Matthias Krause
Alexander Moch

Mannheim, 2020-05-19

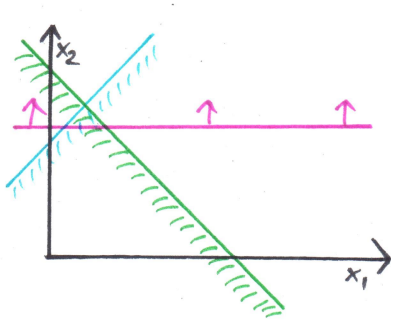
CS 550 Algorithmics, Spring 2020

Additional Exercise Sheet

Exercise 1.1:

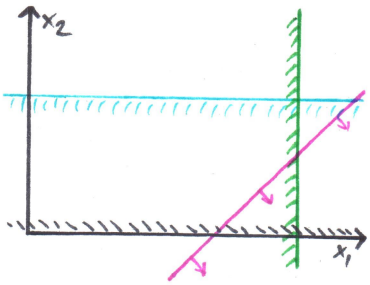
For the following two-dimensional optimization problems: solve the problems graphically and determine how many extremal points exist.

1. $$\begin{aligned} &\textbf{maximize} && x_2 \\ &\textbf{subject to} && x_1 + x_2 \leq 5 \\ & && -x_1 + x_2 \leq 3 \end{aligned}$$



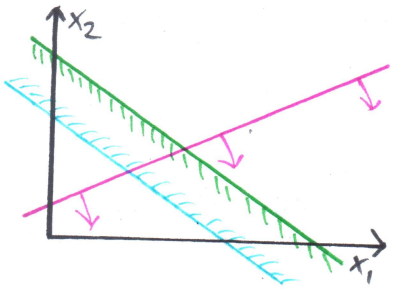
This problem has only one extremal point. The optimal solution is at $(4, 1)$.

2. $$\begin{aligned} &\textbf{minimize} && -x_1 + x_2 \\ &\textbf{subject to} && x_1 \leq 7 \\ & && x_2 \leq 4 \\ & && x_2 \geq 0 \end{aligned}$$



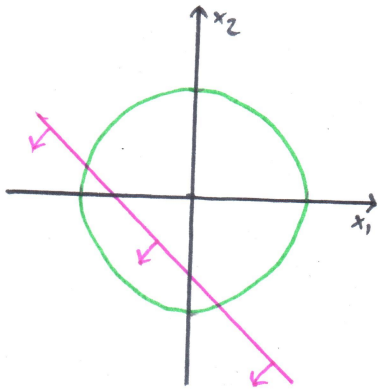
This problem has two extremal points. The optimal solution is at $(7, 0)$.

3. $$\begin{aligned} &\textbf{maximize} && 13x_1 - 27x_2 \\ &\textbf{subject to} && 8x_1 + 12x_2 \leq 19 \\ & && 2x_1 + 3x_2 \geq 3 \end{aligned}$$



This problem has no extremal points and it is unbounded. No optimal solution.

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



This problem has infinitly many extremal points. The optimal solution is at $(-\sqrt{0.5}, -\sqrt{0.5})$. Please note that this is not a linear optimization problem.

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. $$\begin{aligned} \textbf{minimize} \quad & -2x_1 + 5x_2 \\ \textbf{subject to} \quad & 4x_2 \geq 1 \\ & 2x_1 + 1x_2 \geq 2 \\ & -2x_1 - 3x_2 \geq 4 \\ & x_1 \;,\; x_2 \geq 0 \end{aligned}$$

This problem does not have a feasible solution.

$$\begin{aligned} \textbf{maximize} \quad & 2x_1 - 5x_2 \\ \textbf{subject to} \quad & -4x_2 \leq -1 \\ & -2x_1 - 1x_2 \leq -2 \\ & +2x_1 + 3x_2 \leq -4 \\ & x_1 \;,\; x_2 \geq 0 \end{aligned} \implies$$

		x_0	x_1	x_2
	0	−1	0	0
	0	0	2	−5
c_1	−1	−1	0	−4
c_2	−2	−1	−2	−1
c_3	−4	−1	2	3

$\implies$

		c_3	x_1	x_2
	4	−1	−2	−3
	·	·	·	·
c_1	·	·	·	·
c_2	·	·	·	·
x_0	·	·	·	·

2. $$\begin{aligned} \textbf{maximize} \quad & 1x_1 - 2x_2 \\ \textbf{subject to} \quad & 2x_1 - 1x_2 \geq -4 \\ & -1x_1 + 4x_2 \leq 8 \\ & 1x_1 + 1x_2 \leq 0 \\ & x_1 \geq 0 \end{aligned}$$

This problem is unbounded. See the x_2'' column.

$$\begin{aligned} \textbf{maximize} \quad & 1x_1 - 2x_2' + 2x_2'' \\ \textbf{subject to} \quad & -2x_1 + 1x_2' - 1x_2'' \leq 4 \\ & -1x_1 + 4x_2' - 4x_2'' \leq 8 \\ & 1x_1 + 1x_2' - 1x_2'' \leq 0 \\ & x_1 \;,\; x_2' \;,\; x_2'' \geq 0 \end{aligned} \implies$$

		x_1	x_2'	x_2''
	0	1	−2	2
c_1	4	−2	1	−1
c_2	8	−1	4	−4
c_3	0	1	1	−1

3. $$\begin{aligned} \textbf{maximize} \quad & 3x_1 + 2x_2 + 1x_3 \\ \textbf{subject to} \quad & 1x_2 - 1x_3 = 0 \\ & 1x_1 + 2x_2 + 3x_3 \leq 5 \\ & x_1 \;,\; x_2 \;,\; x_3 \geq 0 \end{aligned}$$

This problem has the optimal solution $(x_1, x_2, x_3) = (5, 0, 0)$ with $t(5, 0, 0) = 15$.

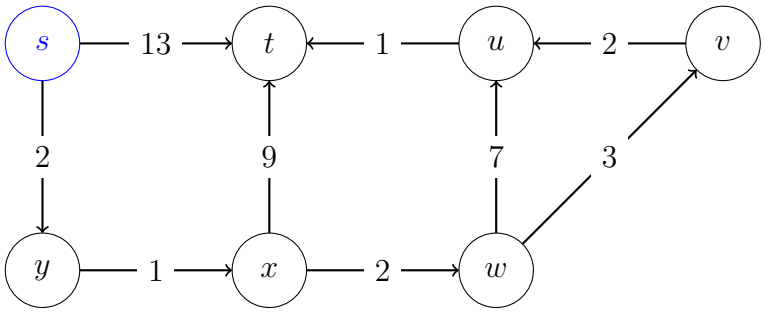
$$\begin{aligned} \textbf{maximize} \quad & 3x_1 + 2x_2 + 1x_3 \\ \textbf{subject to} \quad & 1x_2 - 1x_3 \leq 0 \\ & -1x_2 + 1x_3 \leq 0 \\ & 1x_1 + 2x_2 + 3x_3 \leq 5 \\ & x_1 \;,\; x_2 \;,\; x_3 \geq 0 \end{aligned} \implies$$

		x_1	x_2	x_3
	0	3	2	1
c_1	0	0	1	−1
c_2	0	0	−1	1
c_3	5	1	2	3

$\implies$

		c_3	x_2	x_3
	−15	−3	−4	−8
c_1	·	·	·	·
c_2	·	·	·	·
x_1	5	·	·	·

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



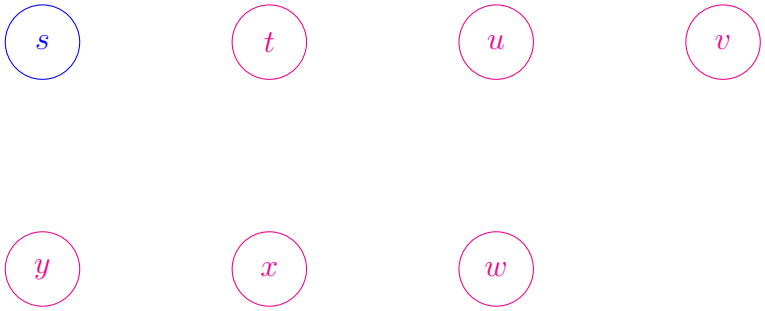
Draw the graph $G_\pi = (V_\pi, E_\pi)$ including the values $v.d$ for all $v \in V_\pi$ before iteration 1 and after each iteration i, $0 < i < 7$, during the first phase of `BellmanFord`(G, ω, s) under the condition that in each iteration the edges are processed in lexicographic order, i.e.

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

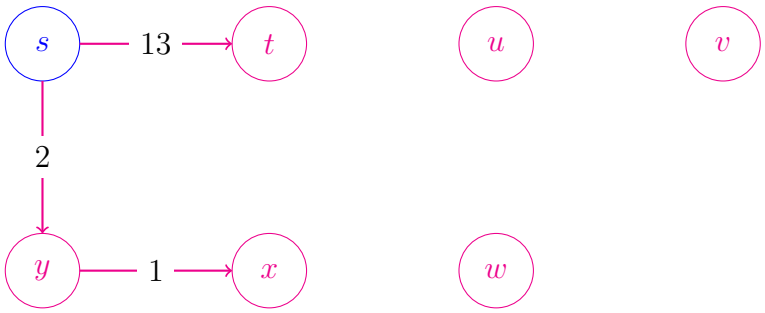
You may use the table below if it helps you.

	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	0	–												
t	∞	–	13	s	12	x					11	u		
u	∞	–					12	w	10	v				
v	∞	–					8	w						
w	∞	–			5	x								
x	∞	–	3	y										
y	∞	–	2	s										

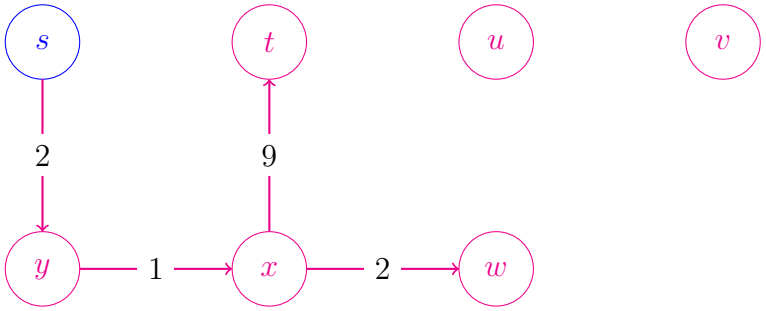
Init



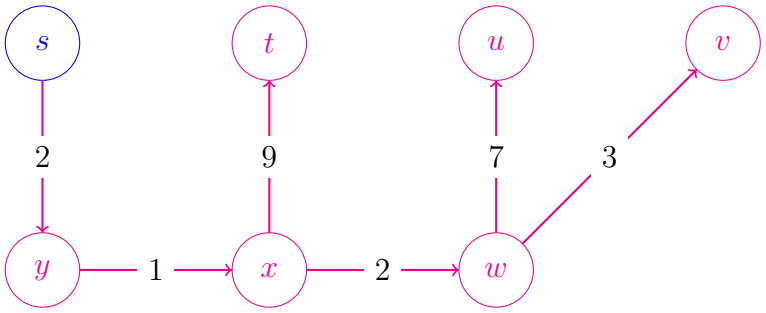
R_1^*



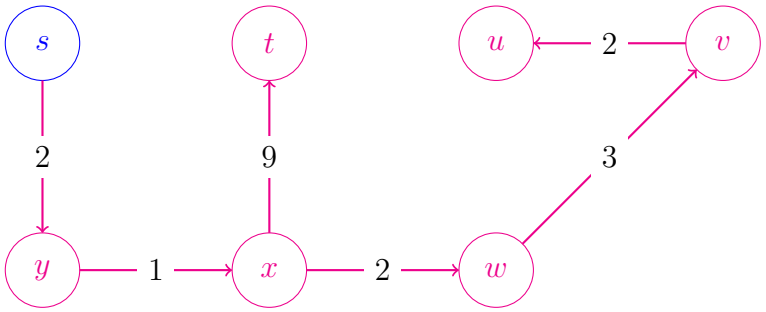
R_2^*


s
t
u
v
2
9
y
1
x
2
w

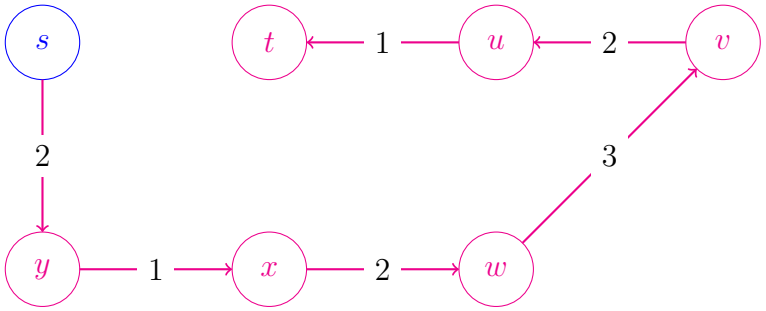
R_3^*


s
t
u
v
2
9
7
3
y
1
x
2
w

R_4^*


s
t
u
2
v
2
9
3
y
1
x
2
w

R_5^*, R_6^*


s
t
1
u
2
v
2
3
y
1
x
2
w

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:

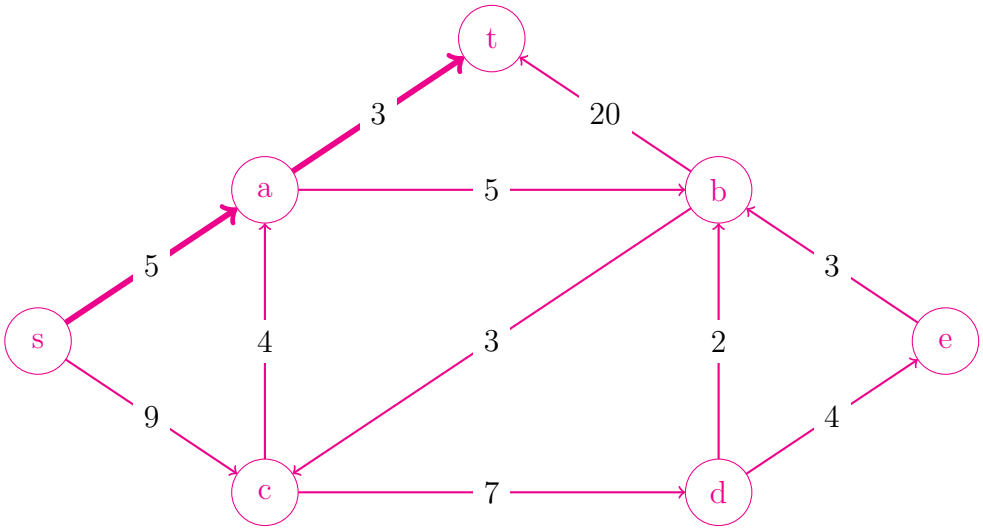
E	(s,a)	(s,c)	(a,b)	(a,t)	(b,c)	(b,t)	(c,a)	(c,d)	(d,b)	(d,e)	(e,b)
$c(e)$	5	9	5	3	3	20	4	7	2	4	3

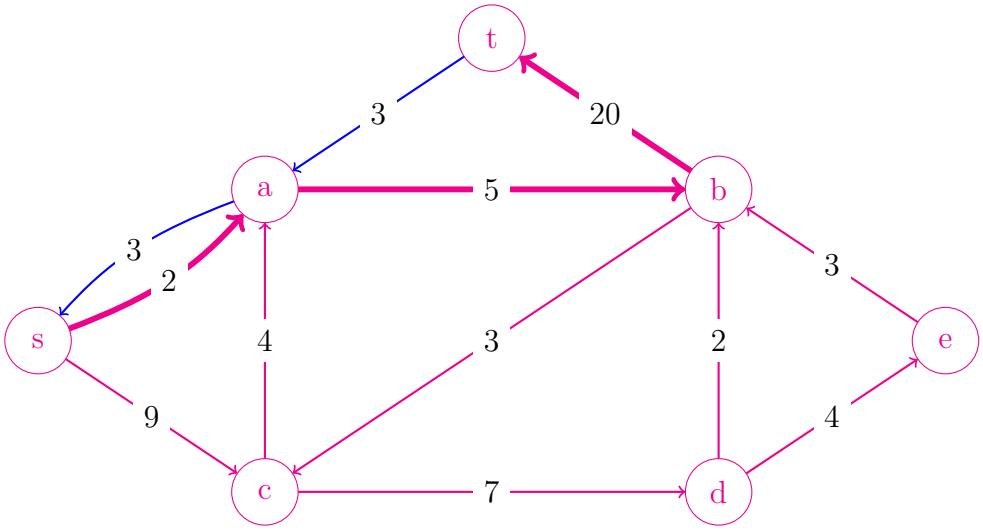
a) Perform the Edmonds-Karp algorithm on the flow network G and draw the residual network after every iteration.

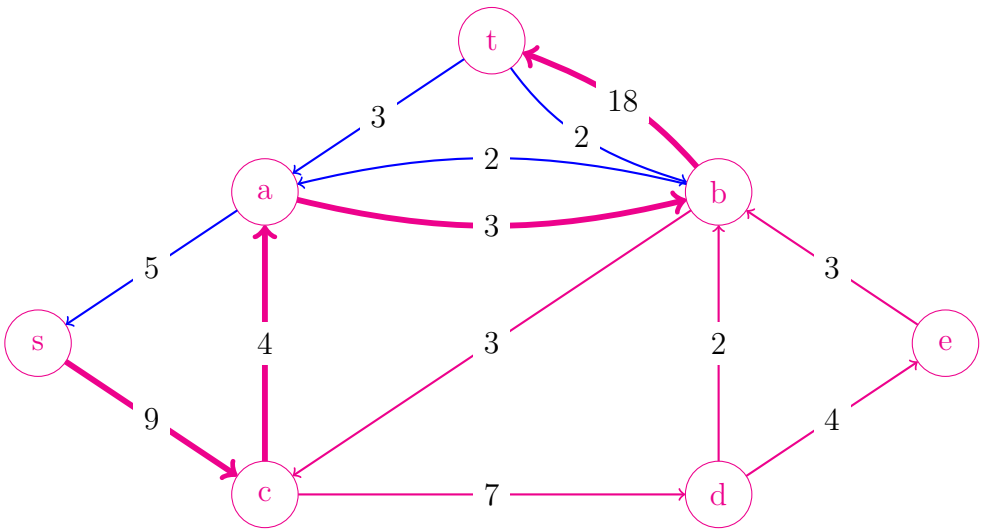
See the following two pages.

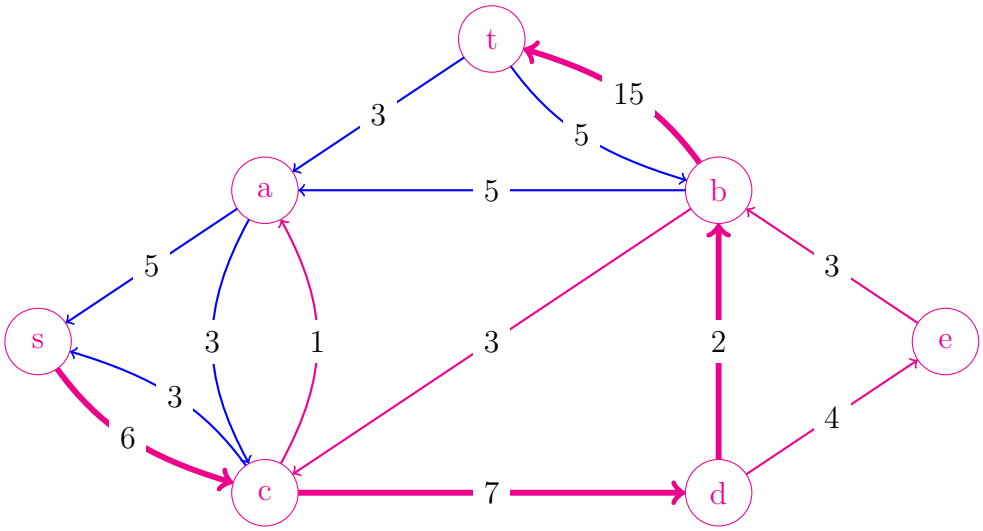
b) Find a minimal cut in the flow network depicted in exercise part (a). **Prove your claim!**

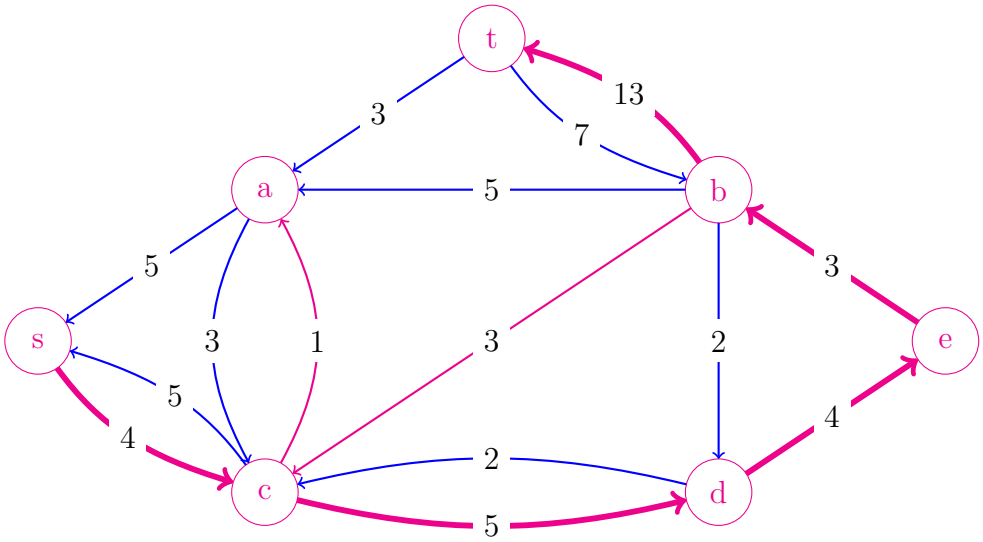
$\{\{s, a, c, d, e\}, \{b, t\}\}$ is a minimal cut as its value is the same as the maximum flow.

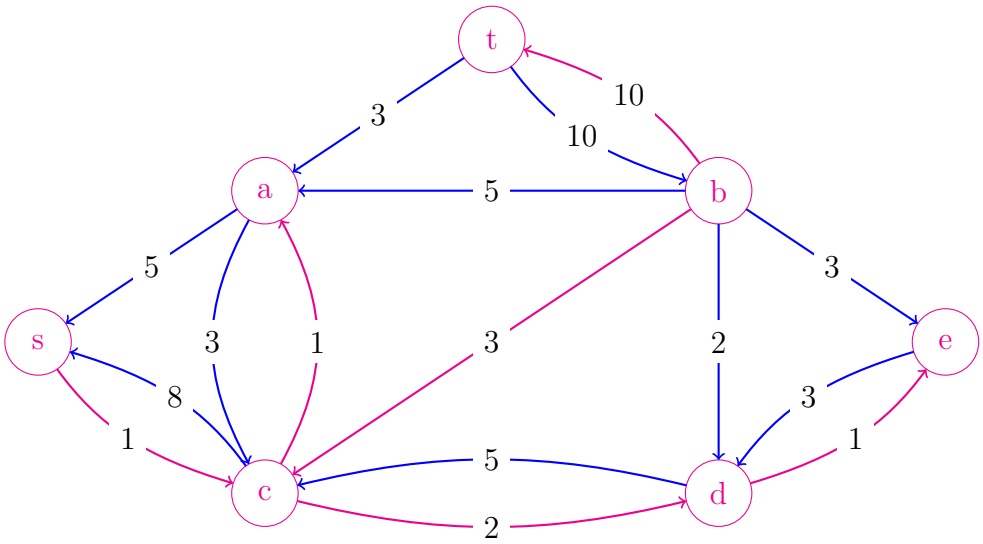

t
3
20
a
5
b
5
3
s
4
3
2
e
9
4
c
7
d


t
3
20
a
5
b
3
2
3
s
4
3
2
e
9
4
c
7
d


t
3
18
2
2
a
b
3
5
3
s
4
3
2
e
9
4
c
7
d







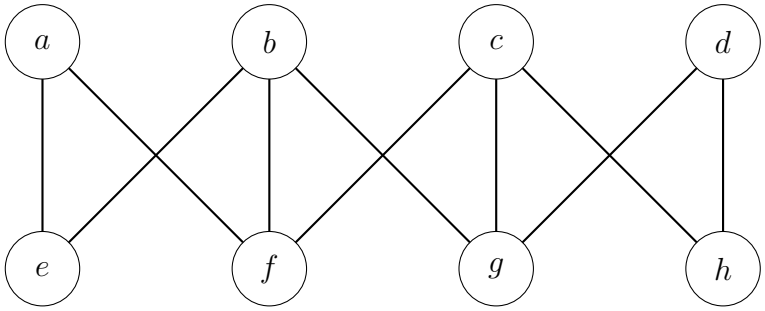
The maximum flow is 13.

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

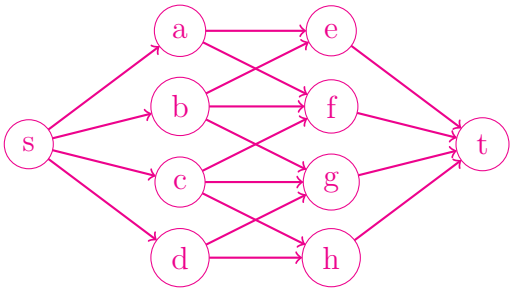
$$V = \{v_1, \ldots, v_n\},$$
$$W = \{w_1, \ldots, w_n\},$$
$$E = \{\{v_i, w_j\} \mid i = 1, \ldots, n,\ j = 1, \ldots, n\}.$$

How many different maximal matchings does $K_{n,n}$ have?
A maximal matching has the size n. The amount of maximal matchings is equal to the amount of permutations over a set with n elements: $n!$.

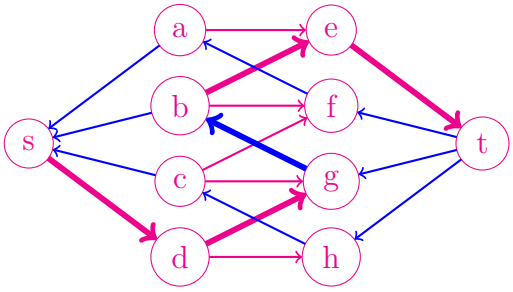
Exercise 1.6: (2 - 3 minutes per subtask)
Consider the following bipartite graph $G = (V, W, E)$ with $V = \{a, b, c, d\}$, $W = \{e, d, f, g\}$ and the edges corresponding to the illustration:



a) Draw the flow network corresponding to G.[1]



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.[1]



[1] Each edge in the graph has a capacity of 1.

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.

From the lecture we know a polynomial reduction from 3-SAT to Clique. See Theorem 76. From exercise 4.3 we know a polynomial reduction from Clique to Vertex Cover. First we perform the reduction from 3-SAT to Clique and then we perform the reduction from Clique to Vertex Cover.

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 $\sum_{i \in I} a_i = n$.

First, we remove all a_i with $a_i > n$. This can be done in linear time. Then we reduce the problem to *MaxKP* where $W = n$ and $c_i = w_i = a_i$. Using the DynamicKP algorithm, we can solve the problem in polynomial time as $c_{\max} = n$. If DynamicKP outputs $c_{opt} = n$, we output TRUE.

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.

Please refer to the lecture script.

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.

  $\mathcal{PTIME}$, $\mathcal{P}$ and $\mathcal{NP}$. Using depth-first search.

- Π_2: **Input:** An undirected bipartite graph $G = (V, W, E)$. **Output:** A maximal matching in G.

  $\mathcal{PTIME}$. Using Ford-Fulkerson.

- 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.

  $\mathcal{PTIME}$, $\mathcal{P}$ and $\mathcal{NP}$. See Exercise 4.4.

- 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 $|V'| = \mathrm{m}$. FALSE otherwise.

  $\mathcal{NP}$, $\mathcal{NPC}$ and $\mathcal{NP}$-hard. Reduction from 3-SAT.

– 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.

$\mathcal{NP}$, $\mathcal{NPC}$ and $\mathcal{NP}$-hard. 3-SAT is a special case.

– 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.

$\mathcal{NP}$, $\mathcal{NPC}$ and $\mathcal{NP}$-hard. Special Knapsack Problem.

– 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 not satisfy C. FALSE otherwise.

$\mathcal{PTIME}$, $\mathcal{P}$ and $\mathcal{NP}$. See Exercise 4.4.

– 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 not satisfy M. FALSE otherwise.

$\mathcal{NP}$, $\mathcal{NPC}$ and $\mathcal{NP}$-hard. See Exercise 4.4.

– 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.

$\mathcal{PTIME}$, $\mathcal{P}$ and $\mathcal{NP}$. Reduction to DynamicKP with $W = \frac{1}{2} \cdot \sum_{i=1}^{n} a_i$.

– 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.

Strongly $\mathcal{NP}$-hard. TSP optimization problem.

– 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.

$\mathcal{NP}$-hard. SAT optimization version.

– 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.

$\mathcal{NP}$, $\mathcal{NPC}$ and $\mathcal{NP}$-hard. Reduction from Hamiltonian Circuit. See Exercise 1.10.

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

$\mathcal{PTIME}$, $\mathcal{P}$ and $\mathcal{NP}$. Exhaustive Search over all $\binom{|V|}{2}$ possibilities.

– 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.

Strongly $\mathcal{NP}$-hard. Travelling Salesman optimization problem.

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.
Fix some node x and set $s = t = x$ and $k = n$. Ask the oracle.