient Π_{12}-oracle algorithm which solves the directed Hamiltonian circuit problem.