

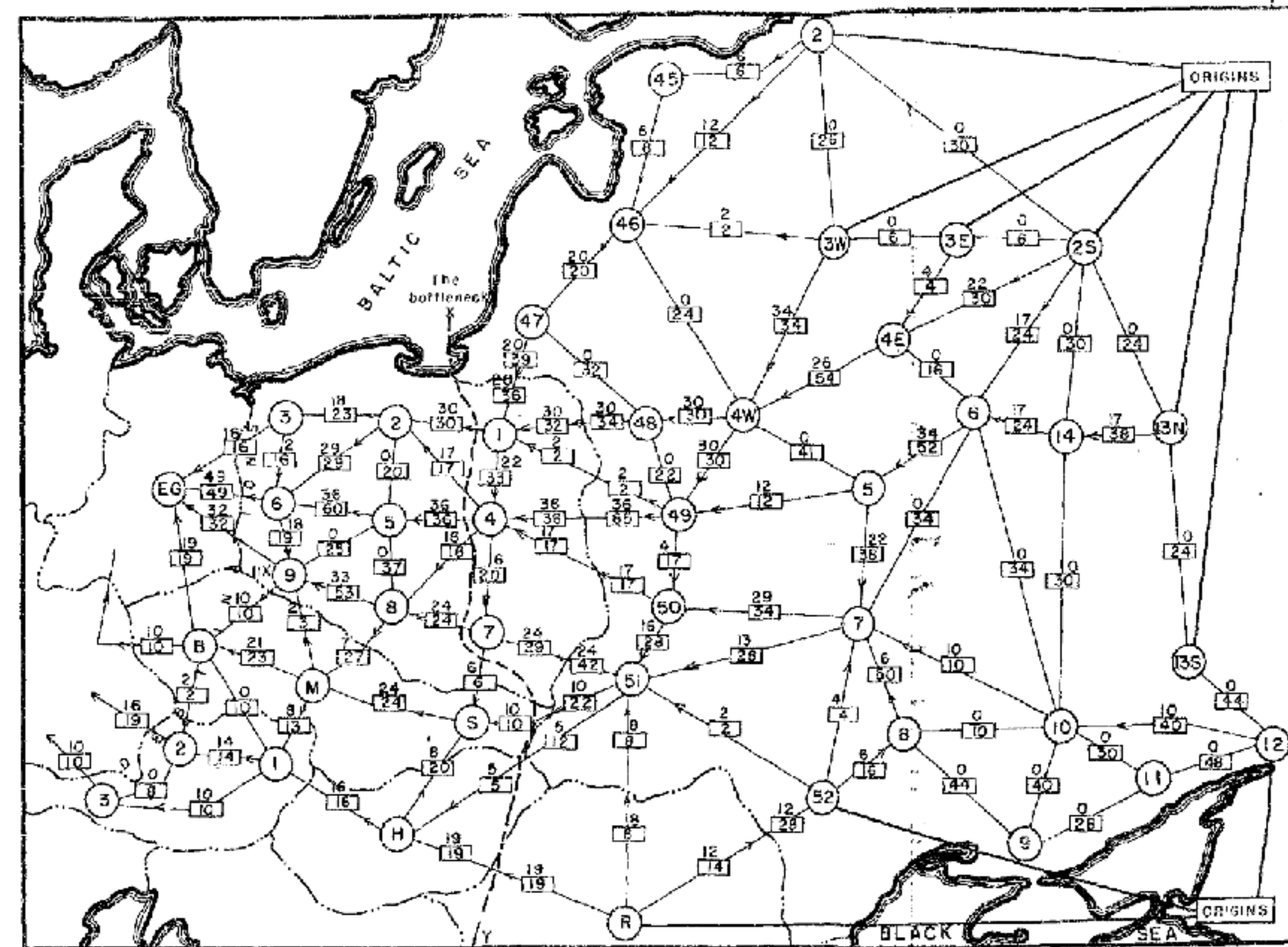


Advanced Algorithms

Network Flow & Linear Programming

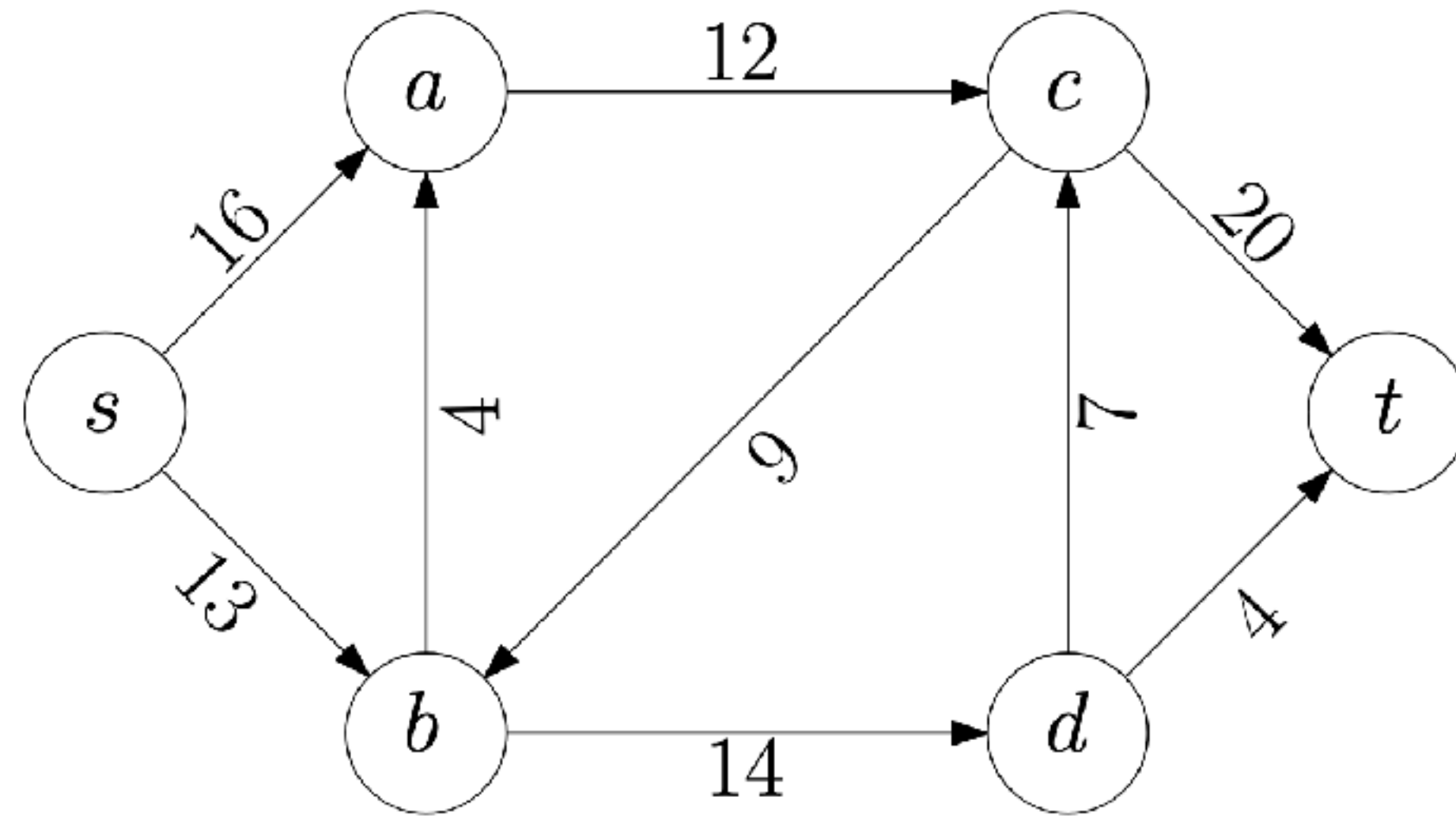
刘明谋 Nanjing University, Suzhou, 2025

Network Flow



Graph With Capacity

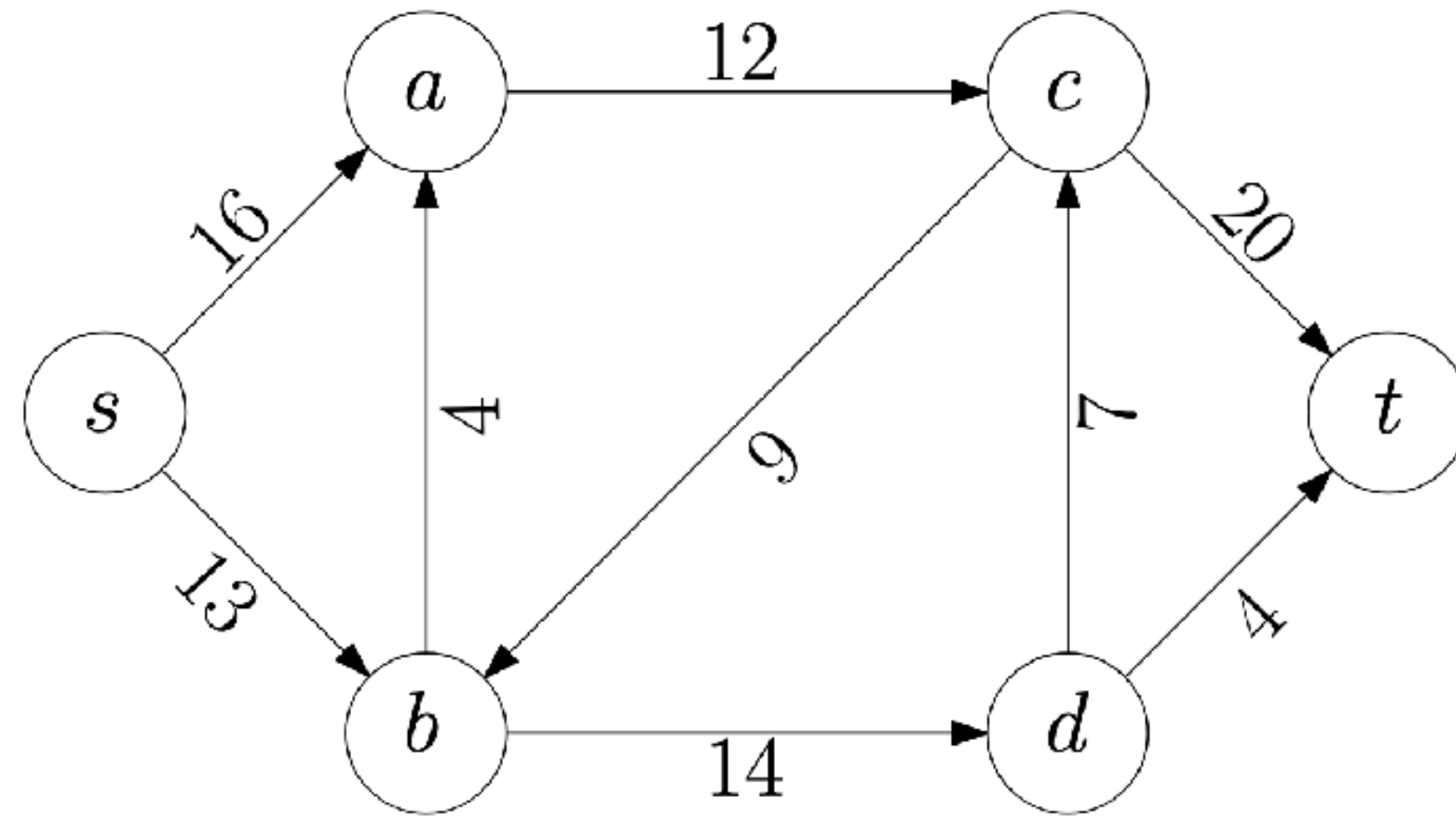
Flow



- Digraph $G = (V, E)$: a set of vertices V and a set of directed edges E
- Edge capacity $c_e \in \mathbb{R}^+$ for edge $e \in E$
- Flow function $f: E \rightarrow \mathbb{R}^+$
 - Valid flow: $\forall e \in E, 0 \leq f(e) \leq c_e$ & $\forall v \in V, \sum_{e \in \delta_{in}(v)} f(e) = \sum_{e \in \delta_{out}(v)} f(e)$

Flow Network

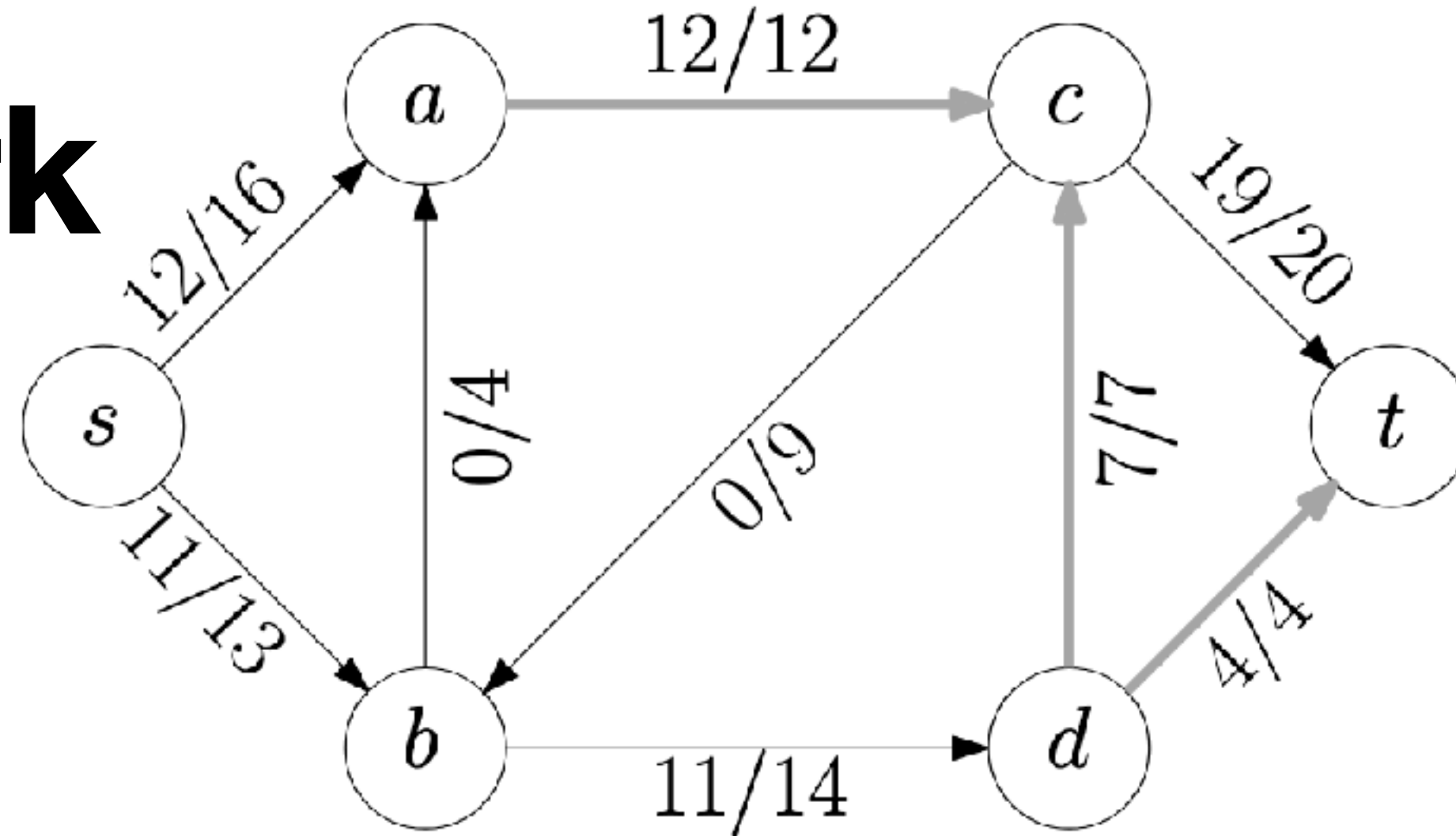
s-t flow



- Digraph $G = (V, E)$: a set of vertices V and a set of directed edges E
- Edge capacity $c_e \in \mathbb{R}^+$ for edge $e \in E$
- Flow function $f: E \rightarrow \mathbb{R}^+$, source $s \in V$, sink $t \in V$
 - Valid flow: $\forall e \in E, 0 \leq f(e) \leq c_e$ & $\forall v \in V \setminus \{s, t\}, \sum_{e \in \delta_{in}(v)} f(e) = \sum_{e \in \delta_{out}(v)} f(e)$

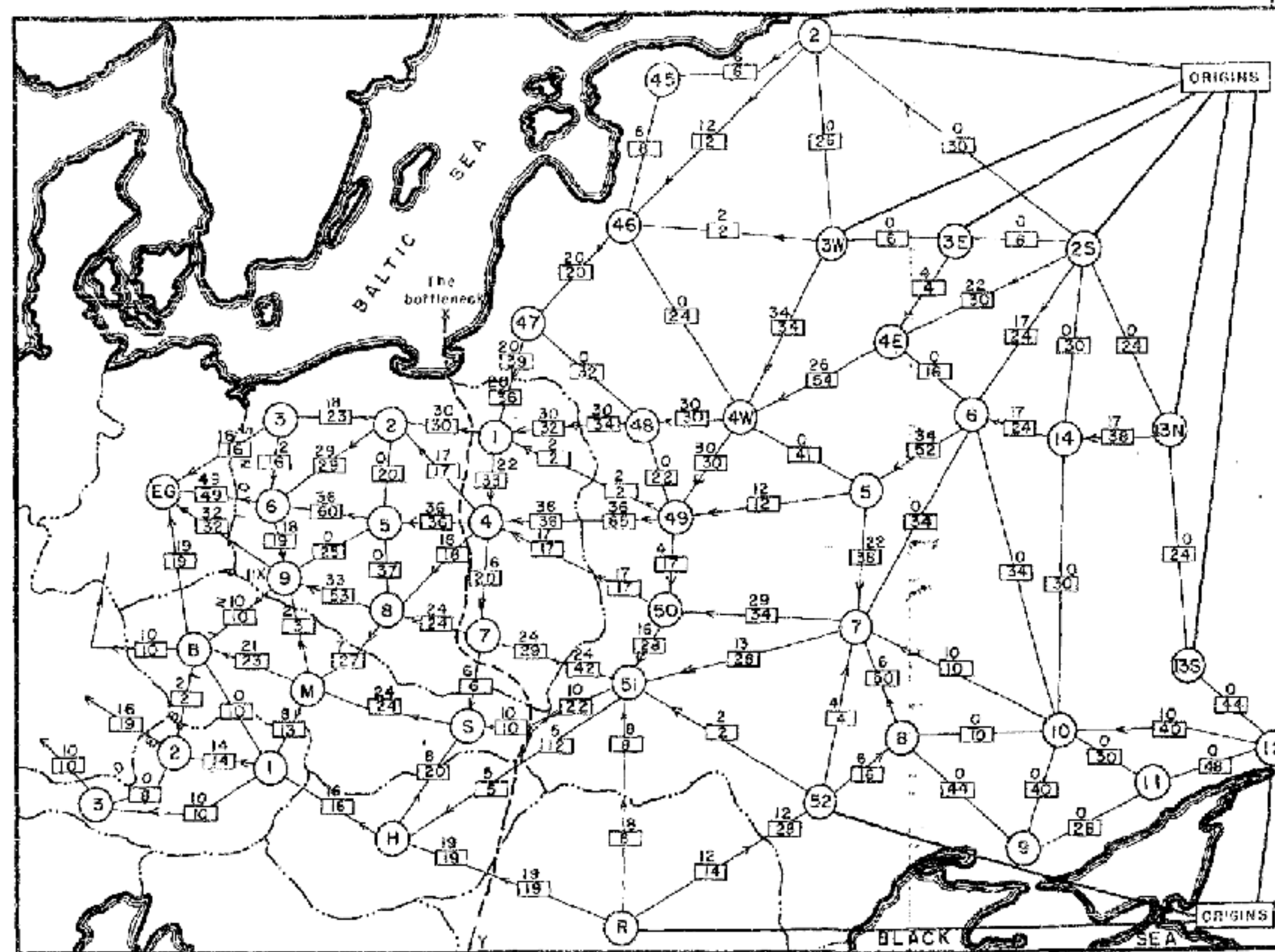
Flow Network

Maximum flow



- Digraph $G = (V, E)$: a set of vertices V and a set of directed edges E
- Edge capacity $c_e \in \mathbb{R}^+$ for edge $e \in E$
- Flow function $f: E \rightarrow \mathbb{R}^+$, source $s \in V$, sink $t \in V$
 - Valid flow: $\forall e \in E, 0 \leq f(e) \leq c_e$ & $\forall v \in V \setminus \{s, t\}, \sum_{e \in \delta_{in}(v)} f(e) = \sum_{e \in \delta_{out}(v)} f(e)$
- Maximum flow problem: find a flow f maximizes $\sum_{e \in \delta_{out}(s)} f(e)$

Army Transportation



Assumption:

Entire network available for east-west traffic (no allowance for civilian or economic traffic)

Results:

- 163,000 tons per day can be delivered from points of origin to destinations.
- 147,000 tons per day can be delivered without using Austrian lines.
- 152,000 tons per day can be delivered into Germany by all lines.
- 126,000 tons per day can be delivered into East Germany without using Austrian lines.

SECRET RM-1573
10-24-55
-33-

Fig. 7 — Traffic pattern: entire network available

Legend:

— International boundary

(B) Railway operating division

← 12 → Capacity: 12 each way per day. Required flow of 9 per day toward destinations (in direction of arrow) with equivalent number of returning trains in opposite direction

All capacities in trains (1000's of tons) each way per day

Origins: Divisions 2, 3W, 3E, 25, 13N, 13S, 12, 52 (USSR), and Roumania

Destinations: Divisions 3, 6, 8 (Poland); B (Czechoslovakia); and 2, 3 (Austria)

Alternative destinations: Germany or East Germany

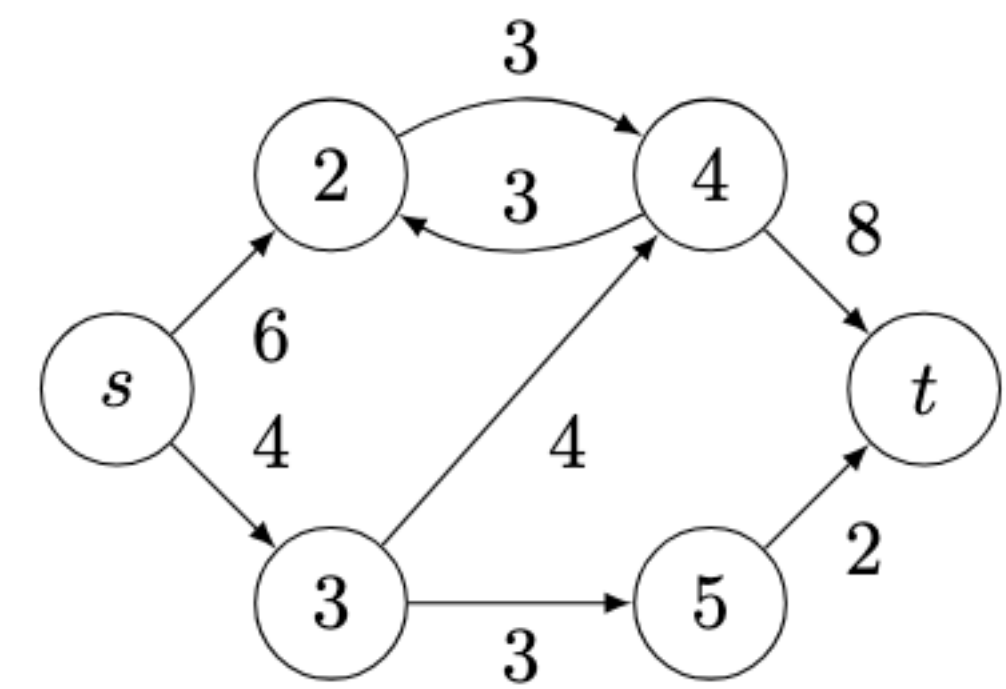
Note: IX at Division 9, Poland

SECRET

RM 1573-7

Flow Decomposition

Maximum flow consists of paths



- Theorem: Any s - t flow can be decomposed into cycles and s - t paths.
- Constructive proof:

- While s has out flow: exists valid s - t path since
$$\sum_{e \in \delta_{out}(s)} f(e) = \sum_{e \in \delta_{in}(t)} f(e)$$
 - find s - t path $P \subseteq E$; let $\delta(P) \triangleq \min_{e \in P} f_e$; update $\forall e \in P, f_e \leftarrow f_e - \delta(P)$
- While exists flow: exists flow cycles since
$$\sum_{e \in \delta_{in}(v)} f(e) = \sum_{e \in \delta_{out}(v)} f(e), \forall v$$
 - find flow cycle $P \subseteq E$, remove it as well

Augmenting Path

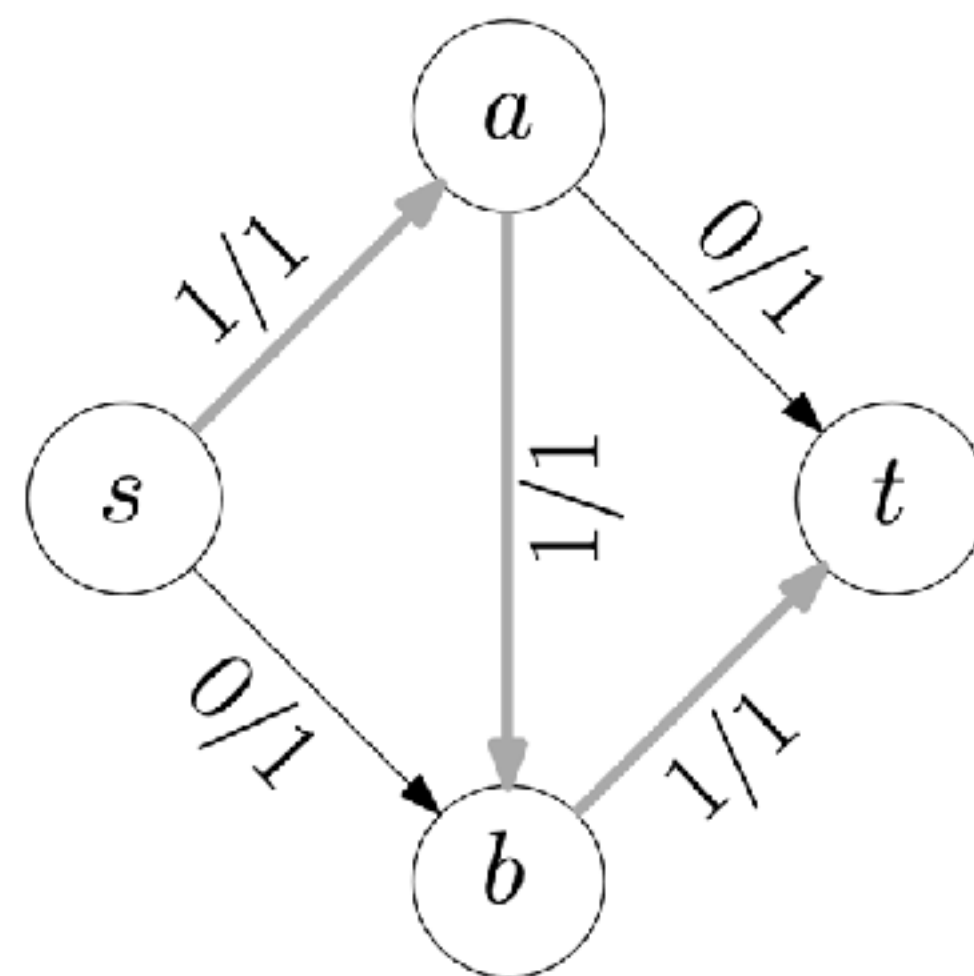
Maximum flow consists of paths

- Theorem: Any s - t flow can be decomposed into cycles and s - t paths.

Greedy: let $\delta(P) \triangleq \min_{e \in P} c_e - f_e$ be residual capacity

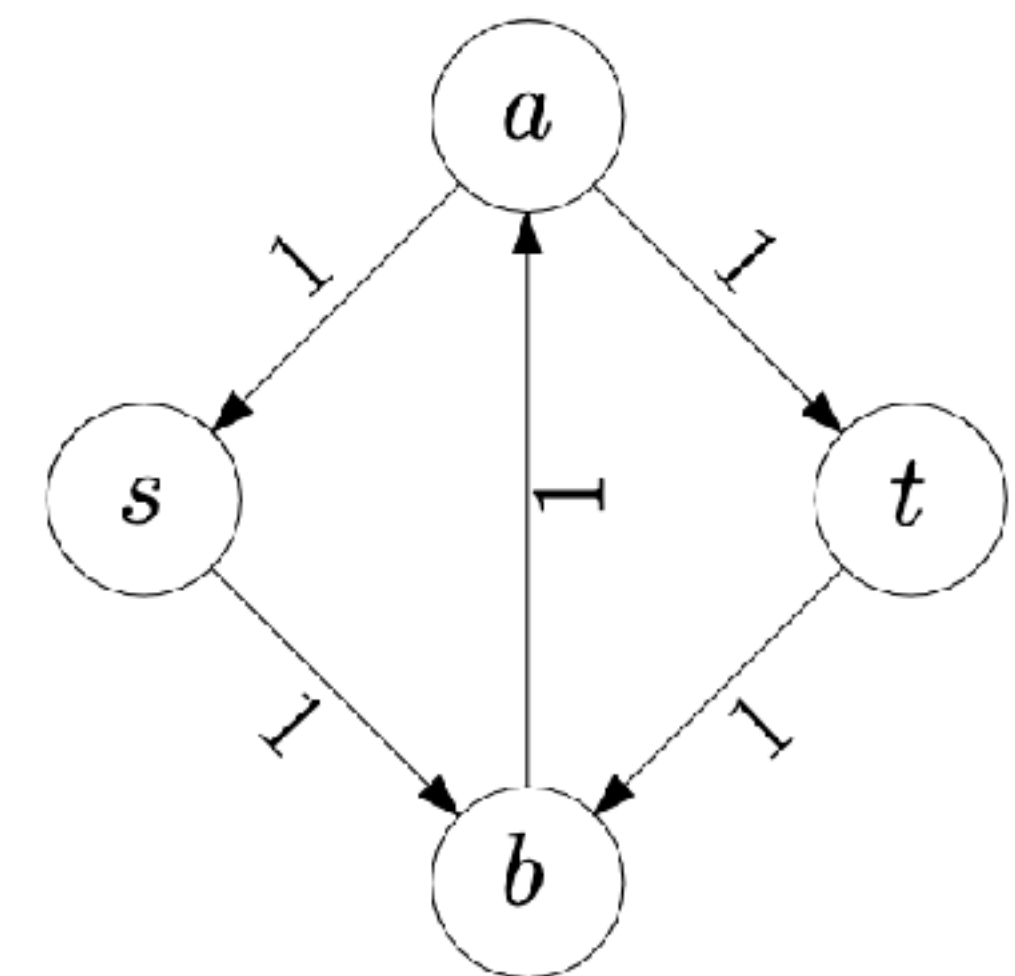
While exists s - t path P with $\delta(P) > 0$:
increase the flow in P by $\delta(P)$

- Any issue?



Observation:
implicit augmenting path

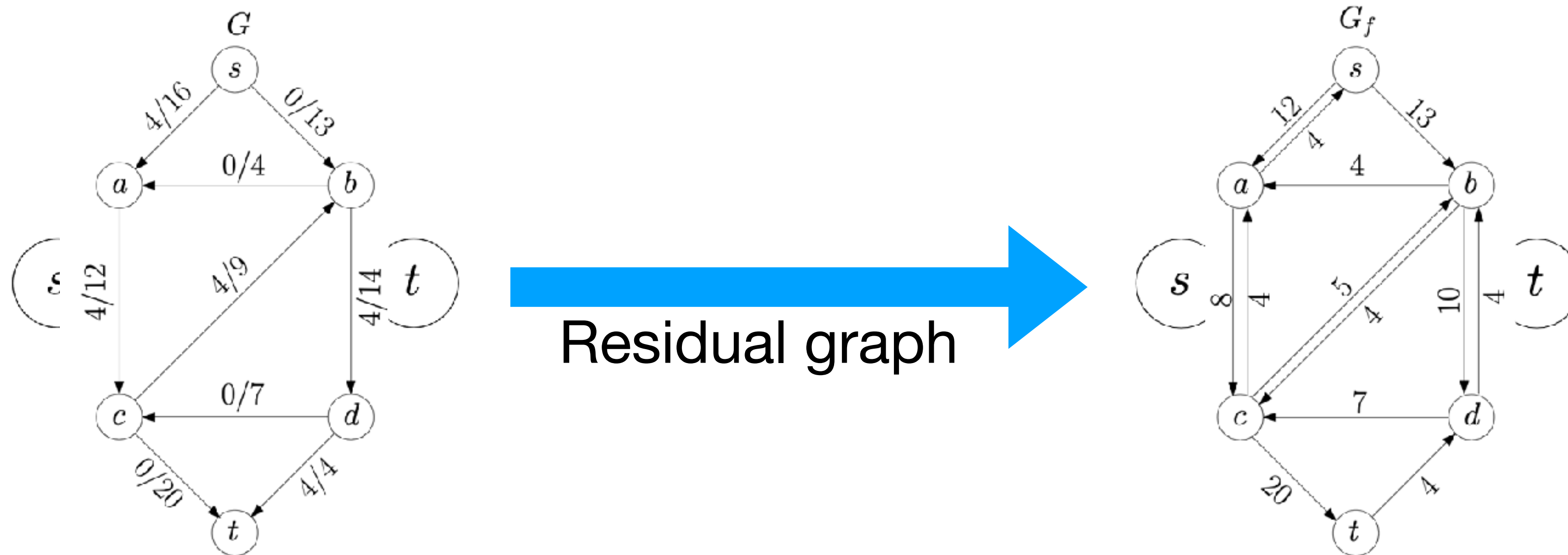
Residual graph



Augmenting Path

Fixing with residual graph

- Theorem: Any s - t flow can be decomposed into cycles and s - t paths.
- **Residual graph** $G_f = (V, E')$ of $G = (V, E)$ with capacity c and flow f :
 - for each $e = (u, v) \in E$ with $c_e > f_e$, add e to E' with $c'_e = c_e - f_e$
 - for each $e = (u, v) \in E$ with $f_e > 0$, add (v, u) to E' with $c'_{e'} = f_e$

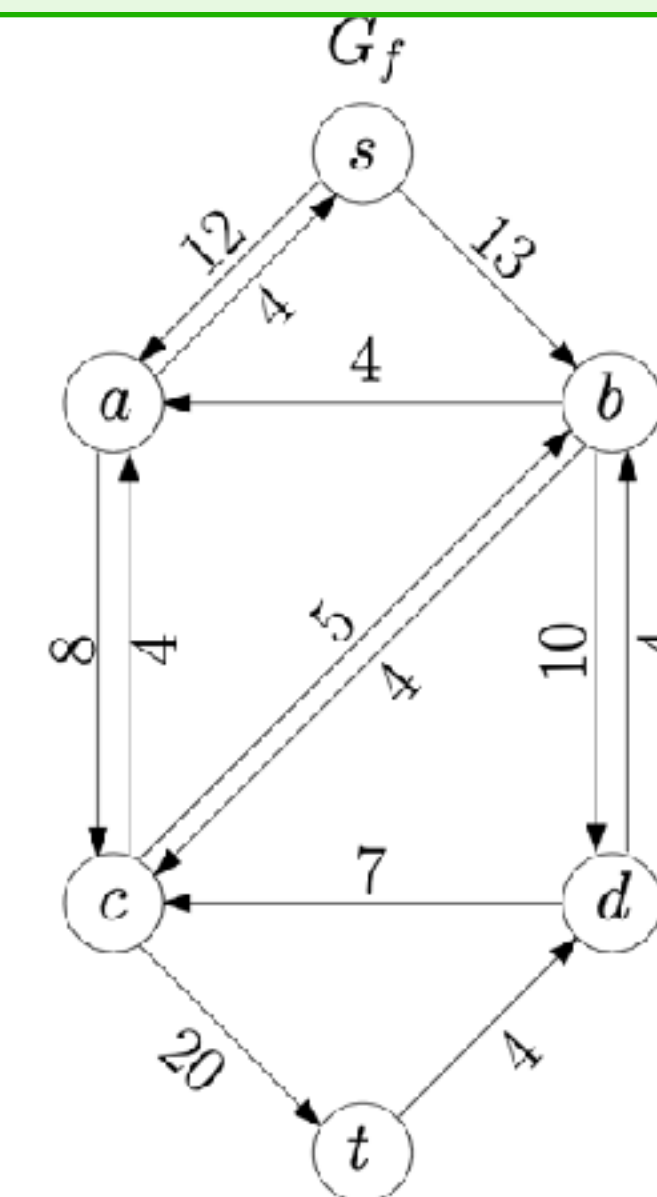
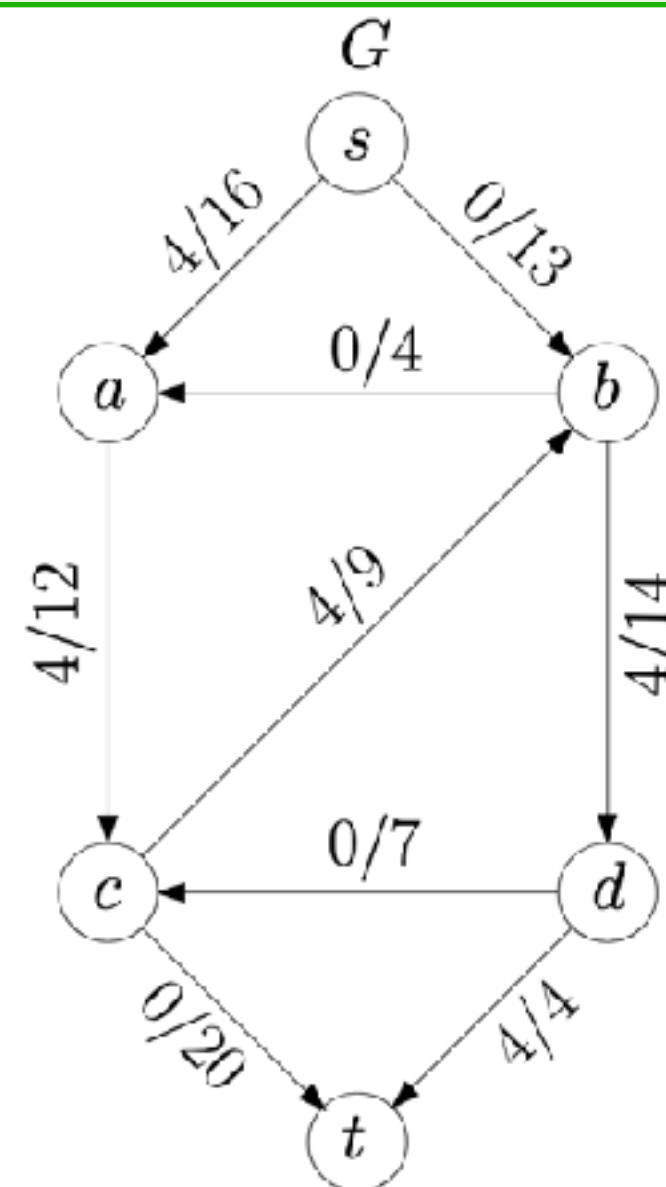


Augmenting Path

Fixing with residual graph

- Theorem: Any s - t flow can be decomposed into cycles and s - t paths.

Ford-Fulkerson: let $\delta(P)$ be residual capacity for path P
While exists s - t path $P \in G^f$ with $\delta(P) > 0$:
increase the flow f in P by $\delta(P)$



Ford-Fulkerson: let $\delta(P)$ be residual capacity for path P
While exists s - t path $P \in G^f$ with $\delta(P) > 0$:
increase the flow f in P by $\delta(P)$

The following statements are equivalent:

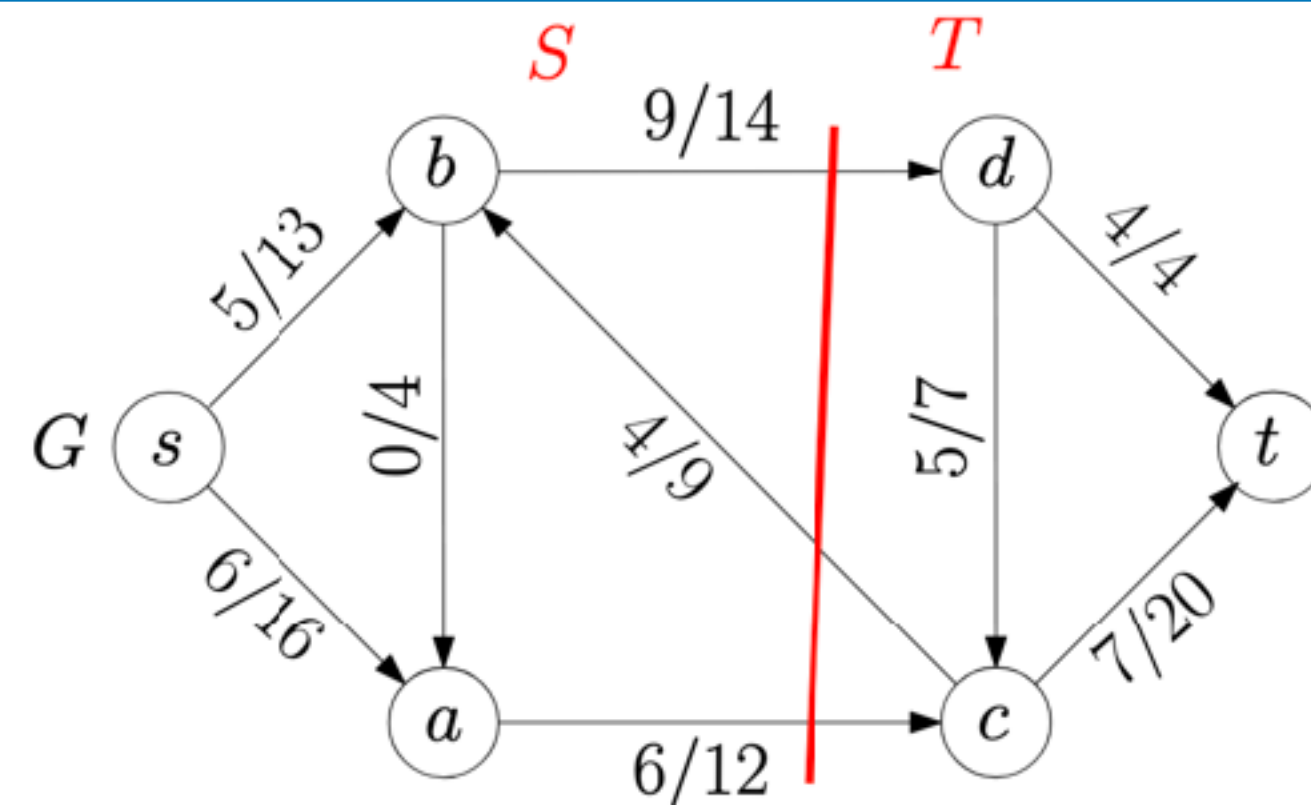
- (1) f is a maximum flow
- (2) There is no s - t path P in G^f with $\delta(P) > 0$
- (3) There is $S, \neg S \subseteq V$ such that $\text{cut}(S) = |f|$

Proof strategy:

(1) \implies (2): proven by $\neg(2)\implies\neg(1)$

(2) \implies (3): every edge is stuck

(3) \implies (1): every flow is at most any cut



The following statements are equivalent:

- (1)** f is a maximum flow
- (2)** There is no s - t path P in G^f with $\delta(P) > 0$
- (3)** There is $S, \neg S \subseteq V$ such that $\text{Cut}(S) = |f|$

(2) \Rightarrow (3): Let S be set of vertices reachable in G^f from s

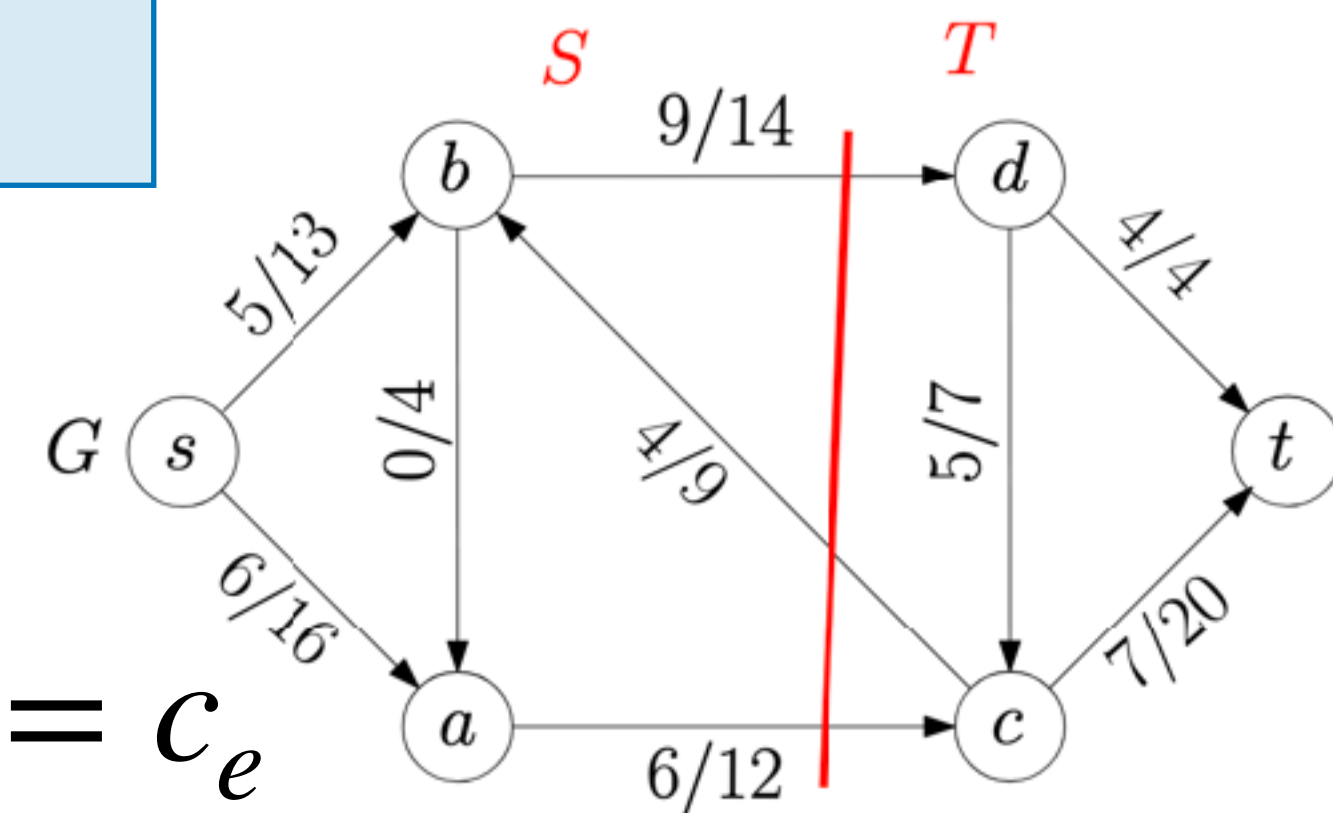
- Observation:

- Every edge $e = (a, b) \in E$ with $a \in S, b \in \neg S$, then $f_e = c_e$
- Every edge $e = (b, a) \in E$ with $a \in S, b \in \neg S$, then $f_e = 0$

- Otherwise b is reachable in G^f as well.

- By flow conservation:

$$|f| = \sum_{a \in S, b \notin S} f_{a,b} - \sum_{a \in S, b \notin S} f_{b,a} = \sum_{\substack{(a,b) \in E \\ a \in S, b \notin S}} c_{a,b} = \text{Cut}(S)$$

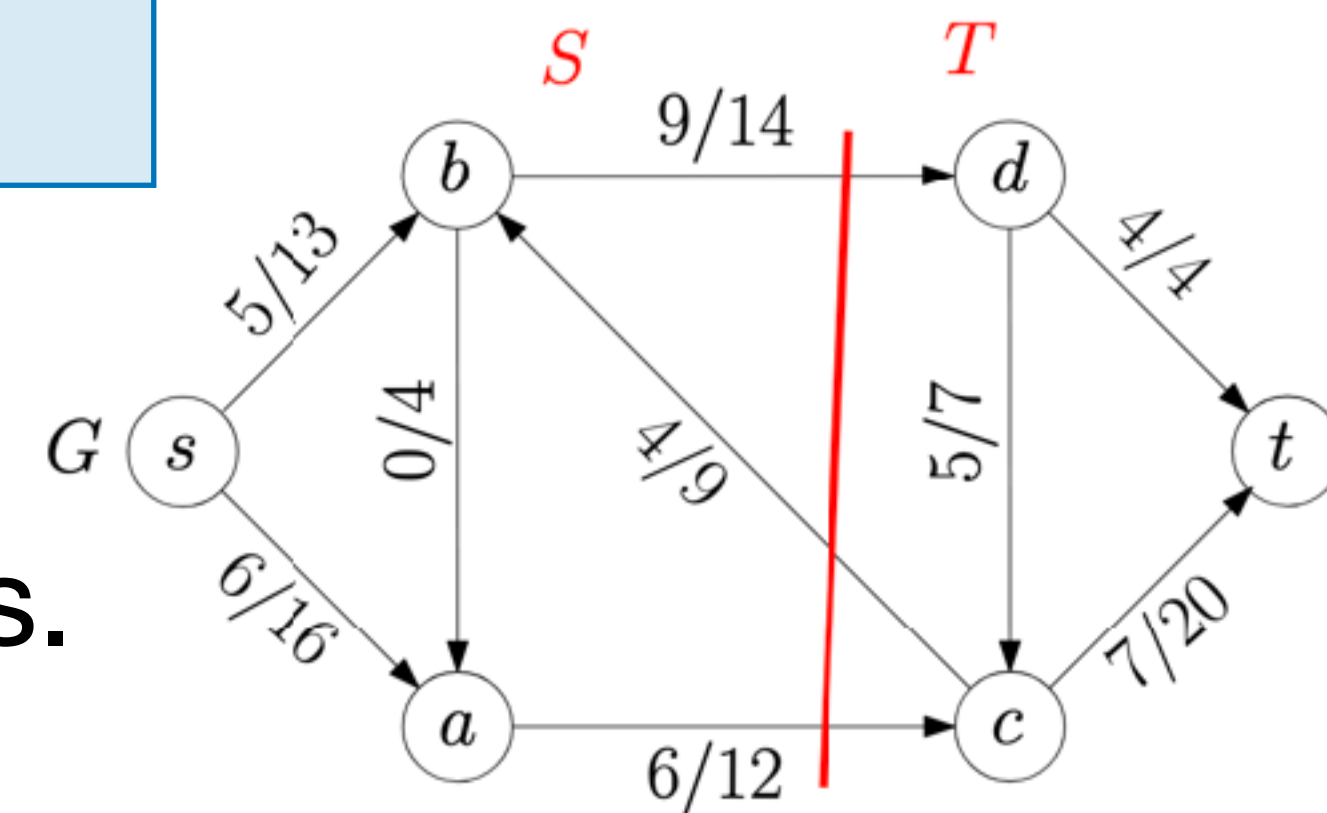


The following statements are equivalent:

- (1)** f is a maximum flow
- (2)** There is no s - t path P in G^f with $\delta(P) > 0$
- (3)** There is $S, \neg S \subseteq V$ such that $\text{Cut}(S) = |f|$

(3) \Rightarrow (1): $|f| \leq \text{Cut}(S)$ for any f, S

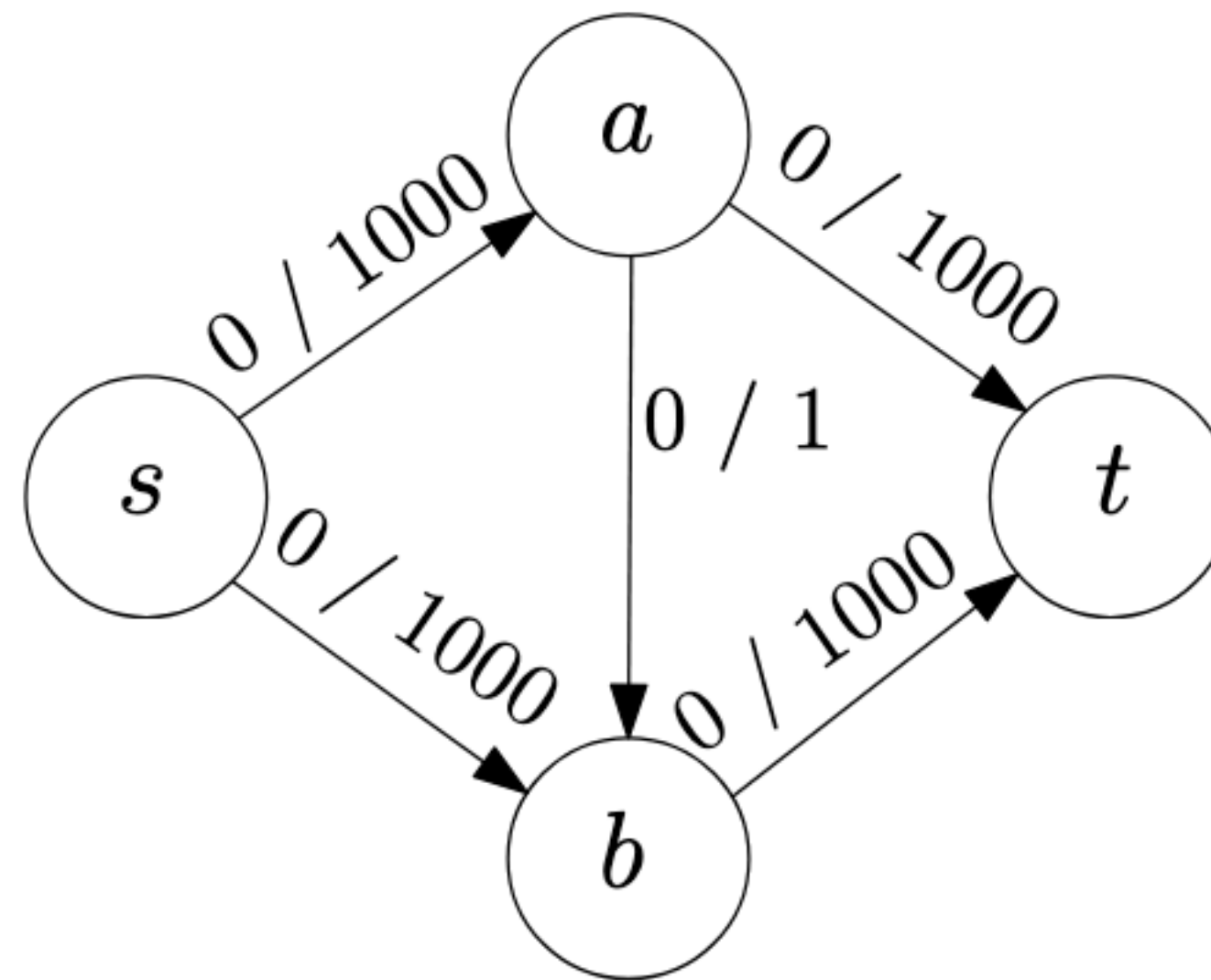
- Proof by decomposition: decompose f , keep only s - t paths.
- Consider a path P with p flow.
 - Pass from S to $\neg S$
 - Uses $\geq p$ capacity of $\text{Cut}(S)$ (backward is allowed)
- $|f| = \sum_P p \leq \text{Cut}(S)$



Augmenting Path

Ford-Fulkerson: let $\delta(P)$ be residual capacity for path P
While exists s - t path $P \in G^f$ with $\delta(P) > 0$:
increase the flow f in P by $\delta(P)$

- Time cost? Worst case.

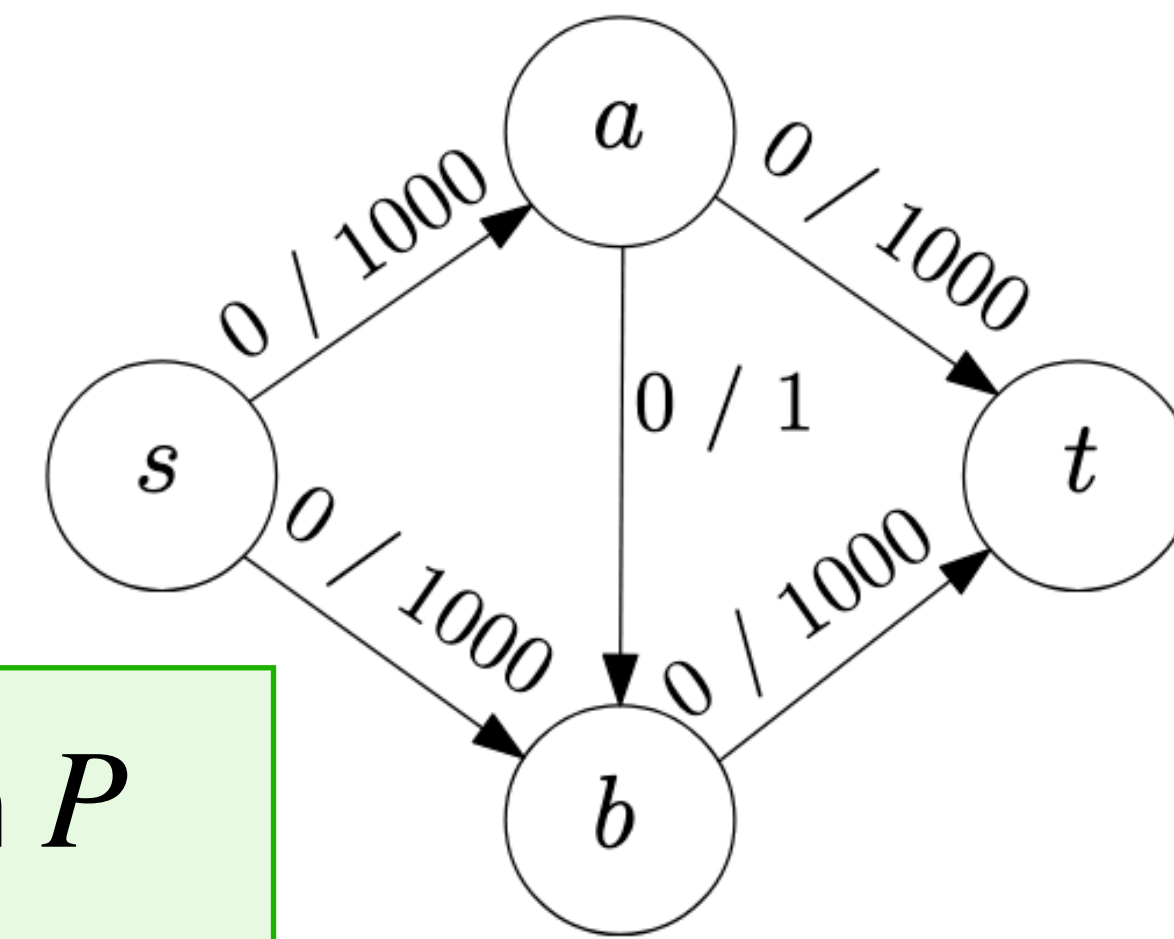


- Naive Ford-Fulkerson algorithms takes $O(|f^*| \cdot m)$ time

Augmenting Path

Greedy

Ford-Fulkerson: let $\delta(P)$ be residual capacity for path P
While exists s - t path $P \in G^f$ with $\delta(P) > 0$:
increase the flow f in P by $\delta(P)$



- Theorem: Any s - t flow can be decomposed into cycles and s - t paths.
- Greedy idea: adopt the “fattest” augmenting path
- Max flow $|f^*| = |f| + |f'|$, where f' is the max flow in G^f
- By decomposition, exists a path P with flow $\geq |f'|/m$, since $\leq m$ paths
- Convergence: $|f'| \leq |f^*|(1 - 1/m)^t$ after t iterations.
 - #iterations = $O(m \log |f^*|)$. Might not converge for real capacity.

Augmenting Path

Greedy

Observation:
augmenting fattest path
results in **thinner** path

Ford-Fulkerson: let $\delta(P)$ be residual capacity for path P
While exists s - t path $P \in G^f$ with $\delta(P) > 0$:
increase the flow f in P by $\delta(P)$

- Greedy idea: adopt the “fattest” augmenting path
- Max flow $|f^*| = |f| + |f'|$, where f' is the max flow in G^f
- By decomposition, exists a path P with flow $\geq |f'|/m$, since $\leq m$ paths
- Convergence: $|f'| \leq |f^*| (1 - 1/m)^t$ after t iterations.
 - #iterations = $O(m \log |f^*|)$. Might not converge for real capacity.
- Finding “fat” paths: binary search + BFS in $O(m \log |f^*|)$ time

Augmenting Path

Greedy + Scaling

Observation:
augmenting shortest path
results in **longer** path

Ford-Fulkerson: let $\delta(P)$ be residual capacity for path P
While exists s - t path $P \in G^f$ with $\delta(P) > 0$:
increase the flow f in P by $\delta(P)$

- Greedy idea: for $i = \lceil \log_2 C \rceil, \dots, 0$: while exists flow $|f| \geq 2^i$: augment
- Imagine residual graph $G_i^f = (V, E'', f)$ with $E'' \triangleq \{e \in E' : c_e \geq 2^i\}$
 - Any edge e is of capacity $c_e \in [2^i, 2^{i-1})$
 - All augmenting paths contribute $\geq 2^i$
 - #flows in G_i^f is $\leq m$ by decomposition
- Inner loop repeats $O(m)$ times. Time cost $O(\log_2 C \cdot m \cdot m)$

Augmenting Path

Shortest augmenting [Dinitz, Edmonds-Karp]

Observation:
augmenting shortest path
results in **longer** path

Ford-Fulkerson: let $\delta(P)$ be residual capacity for path P

While exists s - t path $P \in G^f$ with $\delta(P) > 0$:

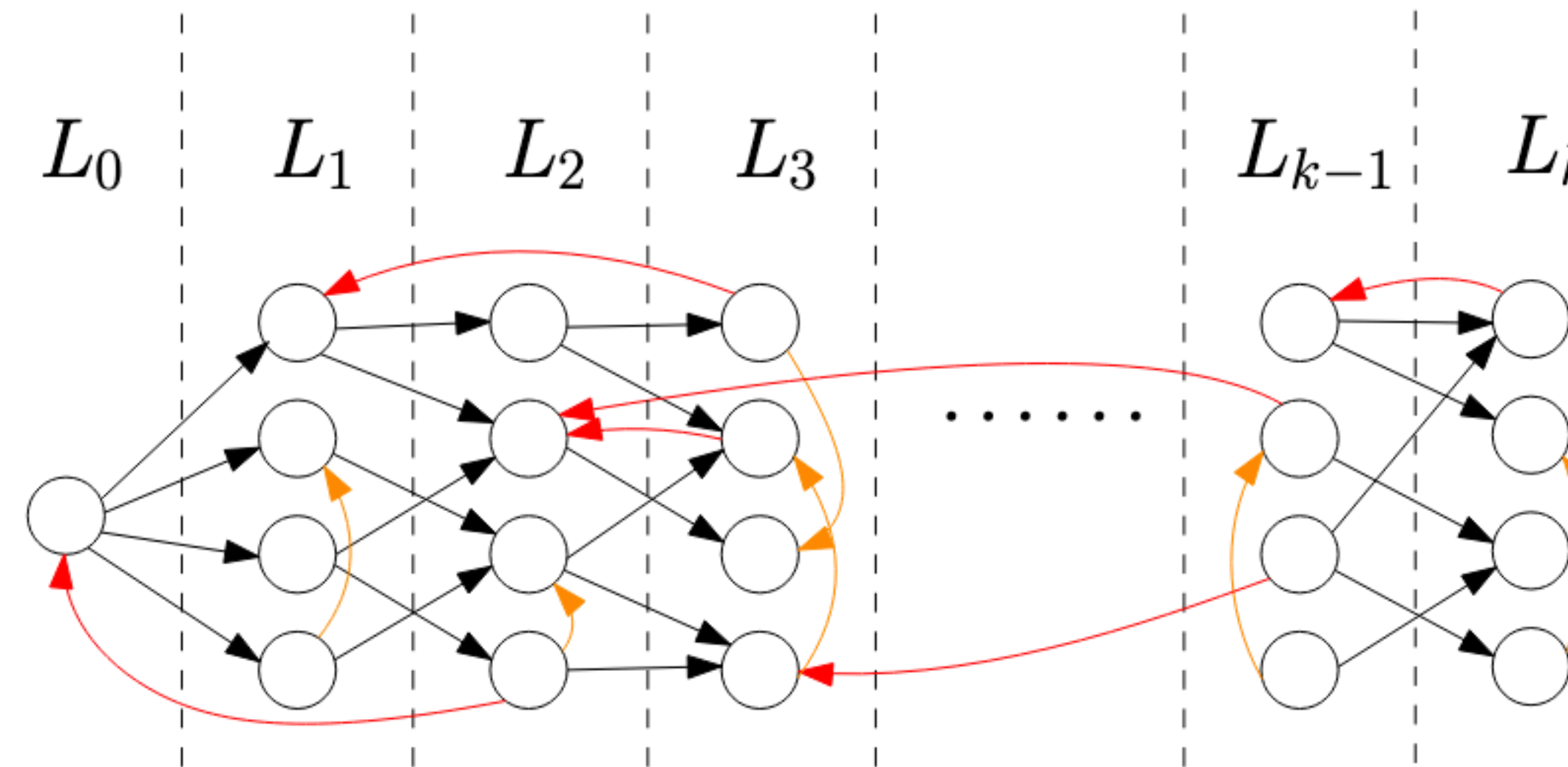
increase the flow f in P by $\delta(P)$

- Algorithm: keep finding shortest path, then augment
- Time cost?
 - Path finding: $O(m)$ time.
 - Path lengths: $1, \dots, n$.
 - #paths for each length: $\leq m$
- $O(m^2n)$ time in total

Augmenting Path

Shortest augmenting [Dinitz, Edmonds-Karp]

Observation:
augmenting shortest path
results in **longer** path

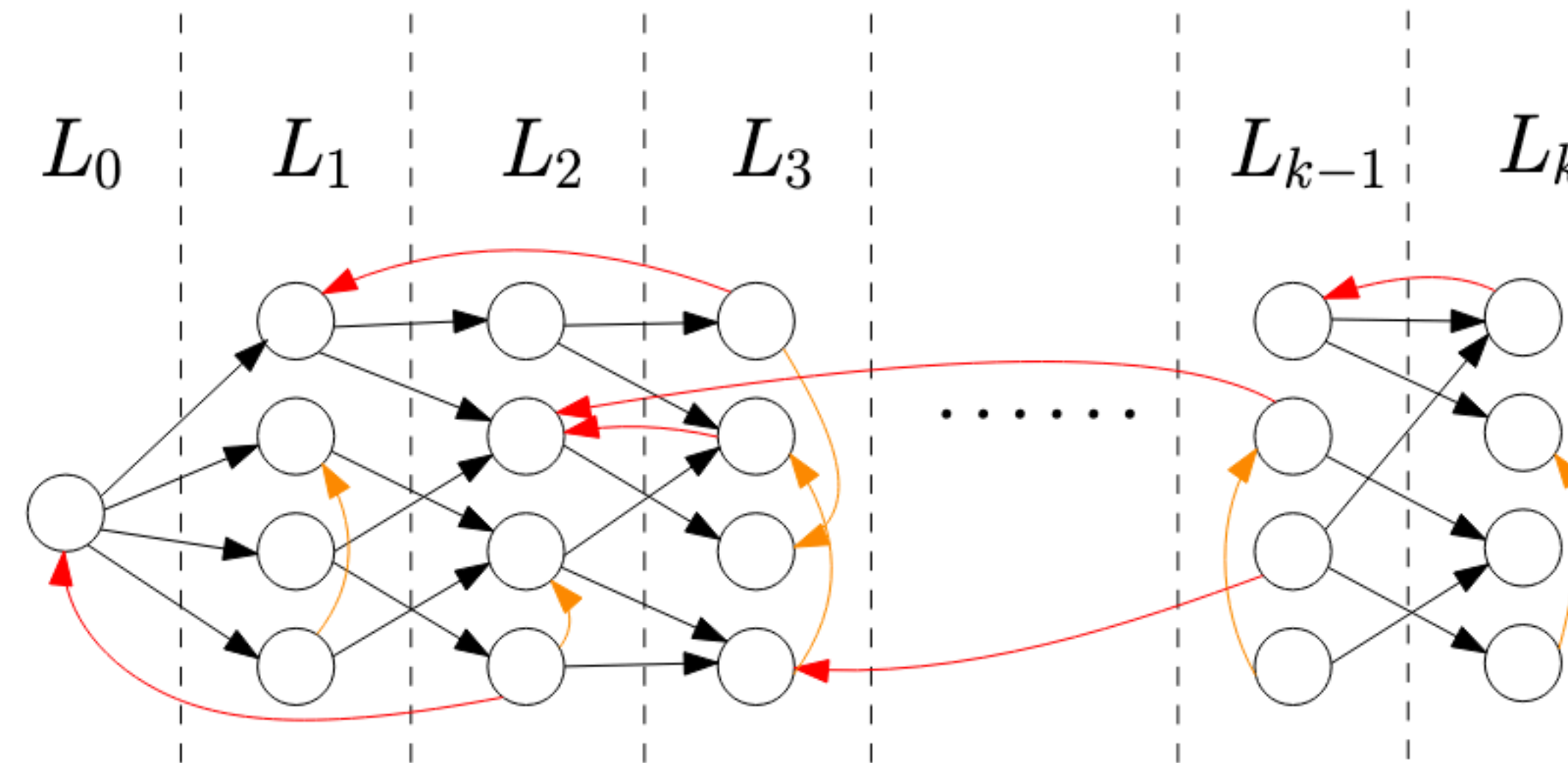


- Imagine the shortest s - t paths in G^f by layers
 - Forth edges: edges between adjacent layers
 - Side edges: edges between the same layer
 - Back edges: edges from further layer to nearer layer
 - Jumping edge: edge from nearer layer to non-adjacent further layer

Augmenting Path

Shortest augmenting [Dinitz, Edmonds-Karp]

Observation:
augmenting shortest path
results in **longer** path

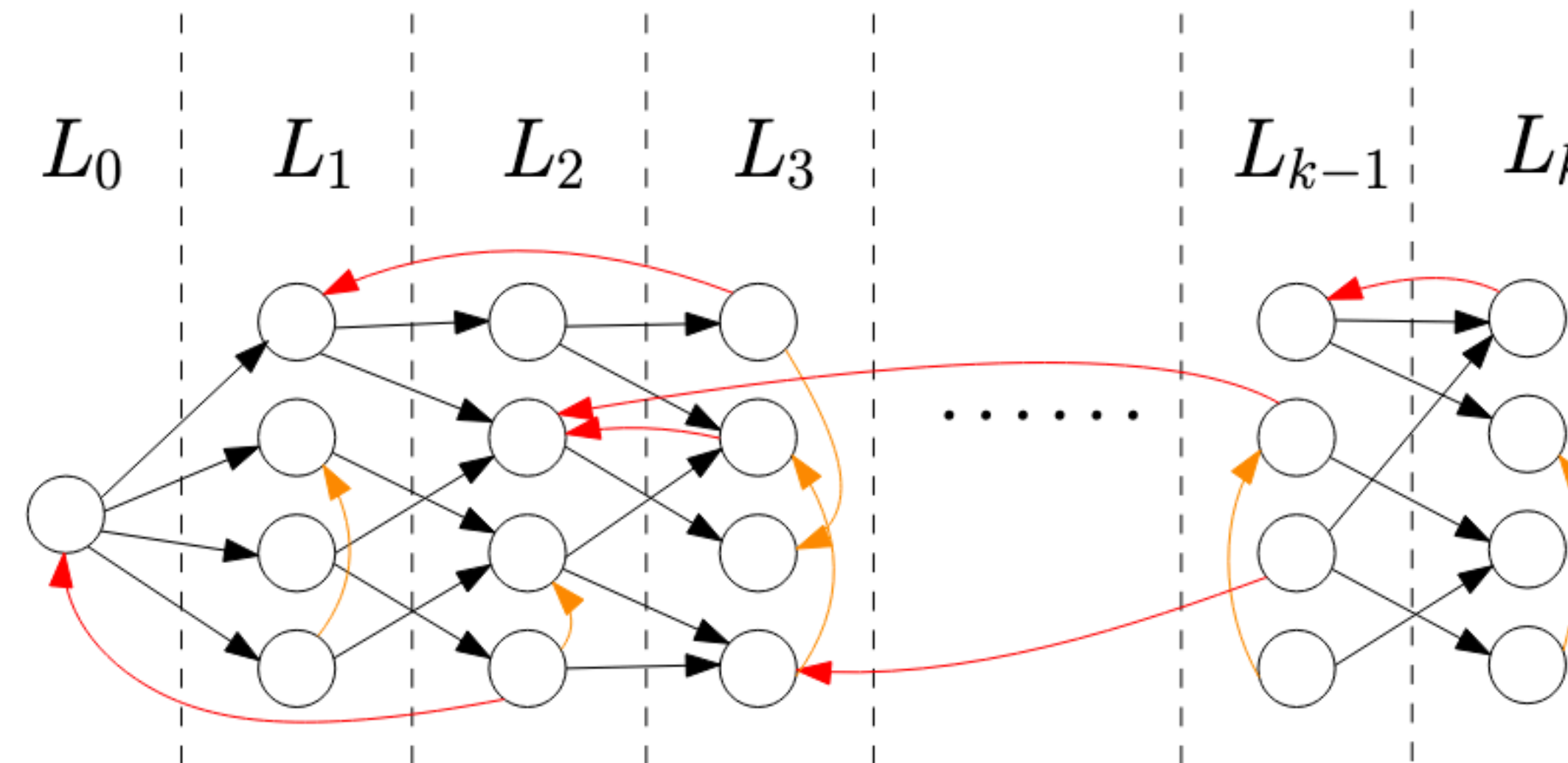


- Imagine the shortest s - t paths in G^f by layers
- Augmenting removes ≥ 1 forth edges, and adds ≥ 0 back edges
- No jumping edge is added by augmenting

Augmenting Path

Improving [Dinitz]

Observation:
augmenting shortest path
results in **longer** path

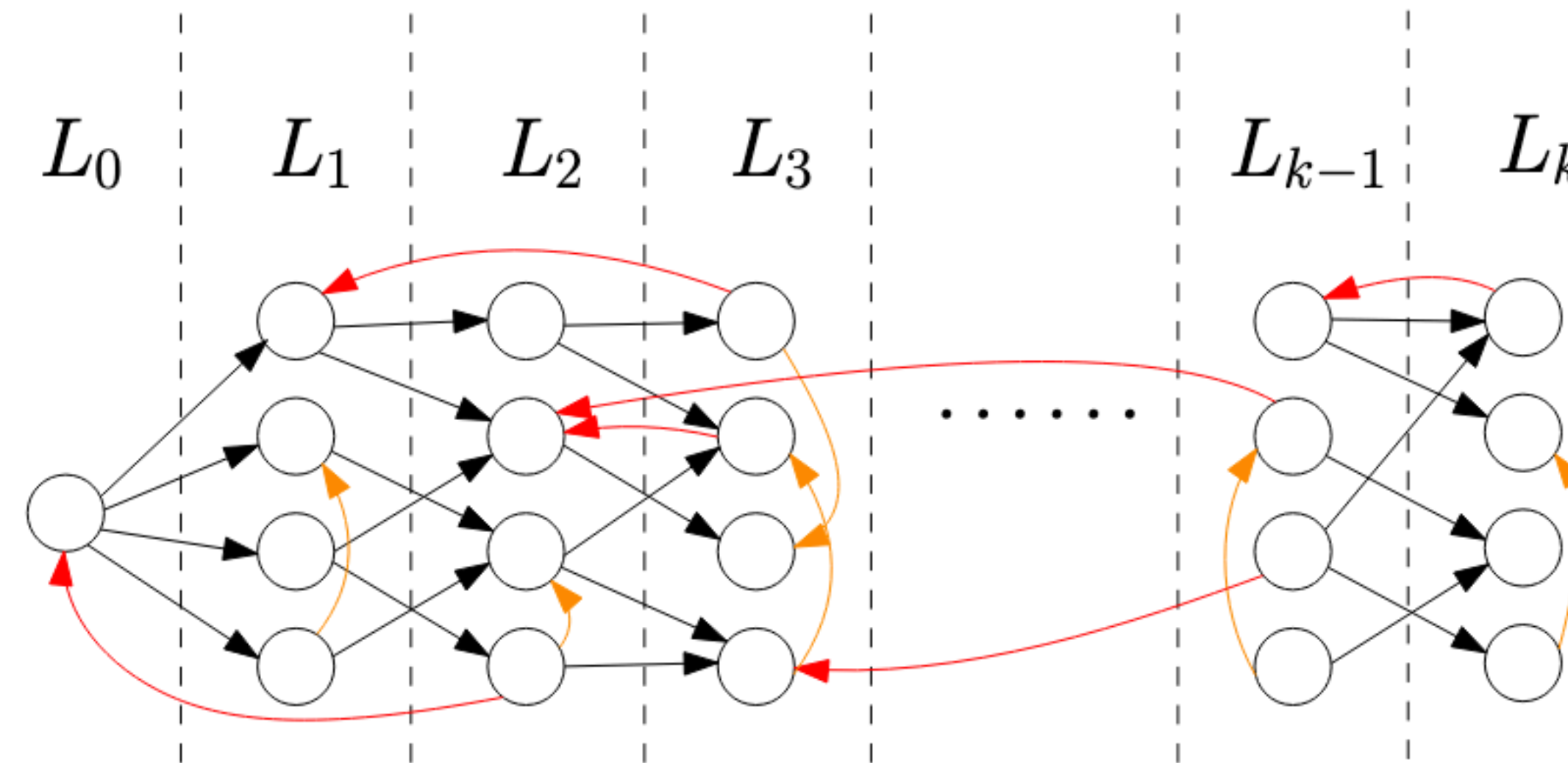


- Imagine the shortest $s-t$ paths in G^f by layers
- Augmenting removes ≥ 1 forth edges, and adds ≥ 0 back edges
- No jumping edge is added by augmenting
- Augmenting all shortest paths at one round: DFS only on forth edges.

Augmenting Path

Improving [Dinitz]

Observation:
augmenting shortest path
results in **longer** path



- Augmenting all shortest paths at one round: DFS only on forth edges.
 - DFS: $O(nm)$ time.
 - Path lengths: $1, \dots, n$.
- $O(n^2m)$ time in total

Augmenting Path

Algorithms	time	type
Ford-Fulkerson	$O(mf)$	pseudo-polynomial
Scaling	$O(m^2 \ln f)$	wealy-polynomial
Edmonds-Karp Dinitz	$O(m^2 n)$ $O(mn^2)$	strongly-polynomial

Polynomial time: in proportion to polynomial of n , where n is the **size of input**

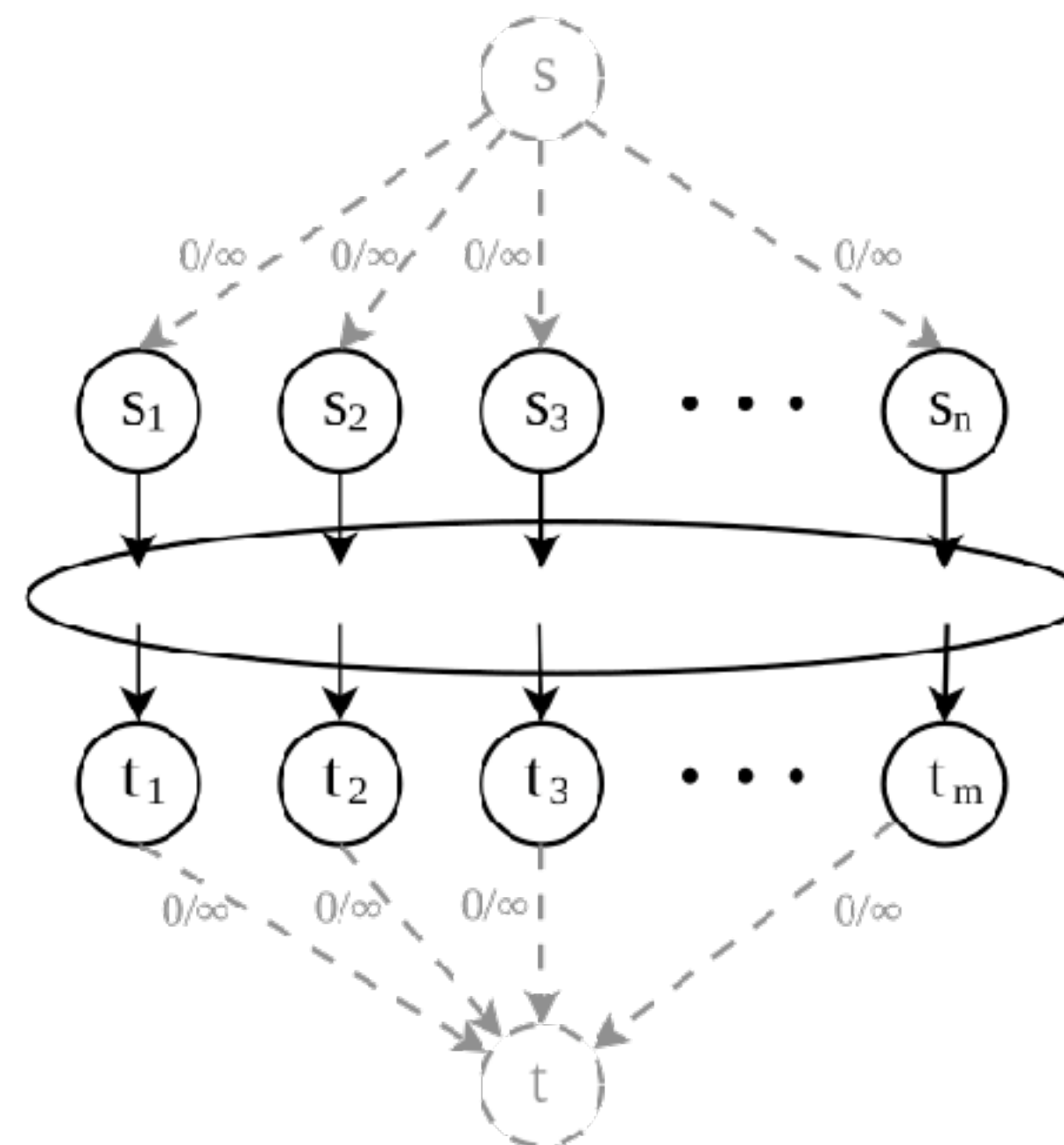
Maximum Flow Problem

Method	Year	Complexity
Linear programming		
Ford–Fulkerson algorithm	1955	$O(EU)$
Edmonds–Karp algorithm	1970	$O(VE^2)$
Dinic's algorithm	1970	$O(V^2E)$
MKM (Malhotra, Kumar, Maheshwari) algorithm ^[1]	1978	$O(V^3)$
Dinic's algorithm with dynamic trees	1983	$O(VE \log V)$
General push–relabel algorithm ^[2]	1986	$O(V^2E)$
Push–relabel algorithm with <i>FIFO</i> vertex selection rule ^[2]	1988	$O(V^3)$
Push–relabel algorithm with <i>maximum distance</i> vertex selection rule ^[3]	1988	$O(V^2\sqrt{E})$
Push-relabel algorithm with dynamic trees ^[2]	1988	$O\left(VE \log \frac{V^2}{E}\right)$
KRT (King, Rao, Tarjan)'s algorithm ^[4]	1994	$O\left(VE \log \frac{E}{V \log V} V\right)$
Binary blocking flow algorithm ^[5]	1998	$O\left(E \cdot \min\{V^{2/3}, E^{1/2}\} \cdot \log \frac{V^2}{E} \cdot \log U\right)$
James B Orlin's + KRT (King, Rao, Tarjan)'s algorithm ^[6]	2013	$O(VE)$
Kathuria-Liu-Sidford algorithm ^[7]	2020	$E^{4/3 + o(1)} U^{1/3}$
BLNPSSSW / BLLSSSW algorithm ^{[8][9]}	2020	$\tilde{O}((E + V^{3/2}) \log U)$
Gao-Liu-Peng algorithm ^[10]	2021	$\tilde{O}(E^{\frac{3}{2} - \frac{1}{328}} \log U)$
Chen, Kyng, Liu, Peng, Gutenberg and Sachdeva's algorithm ^[11]	2022	$O(E^{1+o(1)} \log U)$
Bernstein, Blikstad, Saranurak, Tu ^[12]	2024	$O(n^{2+o(1)} \log U)$

Multi-Source Multi-Sink

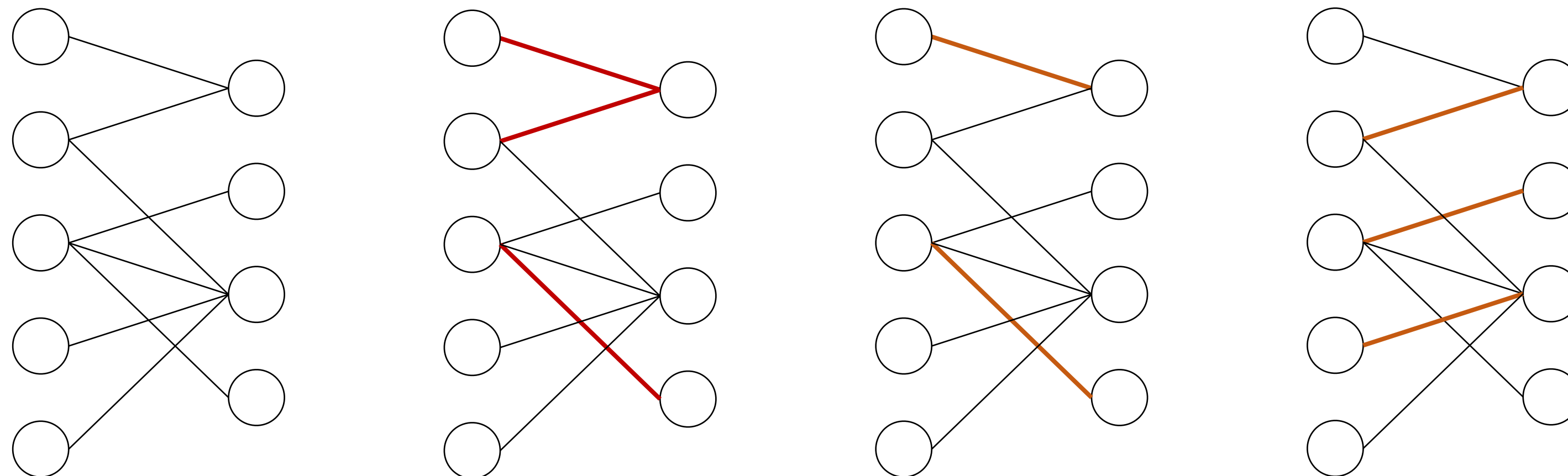
Input: A network $G = (V, E)$ and capacity $c : E \rightarrow \mathbb{R}^+$, with sources $S = \{s_1, \dots, s_n\}$ and sinks $T = \{t_1, \dots, t_m\}$.

Output: A maximum flow from S to T across G .



Maximum Bipartite Matching

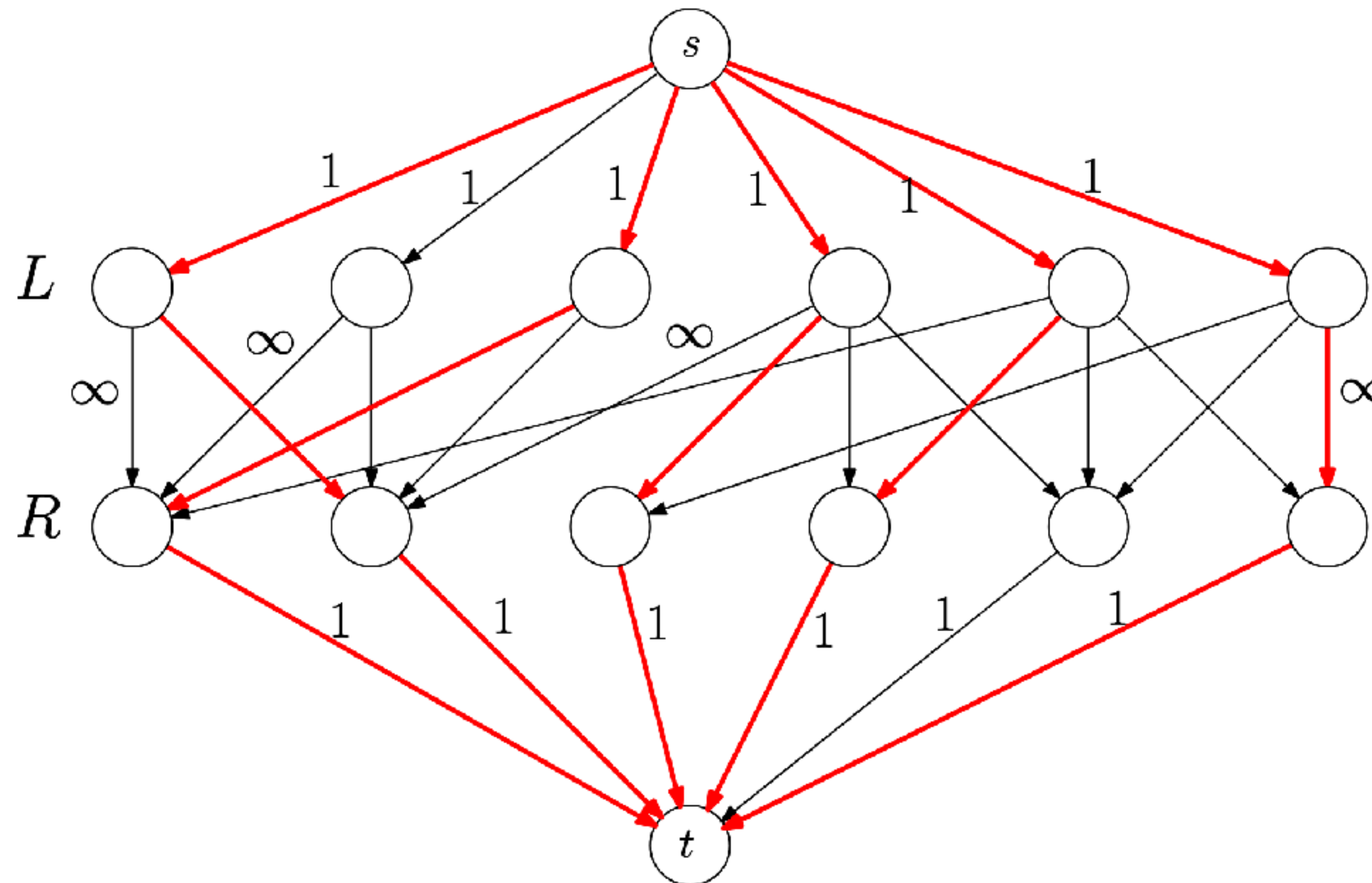
- A graph $G = (V, E)$ is **bipartite** if V can be partitioned into L and R , such that each edge connects a vertex in L and a vertex in R .
- A **matching** of a graph $G = (V, E)$ is a subset $M \subseteq E$, such that for each $v \in V$, at most one edge in M is incident on v .
- A **maximum matching** is a matching of maximum cardinality.
- **Problem:** Find a maximum matching of a bipartite graph.



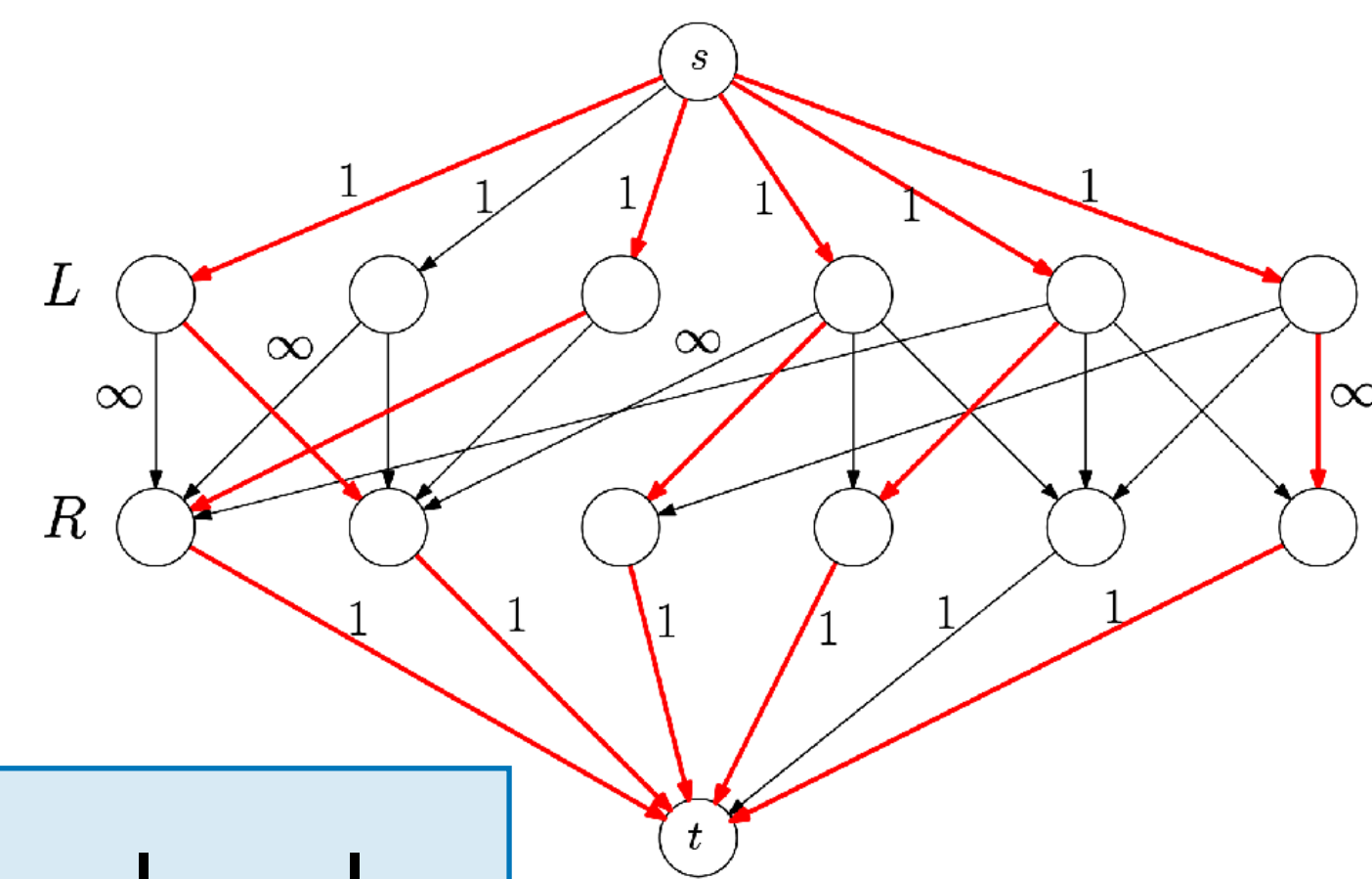
Maximum Bipartite Matching

Input: A bipartite graph $G = (L \cup R, E)$.

Output: A maximum matching of a bipartite graph.



Maximum Bipartite Matching

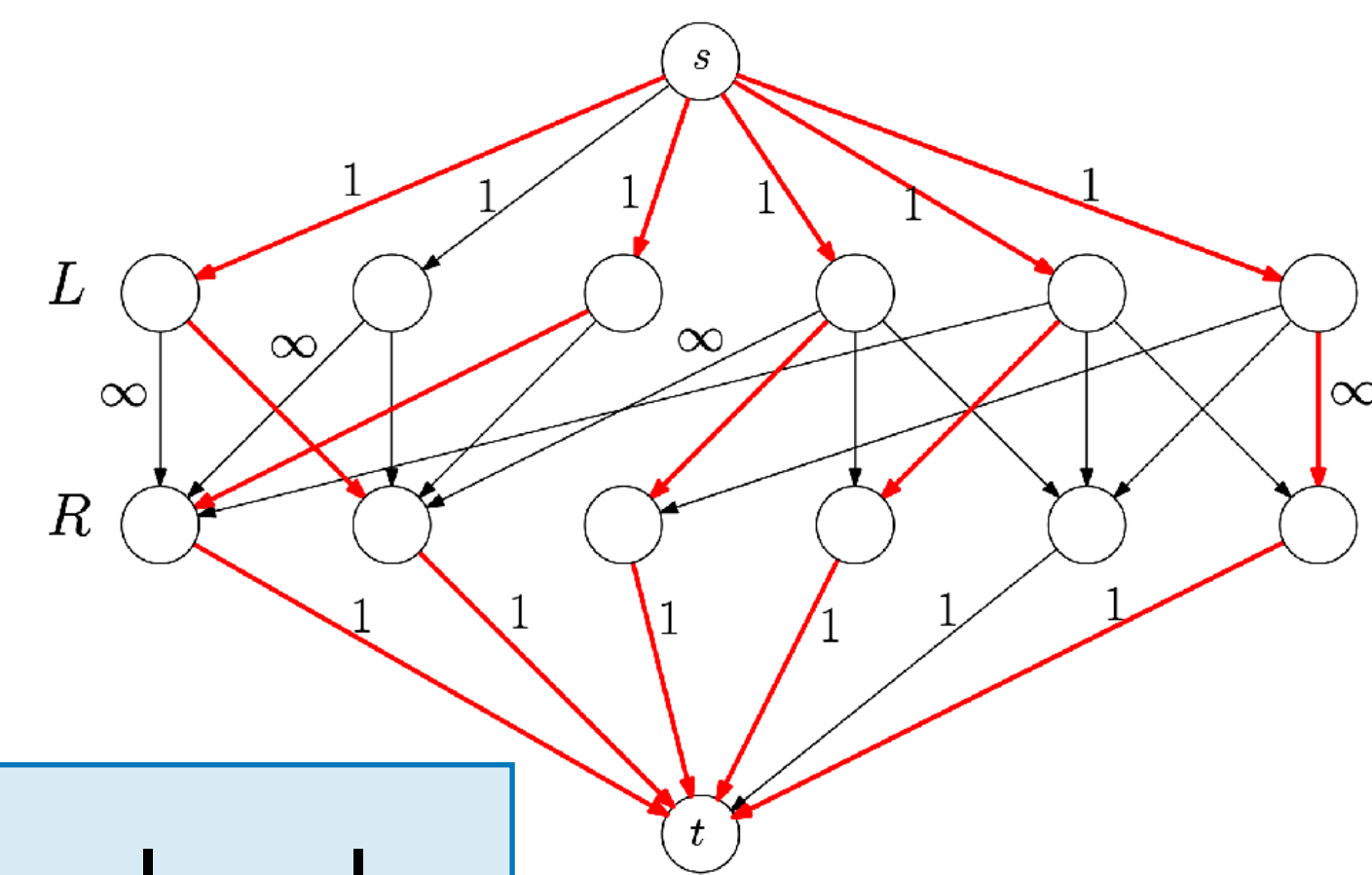


- (1) \exists matching M of $G \Rightarrow \exists$ flow f of G' with $|f| = |M|$
- (2) \exists flow f of $G' \Rightarrow \exists$ matching M of G with $|M| = |f|$.

(1):

- For each $e = (u, v) \in M$, let $f_{su} = f_e = f_{vt} = 1$
- For each $e = (u, v) \notin M$, let $f_e = 0$
- f is valid, and $|f| = |M|$

Maximum Bipartite Matching



- (1) \exists matching M of $G \Rightarrow \exists$ flow f of G' with $|f| = |M|$
- (2) \exists flow f of $G' \Rightarrow \exists$ matching M of G with $|M| = |f|$.

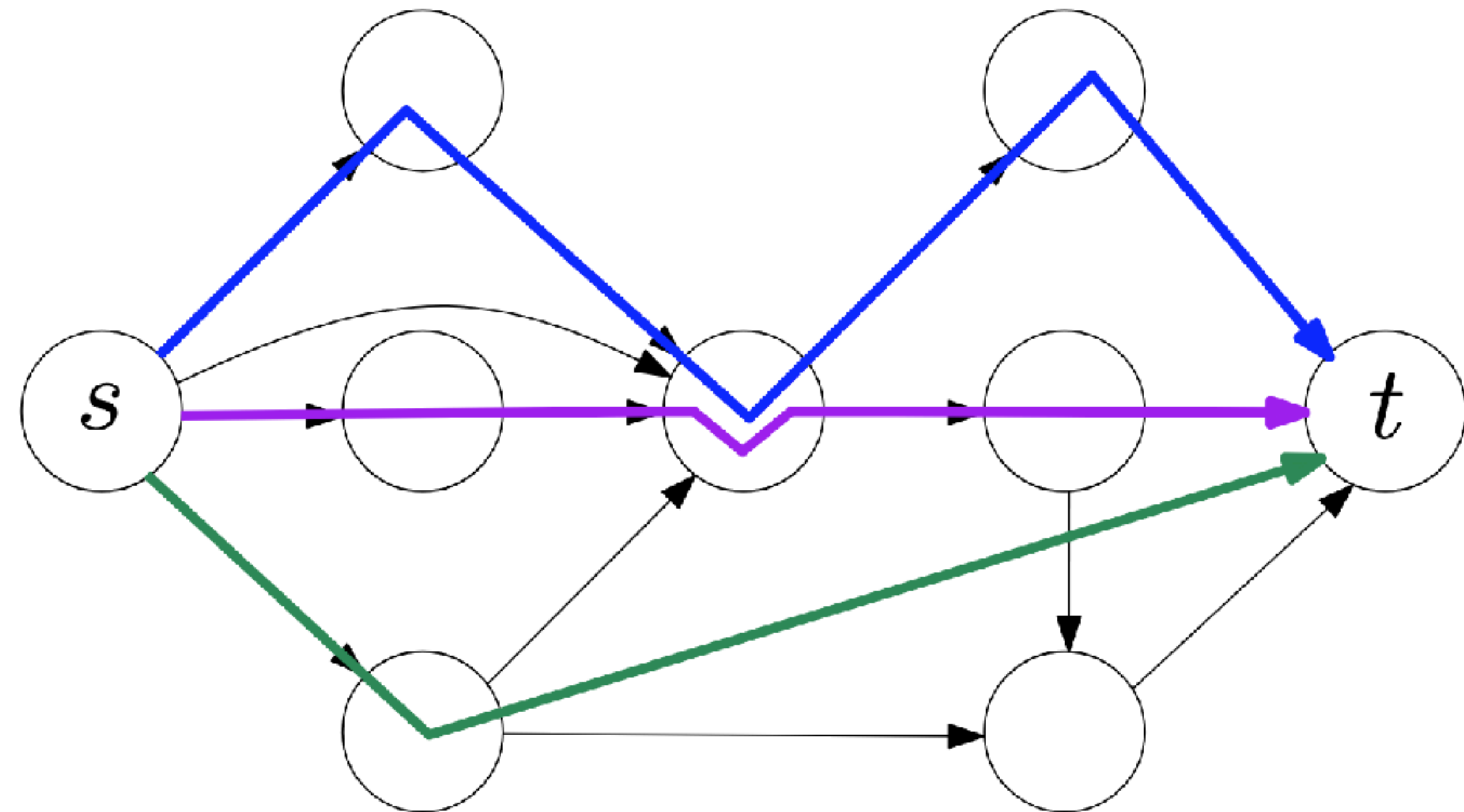
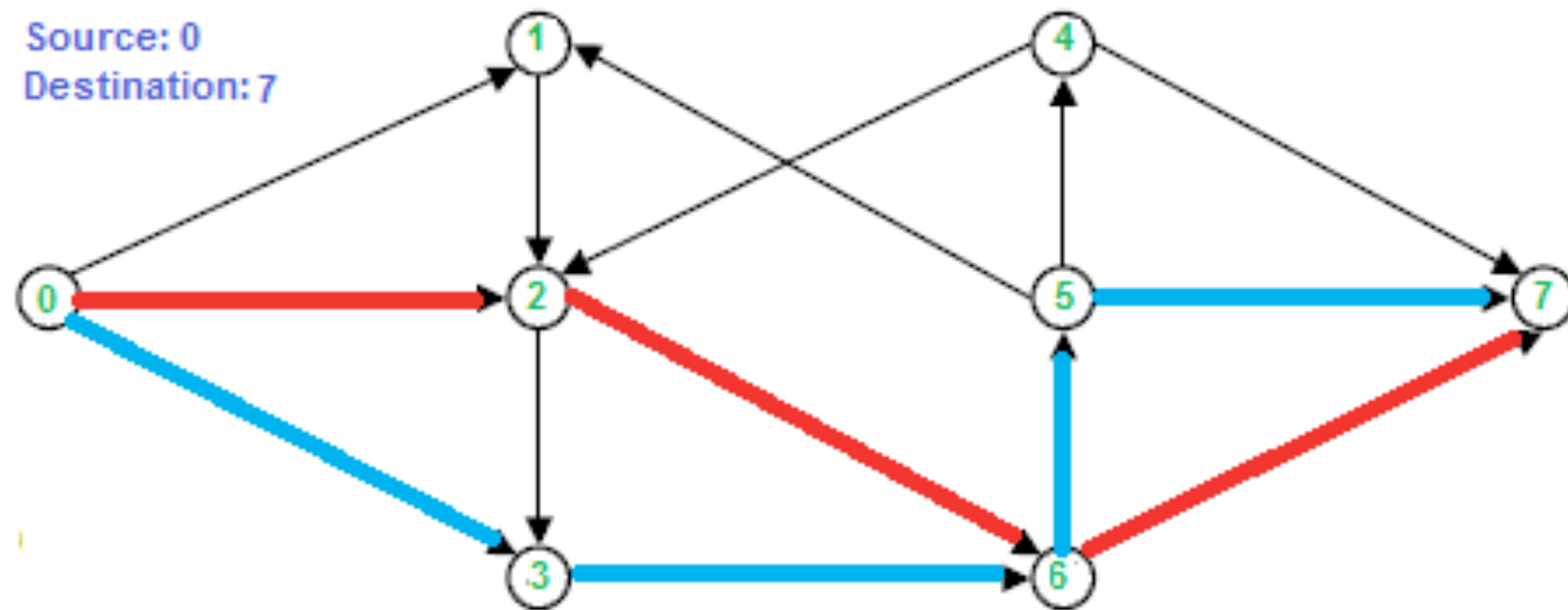
(2): Let $M \triangleq \{(u, v) \in E : u \in L, v \in R, f_e > 0\}$

- M is a matching:
 - for any $u \in L$, $\sum_{v \in \delta(u) \setminus s} f_{uv} \leq 1$; for any $v \in R$, $\sum_{u \in \delta(v) \setminus t} f_{uv} \leq 1$.
 - Any $u \in L$ matches with at most one $v \in R$, so does any $v \in R$.
- $|M| = |f|$

#Disjoint Paths

Input: A digraph $G = (V, E)$ and source & sink $s, t \in V$.

Output: The maximum **#edge-disjoint** paths from s to t



s - t Min-Cut

Input: A digraph $G = (V, E)$ and weights $c : E \rightarrow \mathbb{R}^+$, and source & sink $s, t \in V$.

Output: The minimum cut whose removal disconnects s, t .

The following statements are equivalent:

- (1) f is a maximum flow
- (2) There is no s - t path P in G^f with $\delta(P) > 0$
- (3) There is $S, \neg S \subseteq V$ such that $\text{cut}(S) = |f|$

Global Min-Cut

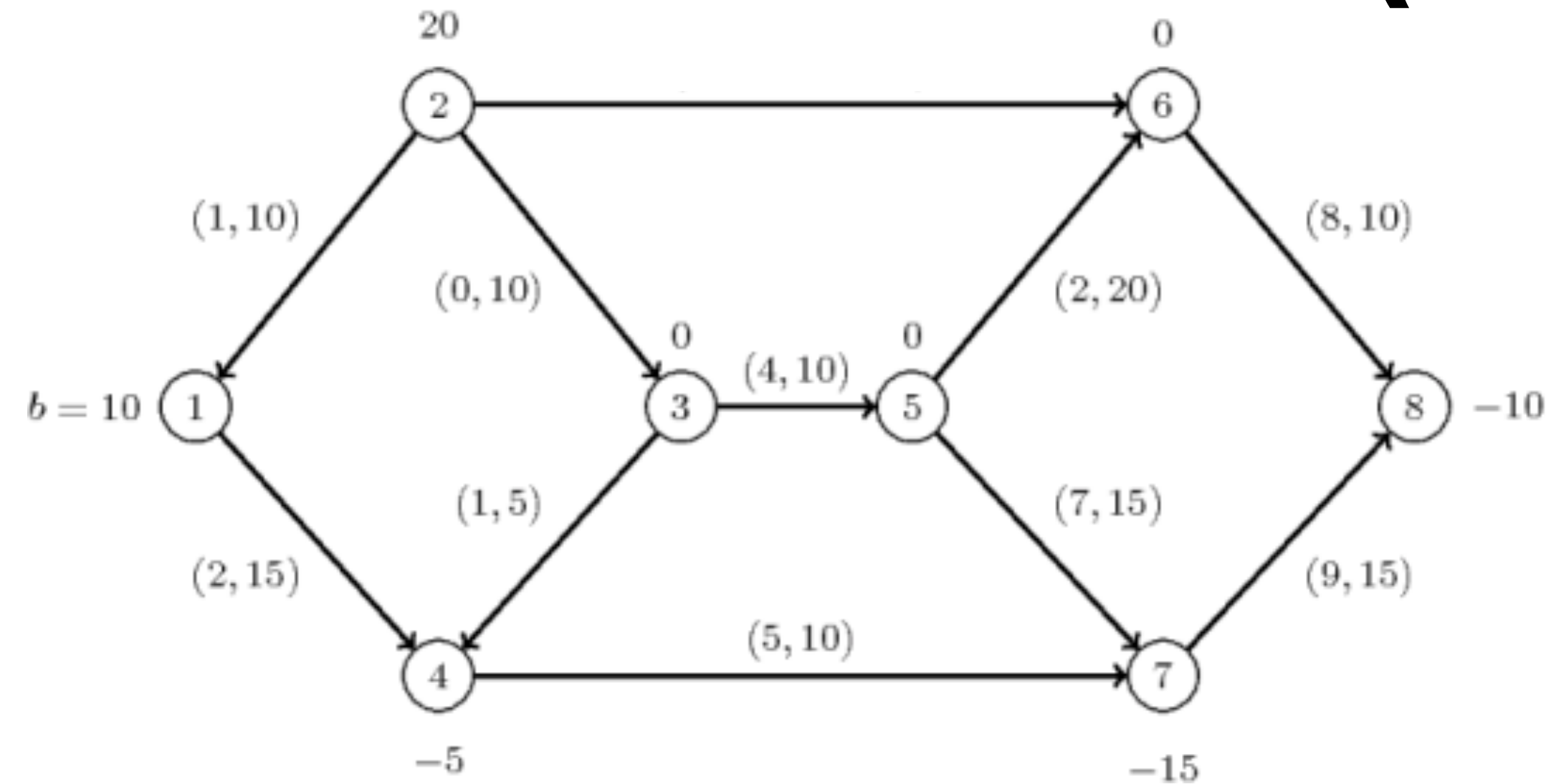
Input: A digraph $G = (V, E)$ and weights $c : E \rightarrow \mathbb{R}^+$.

Output: The minimum cut whose removal disconnects G

The following statements are equivalent:

- (1) f is a maximum flow
- (2) There is no s - t path P in G^f with $\delta(P) > 0$
- (3) There is $S, \neg S \subseteq V$ such that $\text{cut}(S) = |f|$

Minimum-Cost Flow Problem (MCFP)



- Edge capacity $c_e \in \mathbb{R}^+$. **Flow cost** $a_e \in \mathbb{R}^+$.
- Flow function $f: E \rightarrow \mathbb{R}^+$
 - Valid flow: $\forall e \in E, 0 \leq f(e) \leq c_e$ & $\forall v \in V, \sum_{e \in \delta_{in}(v)} f(e) = \sum_{e \in \delta_{out}(v)} f(e)$
- MCFP: find a flow f minimizes $\sum_{e \in \delta_{out}(s)} a_e \cdot f(e)$ while maximizing $\sum_{e \in \delta_{out}(s)} f(e)$

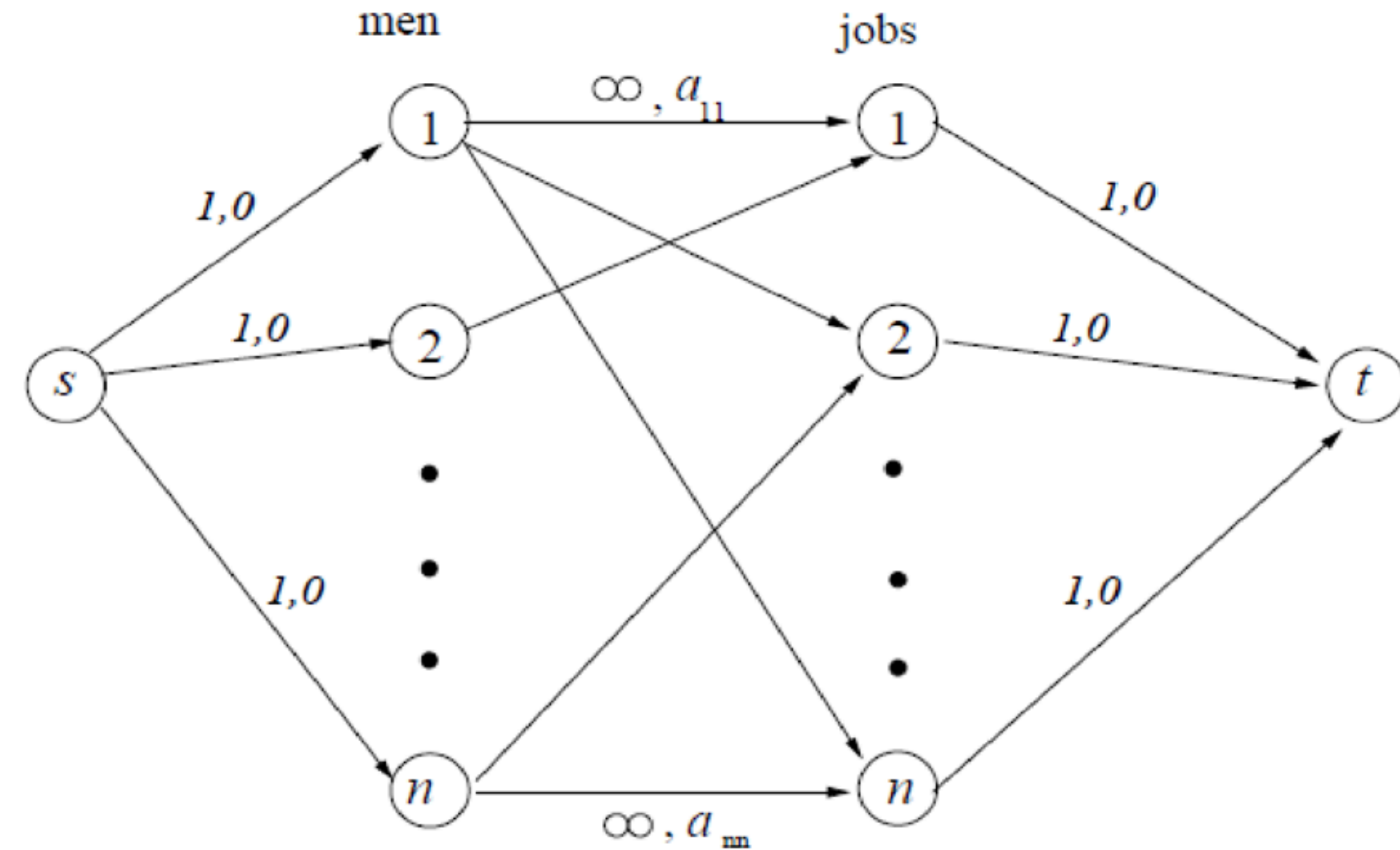
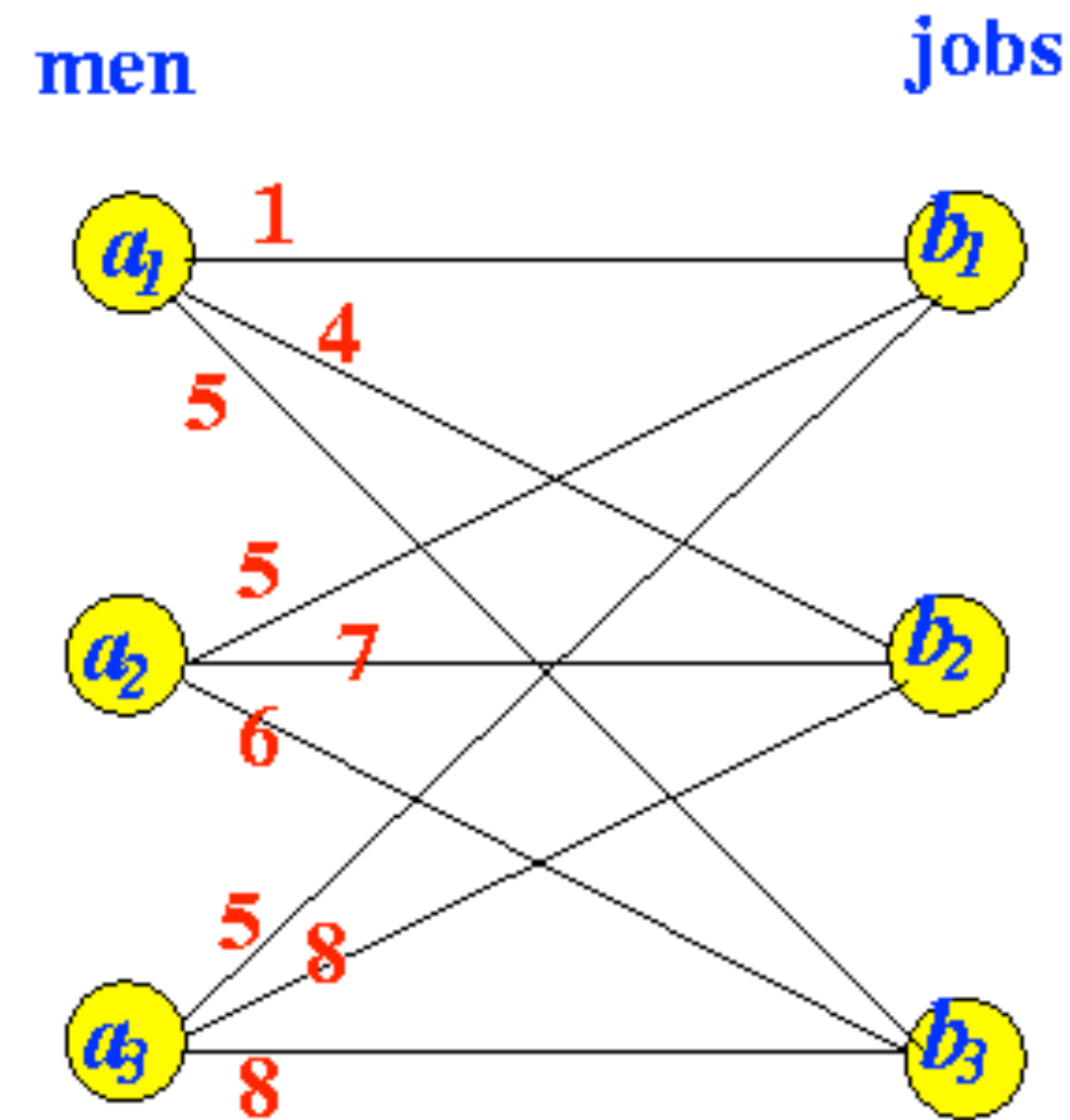
Assignment Problem

Minimum-cost perfect matching

Given matrix of costs

Worker	Task		
	I	II	III
Ali	8	4	7
Baba	5	2	3
Curi	9	6	7
Durian	9	4	8

Make square with dummy column.
Subtract minimum for each column:



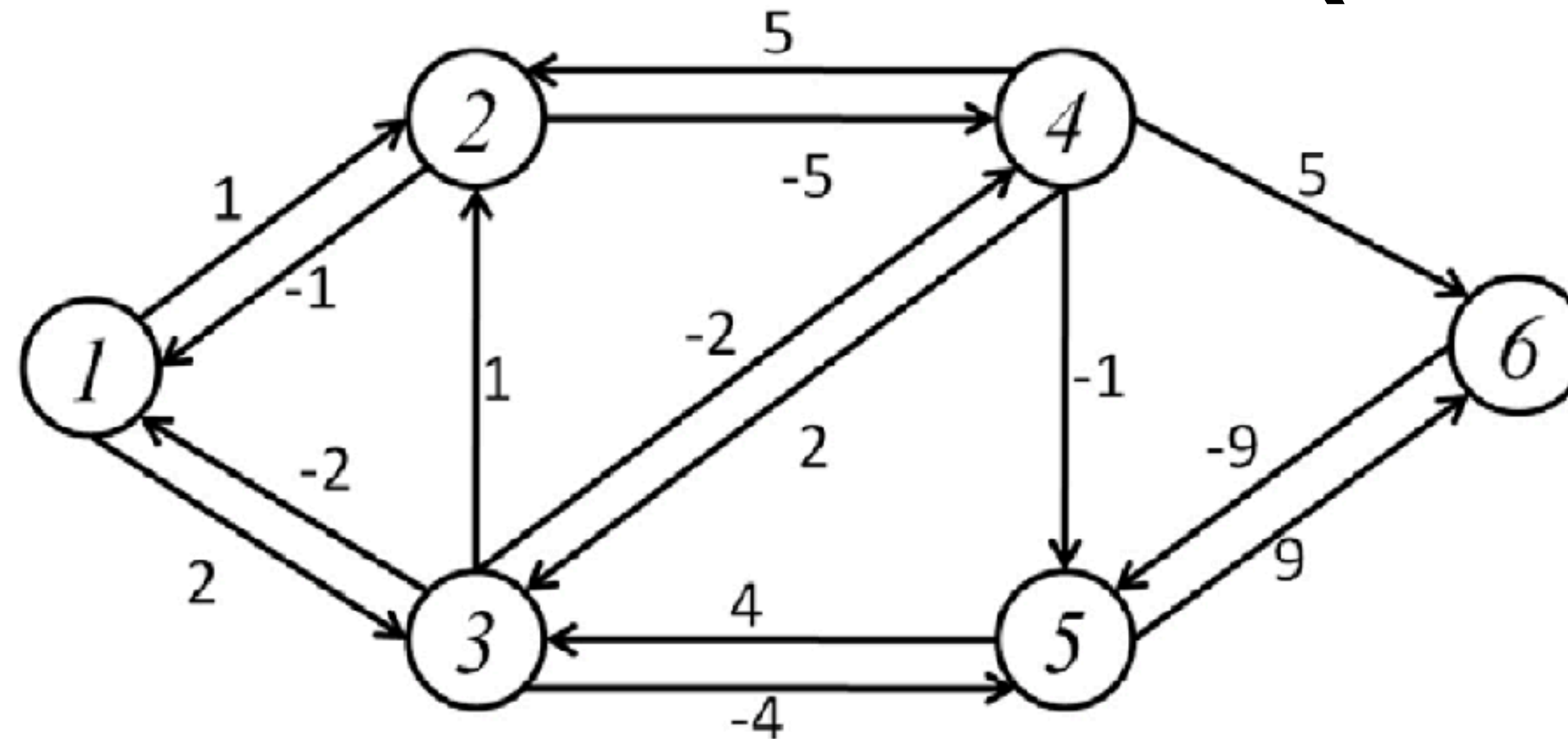
Minimum-Cost Flow Problem (MCFP)

Input: A digraph $G = (V, E)$ and capacity & costs $c, a : E \rightarrow \mathbb{R}^+$,
and source s , sink t .

Output: A flow f minimizes $\sum_{e \in \delta_{out}(s)} a_e \cdot f(e)$ while maximizing $\sum_{e \in \delta_{out}(s)} f(e)$

- An optimal solution be like: a maximum flow which cannot be improved
- An improvement be like: redirecting the flow with lower cost
- A redirection be like:
 - pushing backward a flow from t to s , pushing forward a flow from s to t

Minimum-Cost Flow Problem (MCFP)



- An optimal solution be like: a maximum flow which cannot be improved
- An improvement be like: redirecting the flow with lower cost
- A redirection be like:
 - pushing backward a flow from t to s , pushing forward a flow from s to t
 - a cycle in the residual graph
- An improving redirection be like:
 - a negative cycle in the residual graph

Minimum-Cost Flow Problem (MCFP)

Input: A digraph $G = (V, E)$ and capacity & costs $c, a : E \rightarrow \mathbb{R}^+$, and source s , sink t .

Output: A flow f minimizes $\sum_{e \in \delta_{out}(s)} a_e \cdot f(e)$ while maximizing $\sum_{e \in \delta_{out}(s)} f(e)$

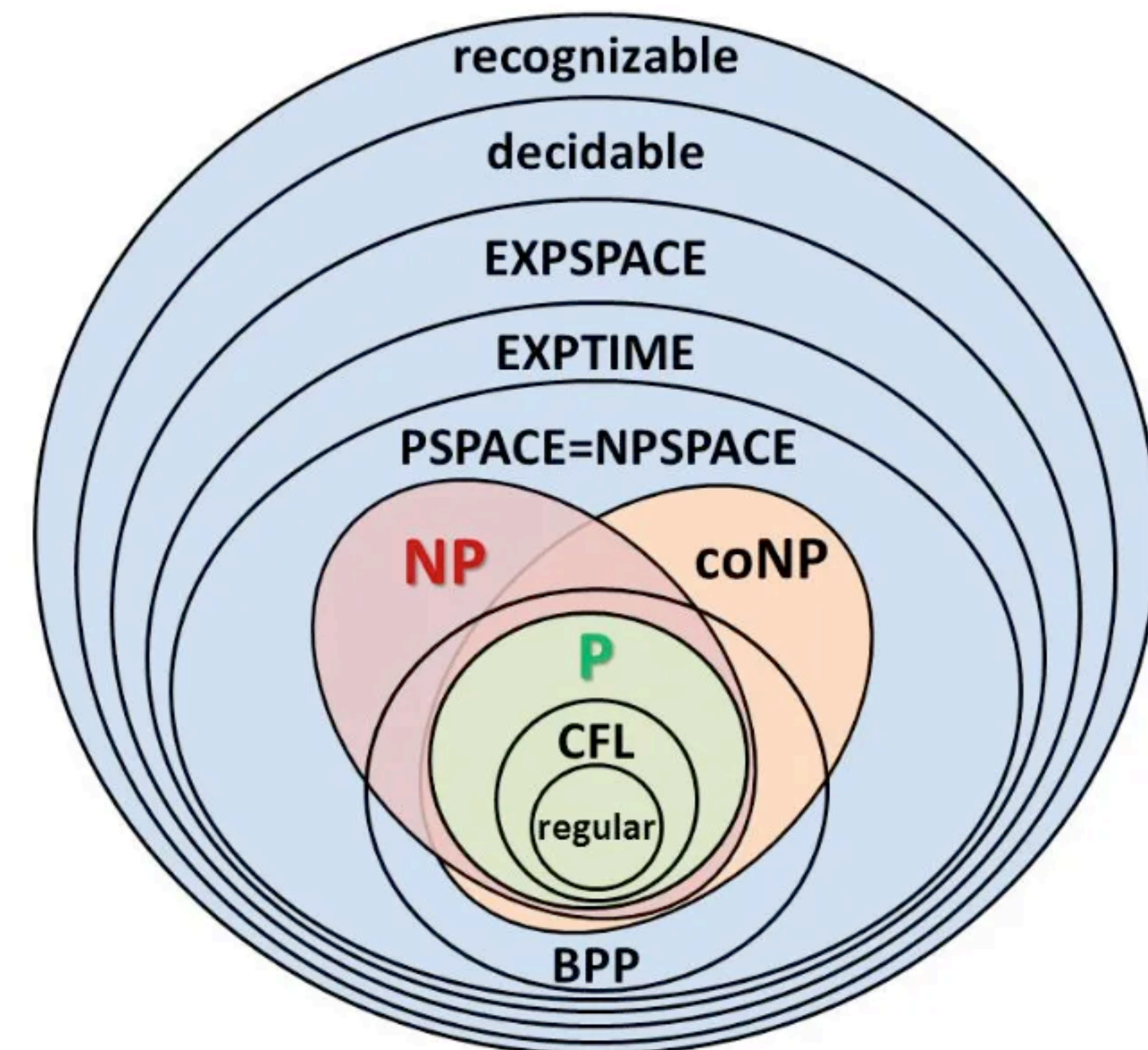
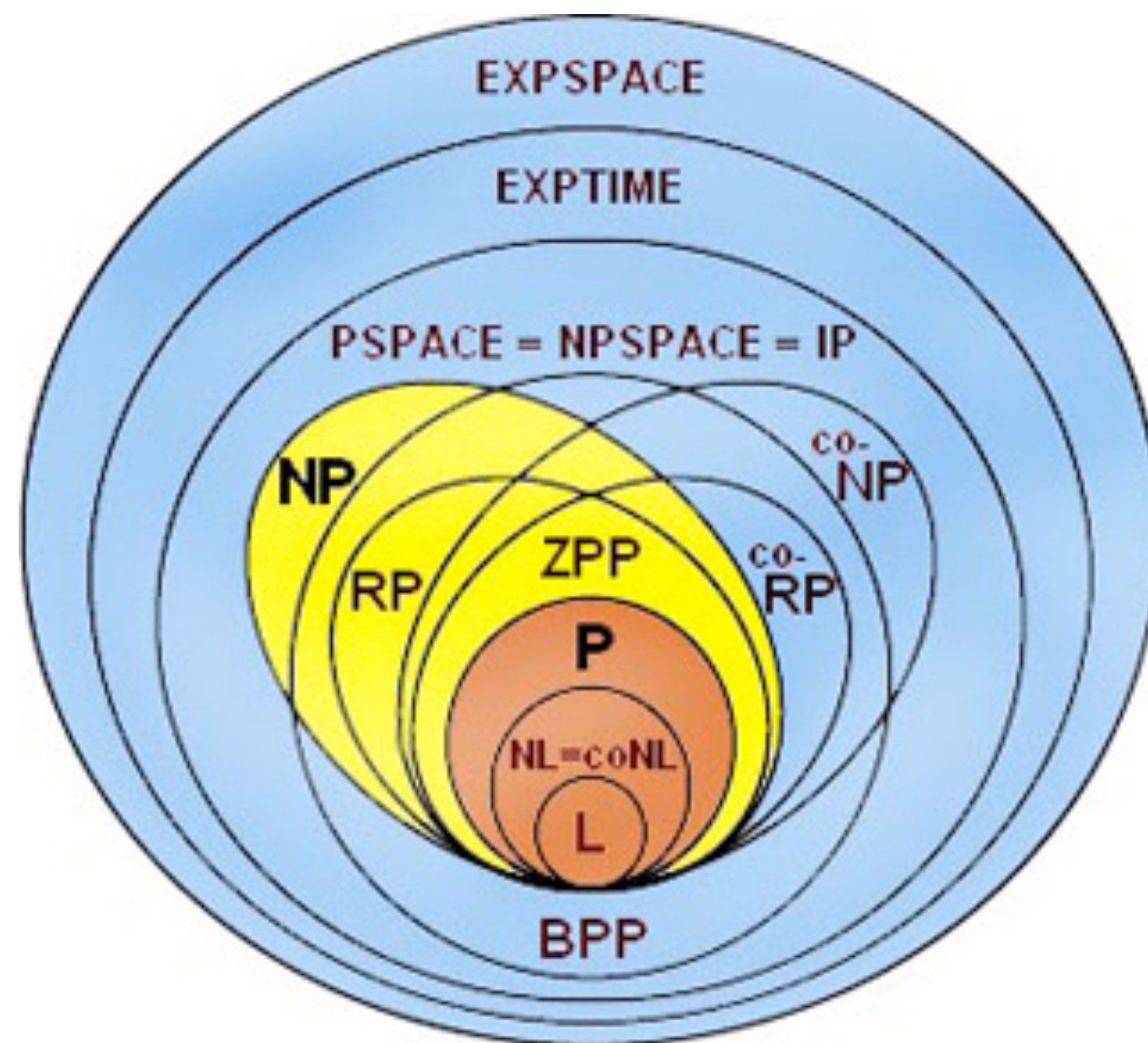
- An improving redirection be like:
 - a negative cycle in the residual graph

Cycle Canceling:

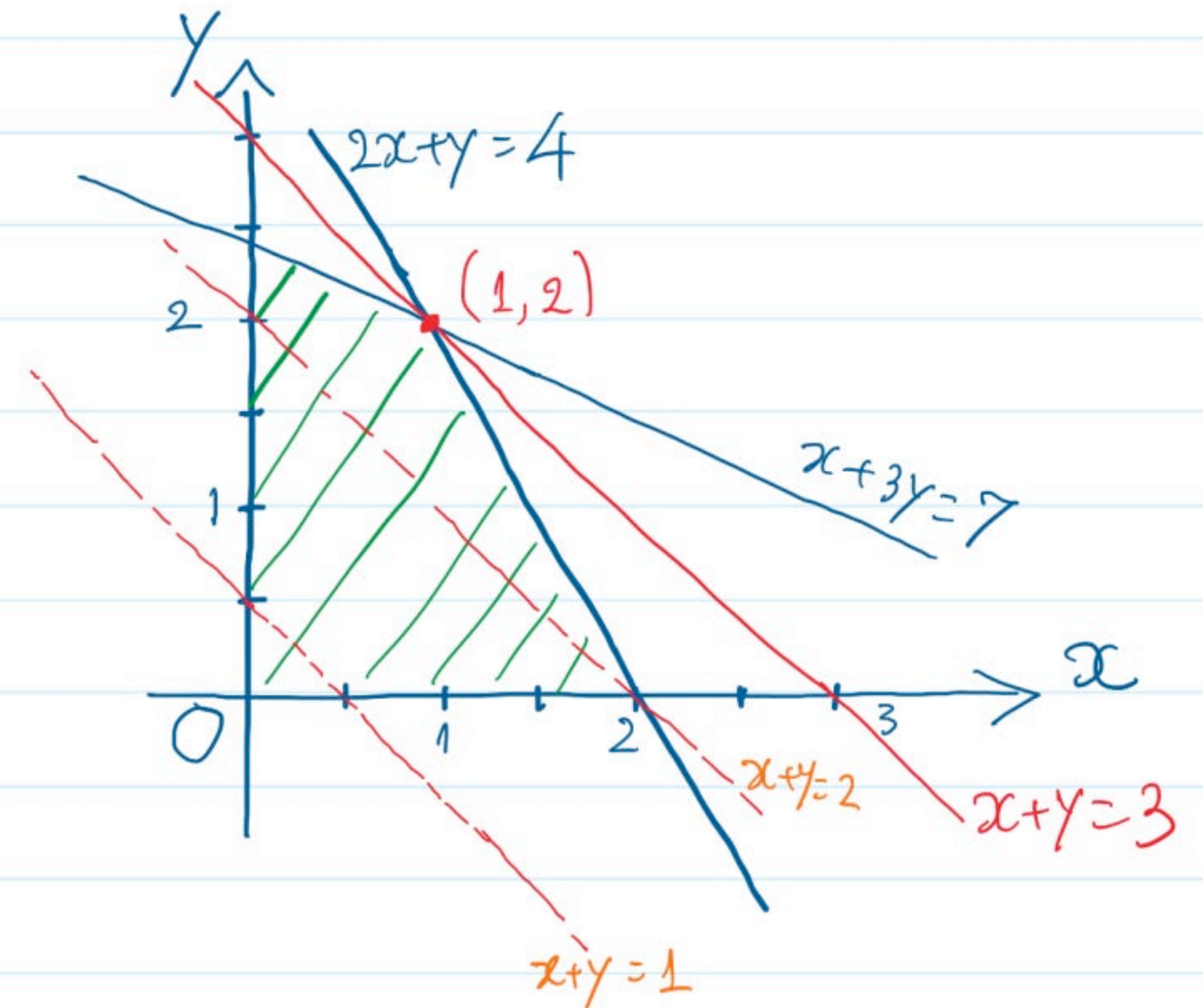
While exists negative cycle in residual graph
cancel the negative cycle by redirecting

Maximum Flow Is P -Complete

Under log space reduction



Linear Programming



A Familiar Problem

某厂生产甲乙两种产品,每生产 1 吨产品的电耗、煤耗、所需劳动力及产值如表 3 所示:

表 3

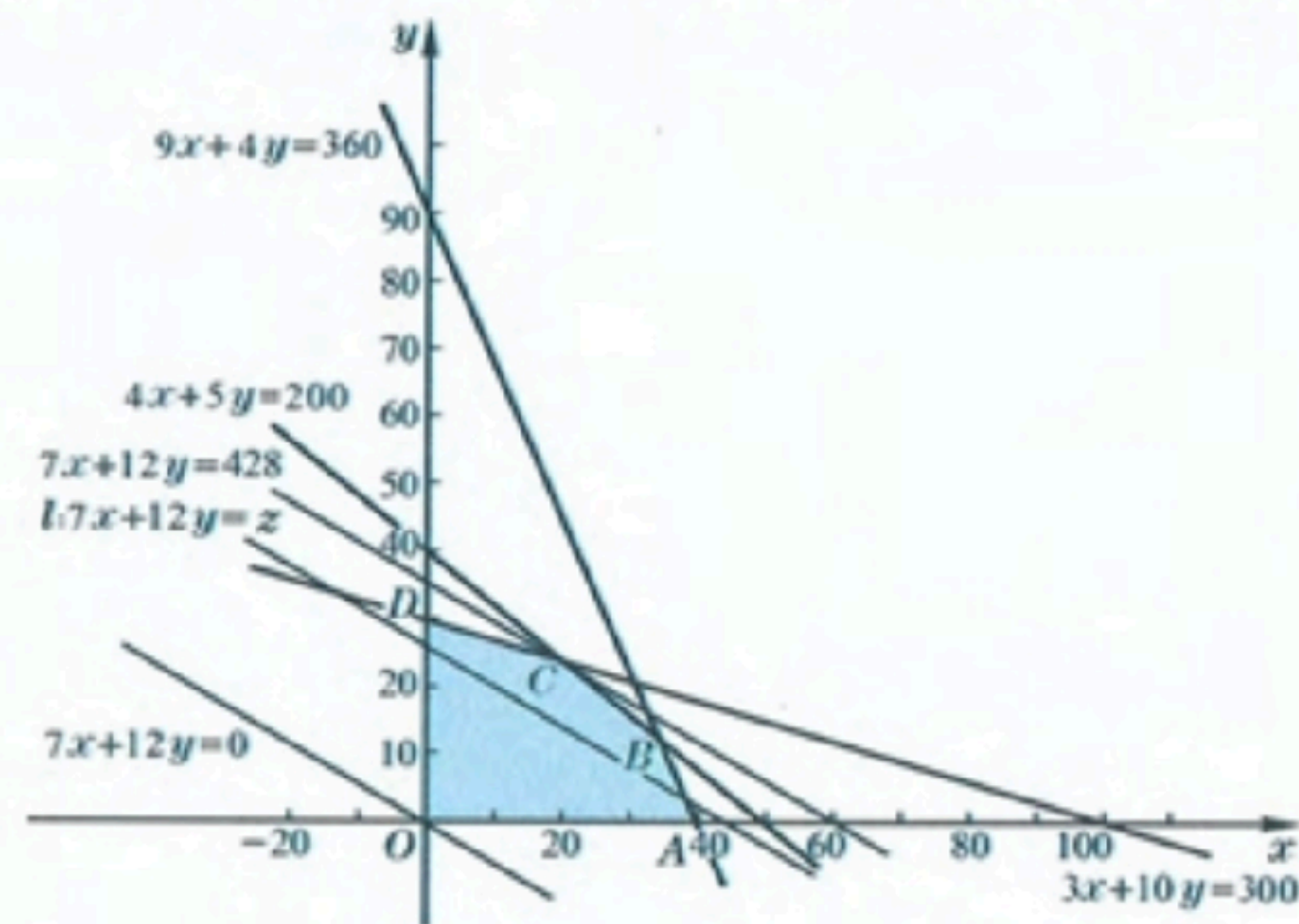
产品	电耗 (千瓦时)	煤耗 (吨)	劳动力(人)	产值(万元)
甲	4	9	3	7
乙	5	4	10	12

已知该厂有劳动力 300 人,按计划煤耗每天不超过 360 吨,电耗每天不超过 200 千瓦时.每天应如何安排生产,可使产值最大?

如果设该厂每天生产甲产品 x 吨,乙产品 y 吨,那么上述问题可转化为在满足以下线性约束条件:

$$(B) \begin{cases} 9x+4y \leq 360, \\ 4x+5y \leq 200, \\ 3x+10y \leq 300, \\ x \geq 0, \\ y \geq 0, \end{cases}$$

求线性目标函数 $z=7x+12y$ 的最大值.



minimize $7x_1 + 4x_2$

subject to $x_1 + x_2 \geq 5$

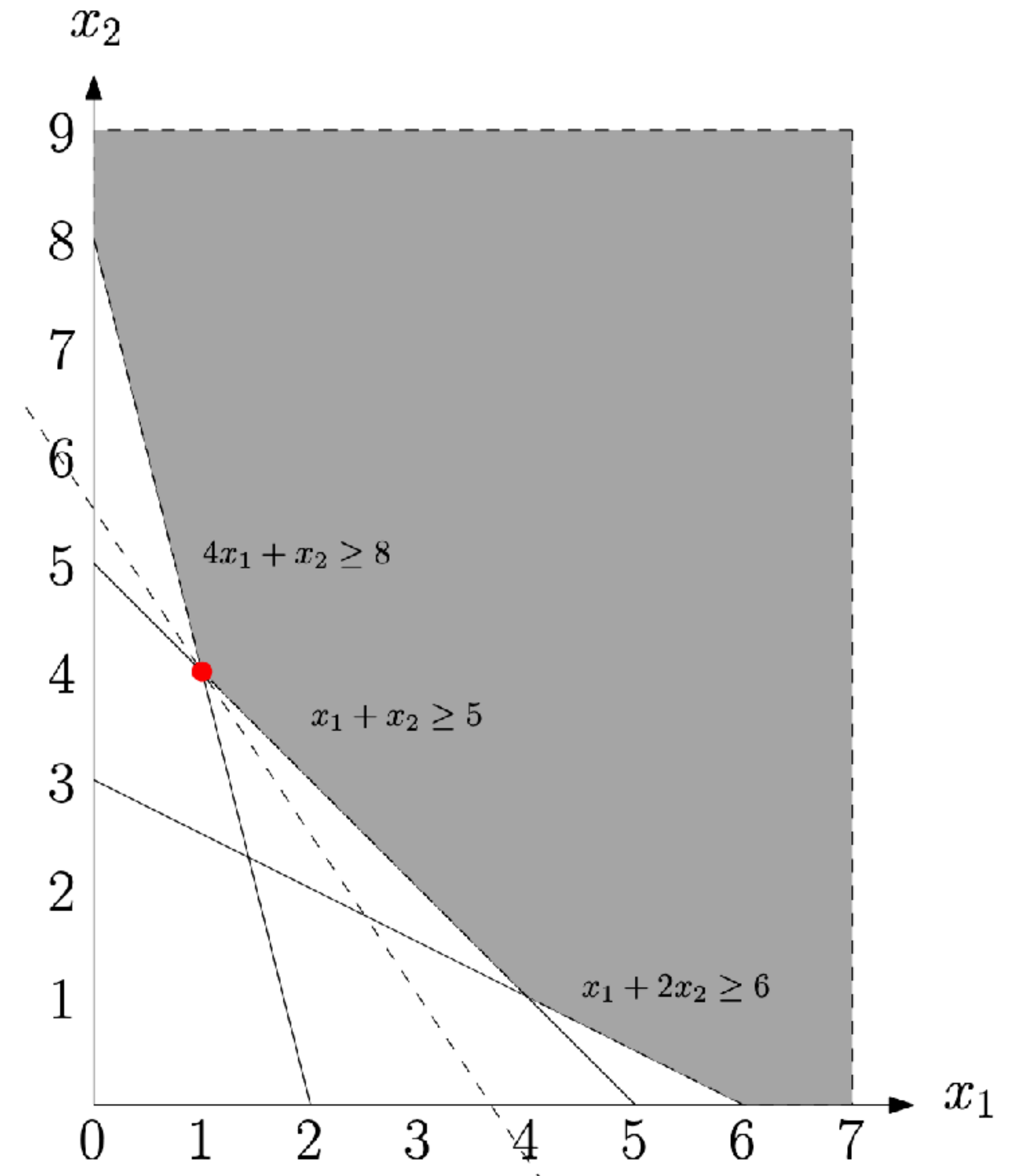
$$x_1 + 2x_2 \geq 6$$

$$4x_1 + x_2 \geq 8$$

$$x_1, x_2 \geq 0$$

optimal point $x_1 = 1, x_2 = 4$

$$\text{value} = 7 \times 1 + 4 \times 4 = 23$$



Linear Programming (LP)

- General form:
 - matrix $A = \{a_{ij}\}_{[m] \times [n]}$, sets $M \subseteq [m]$ and $N \subseteq [n]$

$$\begin{array}{llll} \text{minimize} & \mathbf{c}^T \mathbf{x} & & \\ \text{subject to} & \mathbf{a}_i^T \mathbf{x} = b_i & i \in M & \\ & \mathbf{a}_i^T \mathbf{x} \geq b_i & i \in \bar{M} & \\ & x_j \geq 0 & j \in N & \\ & x_j \text{ unconstrained} & j \in \bar{N} & \end{array}$$

Max-Flow

digraph: $G = (V, E)$ source: s sink: t

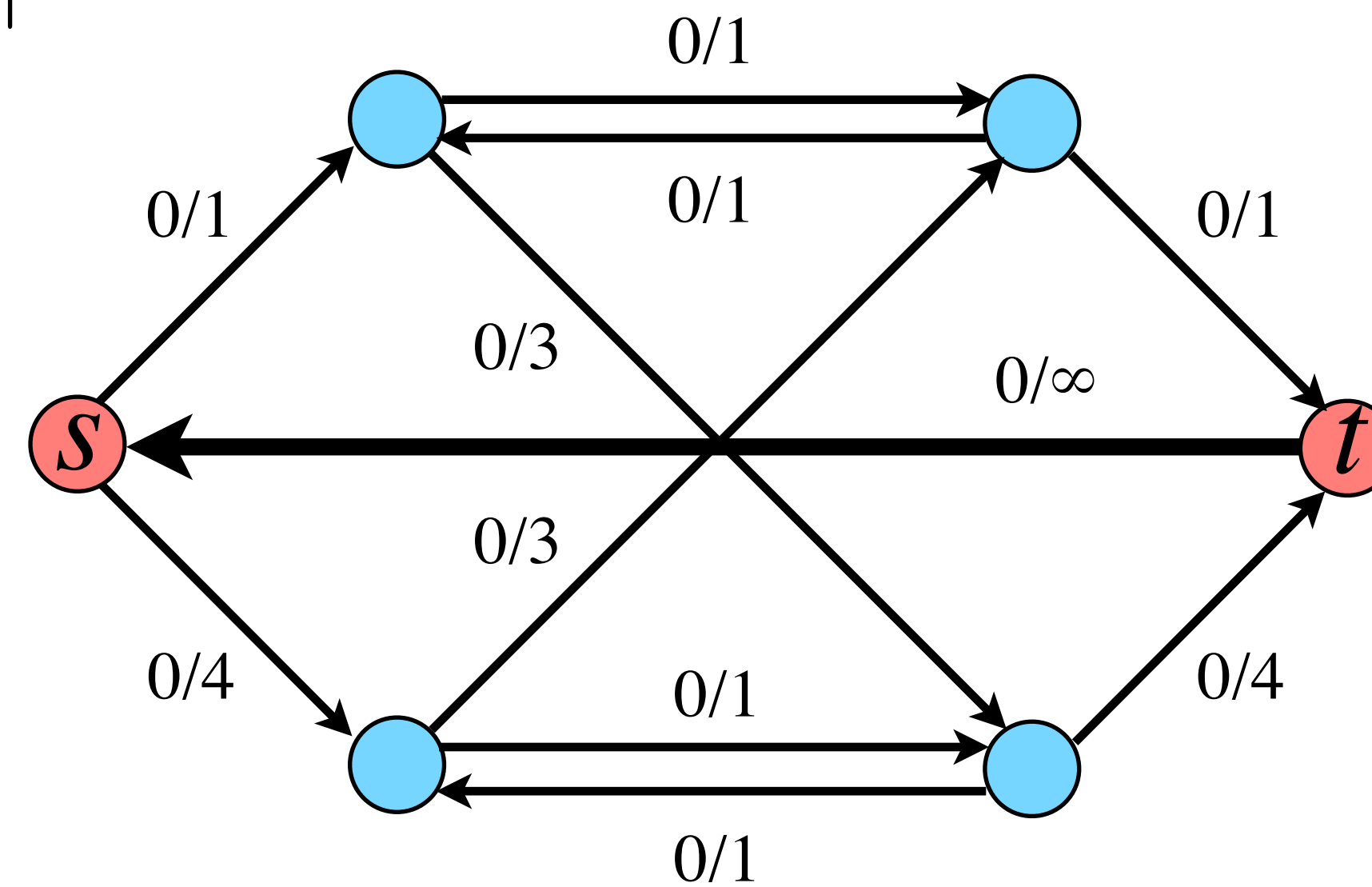
capacity: $c : E \rightarrow \mathbb{R}^+$

$$\max \sum_{u:(s,u) \in E} f_{su}$$

$$\text{s.t.} \quad 0 \leq f_{uv} \leq c_{uv}$$

$$\forall (u, v) \in E$$

$$\sum_{w:(w,u) \in E} f_{wu} - \sum_{v:(u,v) \in E} f_{uv} \geq 0 \quad \forall u \in V \setminus \{s, t\}$$



Linear Programming (LP)

General form:

$$\begin{array}{ll}
 \min & \mathbf{c}^T \mathbf{x} \\
 \text{s.t.} & \mathbf{a}_i^T \mathbf{x} = b_i & i \in M \\
 & \mathbf{a}_i^T \mathbf{x} \geq b_i & i \in \bar{M} \\
 & x_j \geq 0 & j \in N \\
 & x_j \text{ unconstrained} & j \in \bar{N}
 \end{array}$$



Canonical form:

$$\begin{array}{ll}
 \min & \mathbf{c}^T \mathbf{x} \\
 \text{s.t.} & \mathbf{A} \mathbf{x} \geq \mathbf{b} \\
 & \mathbf{x} \geq \mathbf{0}
 \end{array}$$

$$\mathbf{a}_i^T \mathbf{x} = b_i \implies \begin{cases} \mathbf{a}_i^T \mathbf{x} \geq b_i \\ -\mathbf{a}_i^T \mathbf{x} \geq -b_i \end{cases}$$

$$x_j \text{ unconstrained} \implies x_j = x_j^+ - x_j^- \text{ where } \begin{cases} x_j^+ \geq 0 \\ x_j^- \geq 0 \end{cases}$$

Solvable in Polynomial Time

Canonical Form of Linear programming

$$\begin{array}{ll}\min & \mathbf{c}^T \mathbf{x} \\ \text{s.t.} & \mathbf{Ax} \geq \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0}\end{array}$$

Algorithm	Theory	Practice
Simplex Method	Exponential Time	Works Well
Ellipsoid Method	Polynomial Time	Slow
Internal Point Methods	Polynomial Time	Works Well

Convex Polytopes

- **hyperplane:**

subspace of dimension $n - 1$

$$\sum_{j=1}^n a_j x_j = b$$

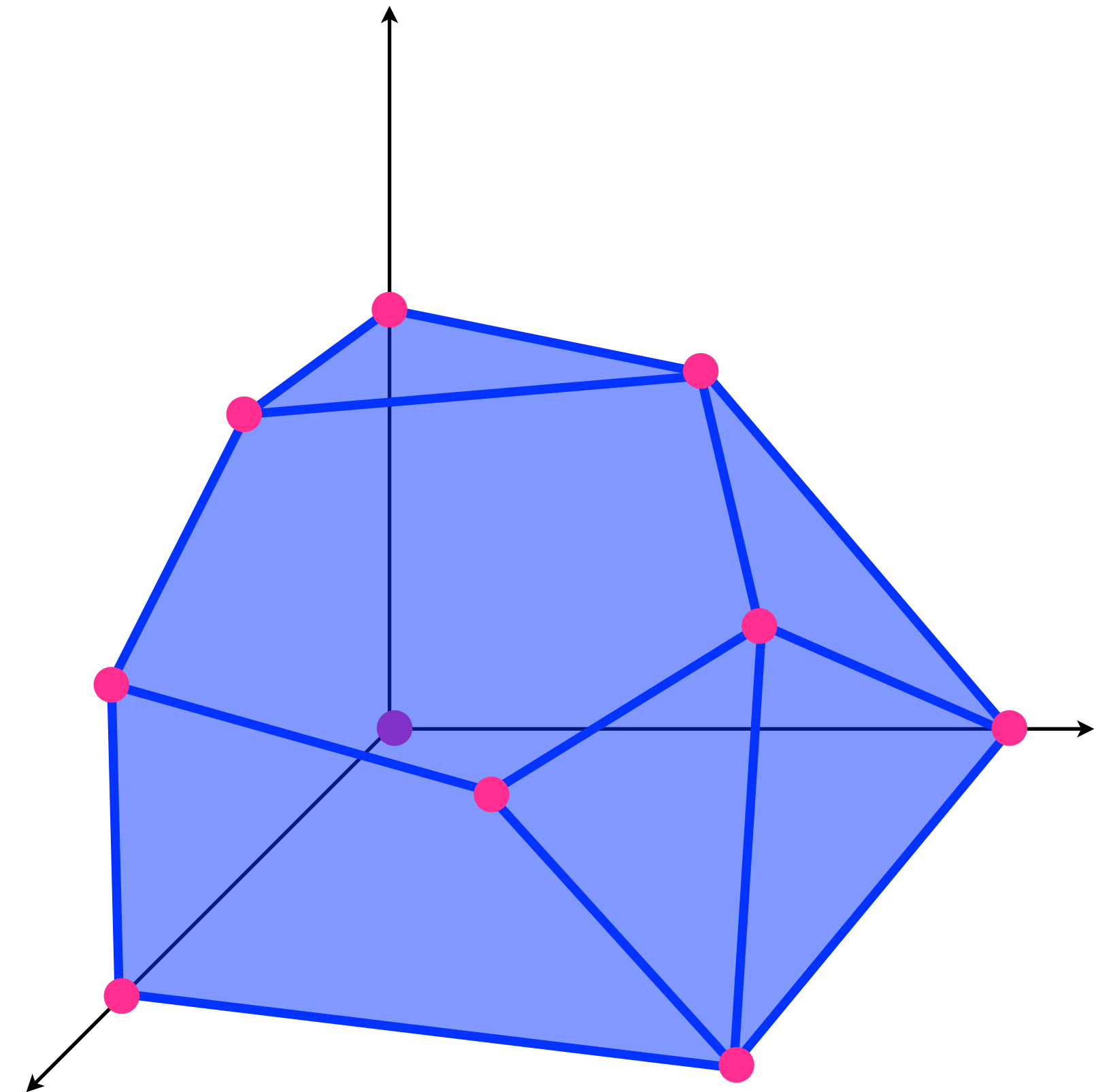
- *(closed, affine)* **halfspace:**

$$\sum_{j=1}^n a_j x_j \geq b$$

- **convex polyhedron:**

intersection of finitely many halfspaces

- **convex polytope:** bounded convex polyhedron



Convex Polyhedron

- A set $S \subseteq \mathbb{R}^n$ is **convex** if $\lambda x + (1 - \lambda)y \in S$ for all $x, y \in S$ and $\lambda \in [0,1]$. A **convex body** is a compact convex set.
- The **convex hull** of a set $S \subseteq \mathbb{R}^n$ is the smallest convex set $\supseteq S$.
- **Affine subspace**: $\{x \in \mathbb{R}^n \mid c^T x \geq b\}$ for $c \in \mathbb{R}^n \setminus \{0\}$ and $b \in \mathbb{R}$.
- **Convex polyhedron**: $\{x \in \mathbb{R}^n \mid Ax \geq b\}$ for $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$.
- **Polytope**: convex hull of a finite $V \subseteq \mathbb{R}^n \iff$ bounded polyhedron
- **Vertex**: point x in a convex polyhedron $P = \{x \in \mathbb{R}^n \mid Ax \geq b\}$ such that $\exists c \in \mathbb{R}^n$ s.t. $c^T x < c^T y$ for all $y \in P$ with $y \neq x$
- **Extreme point**: a point x in a convex polyhedron P that cannot be expressed as a convex combination of any other $y, z \in P$

Convex Polyhedron

Proposition (existence of vertex):

A convex polyhedron P has a vertex iff it does not contain a line, i.e. there're no $x, y \in \mathbb{R}^n$ s.t. $x + \lambda y \in P$ for all $\lambda \in \mathbb{R}$.

Proposition (vertex = extreme point):

For nonempty convex polyhedron P , $\forall x \in P$:
 x is a vertex $\iff x$ is an extreme point

Proposition (vertex as optimal solution):

If the convex polyhedron $\{x \in \mathbb{R}^n \mid Ax \geq b\}$ has a vertex and the LP $\min\{c^T x \mid Ax \geq b\}$ has an optimal solution, then there is an optimal solution that is a vertex.

Linear Programming (LP)

Canonical form:

$$\begin{array}{ll} \min & c^T x \\ \text{s.t.} & Ax \geq b \\ & x \geq 0 \end{array}$$



Standard form:

$$\begin{array}{ll} \min & c^T x \\ \text{s.t.} & Ax = b \\ & x \geq 0 \end{array}$$

$$a_i^T x \leq b_i \Rightarrow \begin{cases} a_i^T x + s_i = b_i \\ s_i \geq 0 \end{cases}$$

slack variable

$$A \Rightarrow A' = \begin{bmatrix} A & I \end{bmatrix}$$

Basic Feasible Solutions (*bfs*)

Standard form:

$$\begin{array}{ll} \min & \mathbf{c}^T \mathbf{x} \\ \text{s.t.} & A\mathbf{x} = \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0} \end{array}$$

WLOG:

$$A \in \mathbb{R}^{m \times n}$$

$$\mathbf{b} \in \mathbb{R}^m \quad \mathbf{c} \in \mathbb{R}^n$$

$$m = \text{rank}(A) \leq n$$

- **Basis** $B \subseteq [n]$: a set of m linearly independent columns of A
- **Basic solution** $\mathbf{x} \in \mathbb{R}^n$: $A_{[m] \times B} \mathbf{x}_B = \mathbf{b}$ and $\mathbf{x}_{[n] \setminus B} = \mathbf{0}$
 - $\implies A\mathbf{x} = \mathbf{b}$ but not necessarily $\mathbf{x} \geq \mathbf{0}$
- **Basic feasible solution (*bfs*)**: a basic solution satisfying $\mathbf{x} \geq \mathbf{0}$

\mathbf{x} is a *bfs* of the LP $\min\{\mathbf{c}^T \mathbf{x} \mid A\mathbf{x} = \mathbf{b} \wedge \mathbf{x} \geq \mathbf{0}\} \iff$
 \mathbf{x} is a vertex of the polyhedron $\{\mathbf{x} \mid A\mathbf{x} = \mathbf{b} \wedge \mathbf{x} \geq \mathbf{0}\}$

The Simplex Algorithm

Standard form:

$$\begin{array}{ll}\min & \mathbf{c}^T \mathbf{x} \\ \text{s.t.} & \mathbf{Ax} = \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0}\end{array}$$

WLOG:

$$\mathbf{A} \in \mathbb{R}^{m \times n}$$

$$\mathbf{b} \in \mathbb{R}^m \quad \mathbf{c} \in \mathbb{R}^n$$

$$m = \text{rank}(\mathbf{A}) \leq n$$

- Two *bfs*'s are neighbors if their bases share $m - 1$ columns of \mathbf{A}

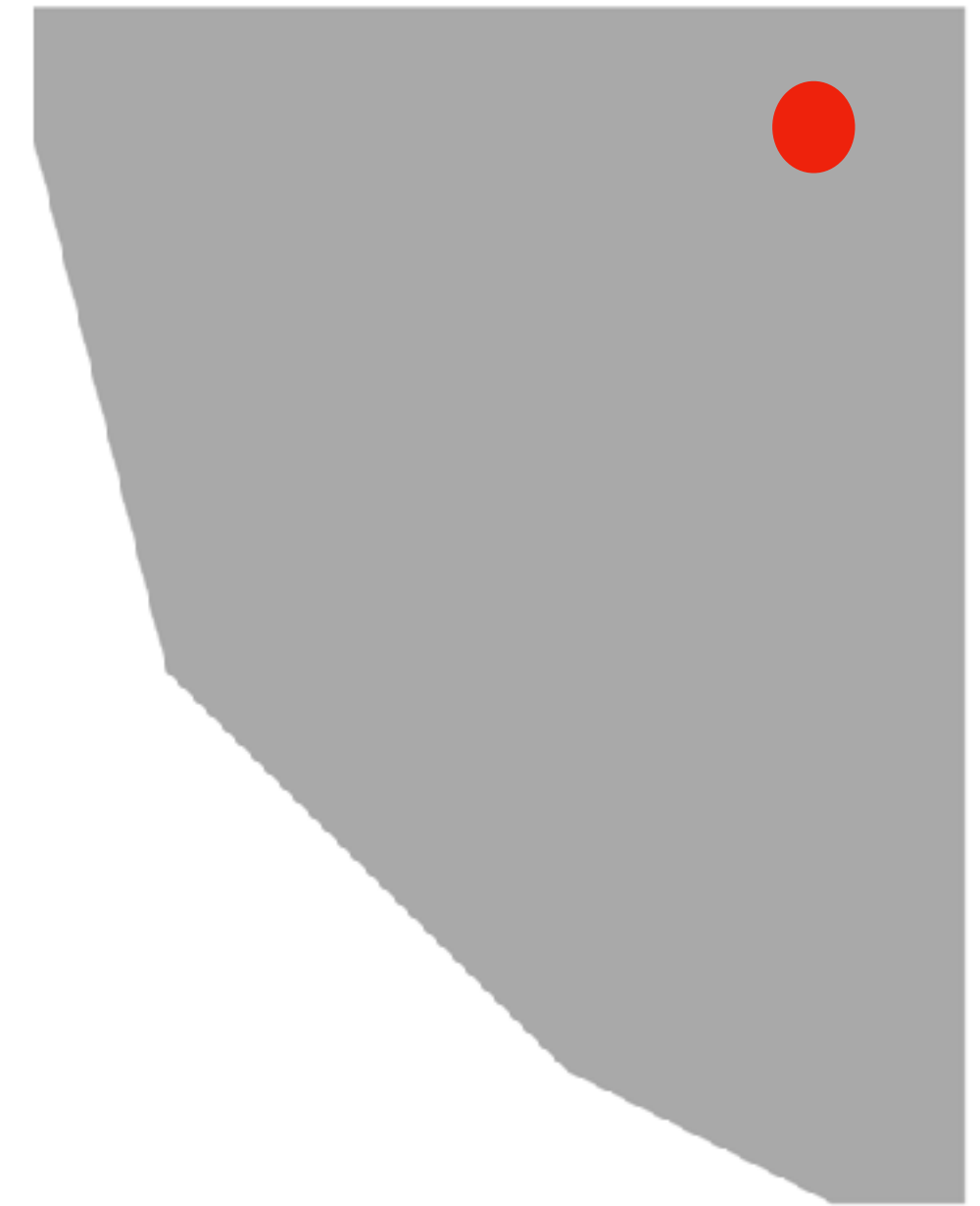
Simplex Algorithm (Dantzig 1947):

start at a *bfs* \mathbf{x} ;

while \exists a neighboring *bfs* \mathbf{x}' with $\mathbf{c}^T \mathbf{x}' < \mathbf{c}^T \mathbf{x}$:

move to one of such \mathbf{x}' ;

- Stops at a *local* optima \implies a *global* optima (by convexity)



Linear Programming (LP) Solvers

$$\min \quad c^T x$$

$$\text{s.t.} \quad Ax = b$$

$$x \geq 0$$

$$A \in \mathbb{R}^{m \times n}$$

$$b \in \mathbb{R}^m \quad c \in \mathbb{R}^n$$

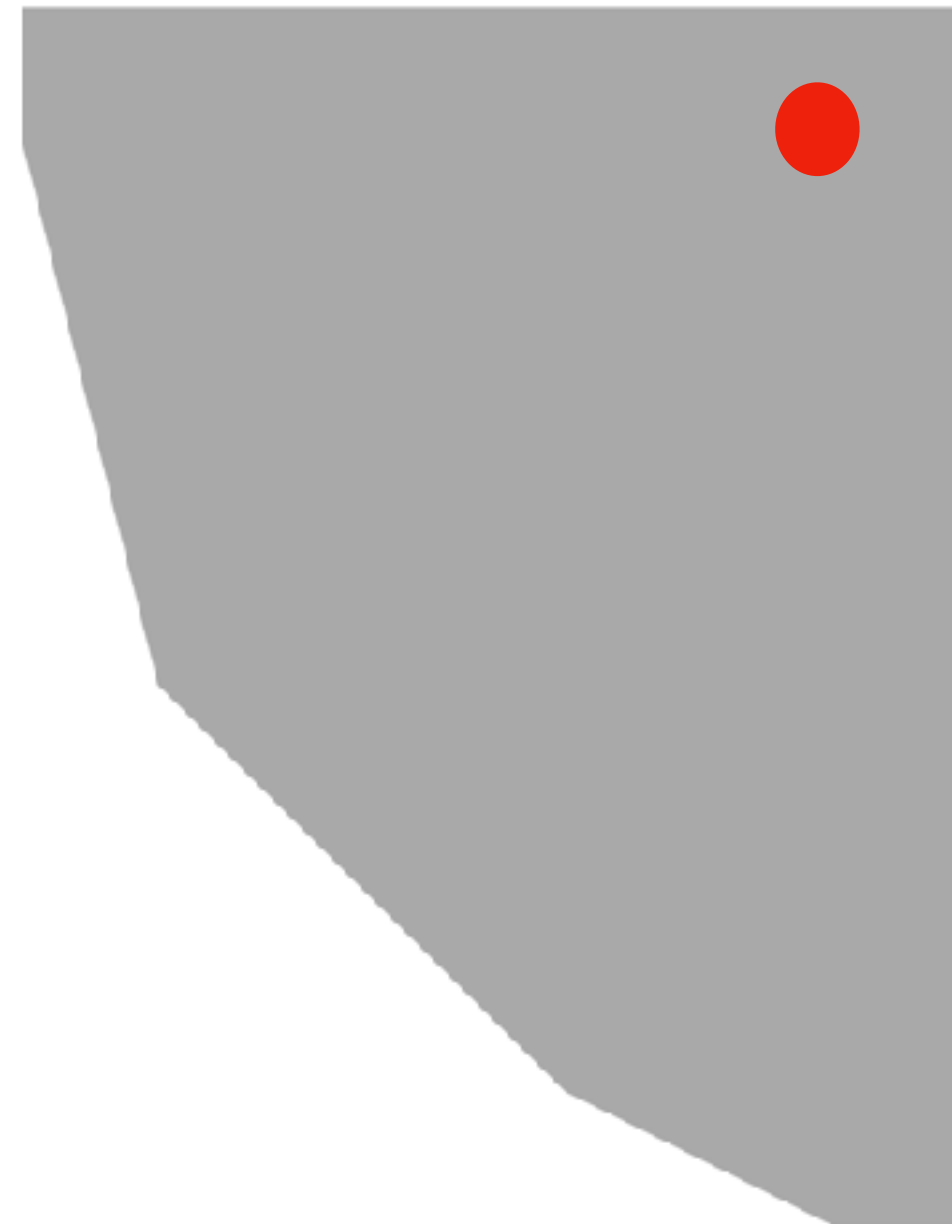
$$m = \text{rank}(A) \leq n$$

- Dantzig's simplex method [Dantzig '47]:
 - walks over polytope vertices along polytope edges
 - exponential time in the worst case (Klee–Minty cube, 1972)
 - poly-time in *smoothed* complexity [Spielman-Teng'01]
- Solvable in (*weakly*) polynomial time:
 - ellipsoid method [Khachiyan '80] in $O(n^6)$ time
 - interior-point methods [Karmarkar '84] in $O(n^{2.5})$ time [Vaidya '89] and recently, in current matrix multiplication time [Cohen, Lee, Song '19] [Jiang, Song, Weinstein, Zhang '21]

Interior Point Method

Interior Point Method (Karmarkar 1984):

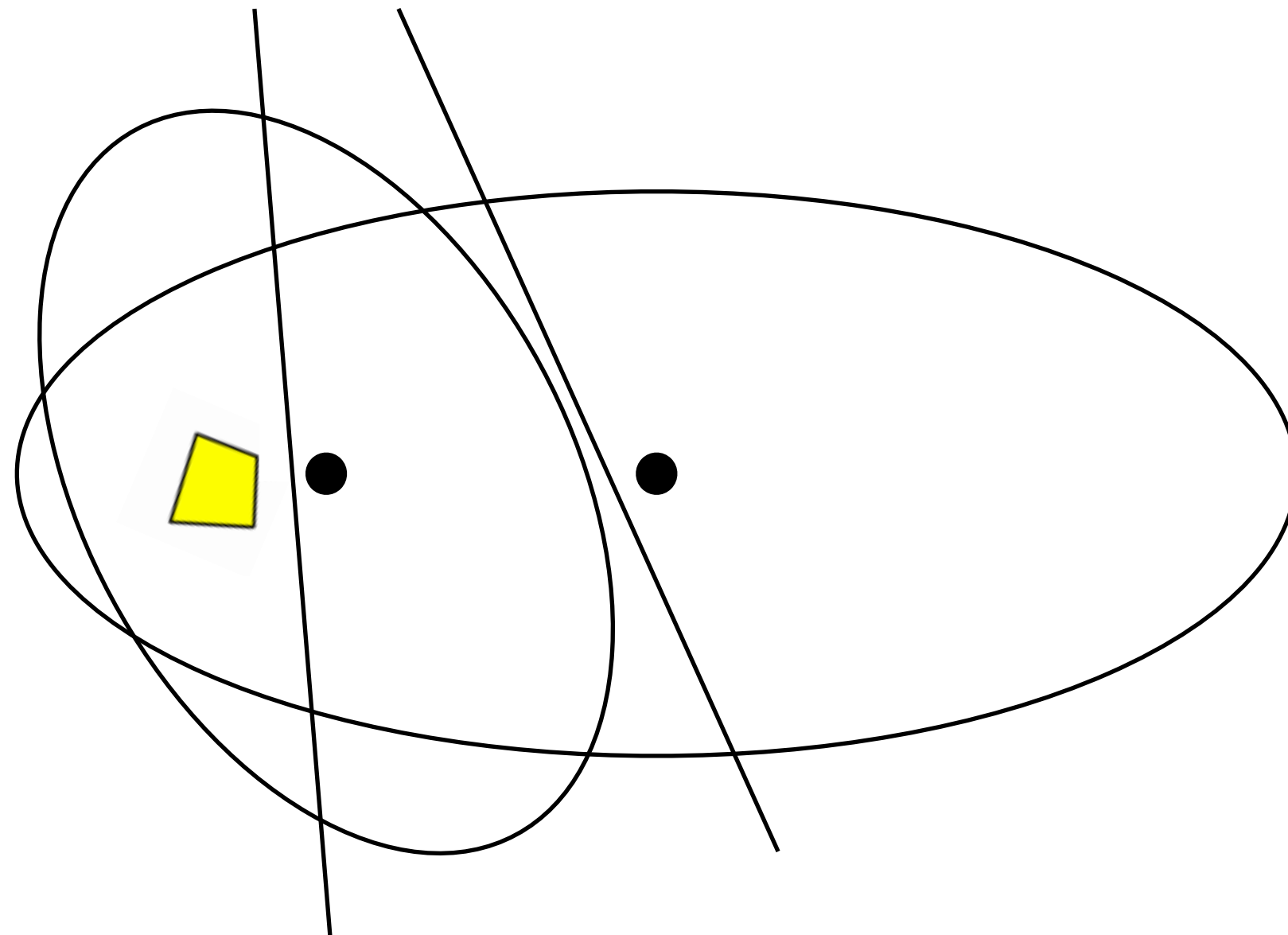
- keep the solution inside the polytope
- design penalty function so that the solution is not too close to the boundary
- the final solution will be arbitrarily close to the optimum solution



Ellipsoid Method

Ellipsoid Method (Khachiyan 1979):

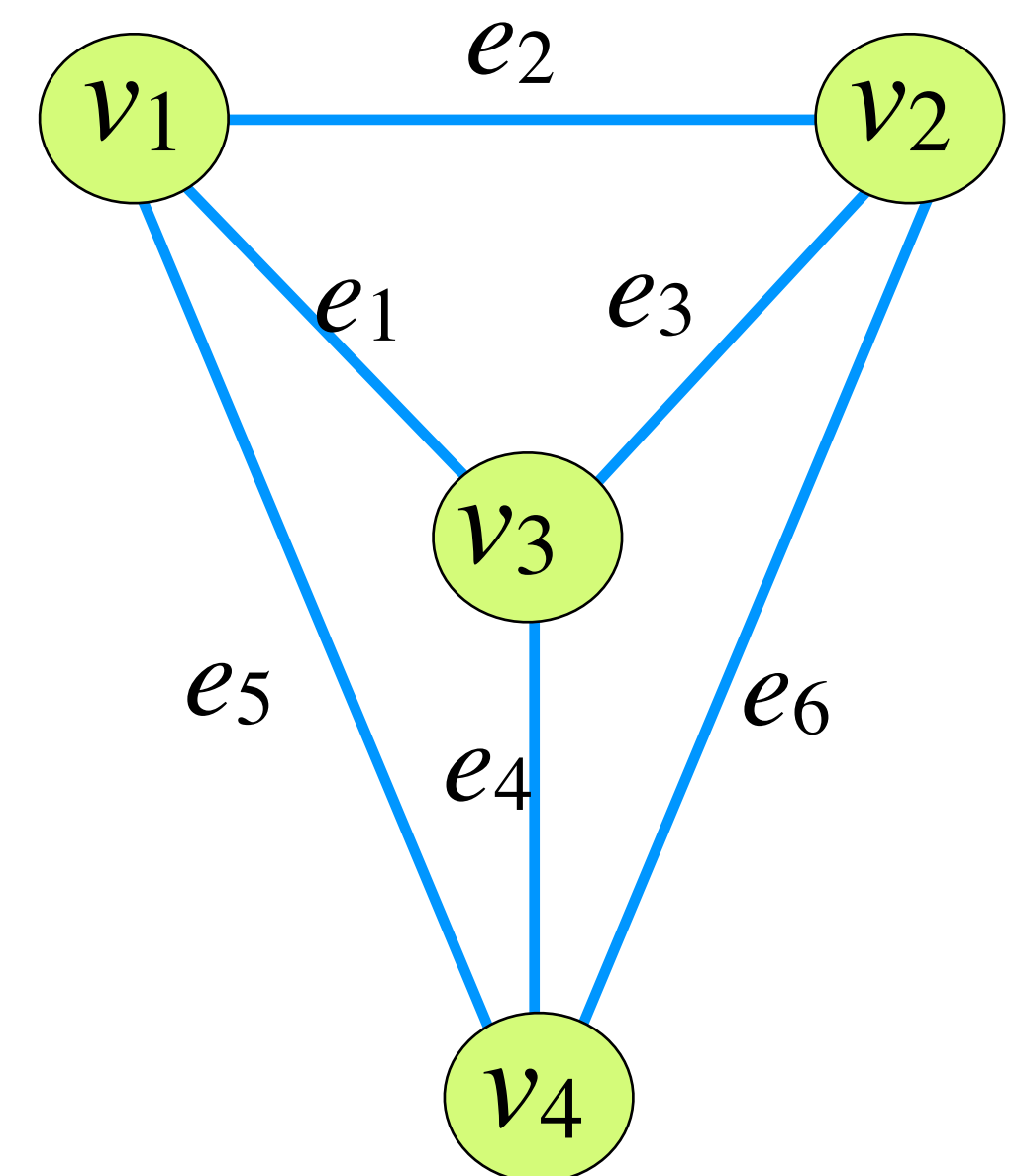
- maintain an ellipsoid that contains the feasible region
- cut the ellipsoid in half, find smaller ellipsoid to enclose the half-ellipsoid, and repeat



Applications of Linear Programming

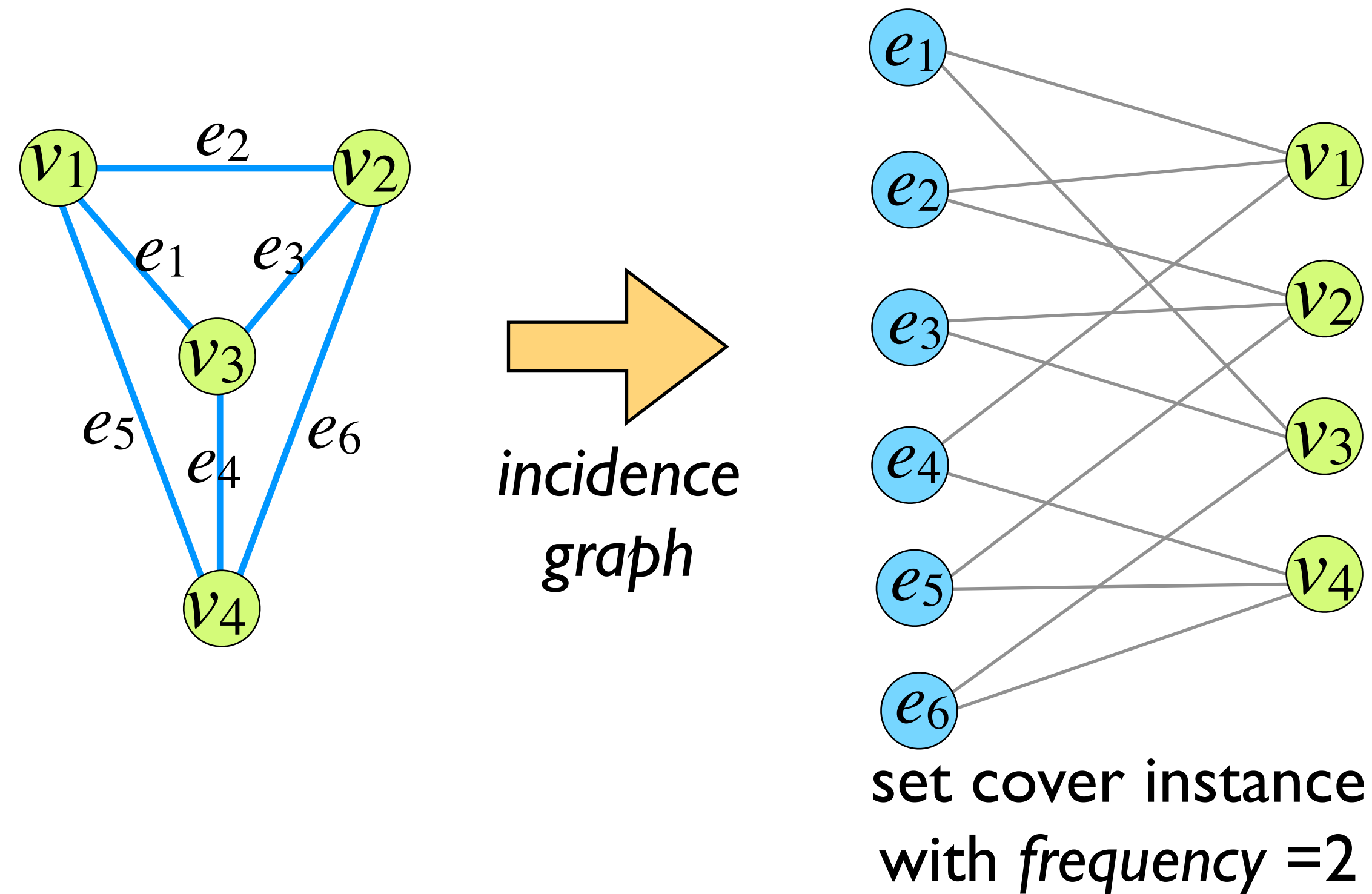
- Domain: computer science, mathematics, operations research, economics
- Types of problems: transportation, scheduling, clustering, network routing, resource allocation, facility location
- Research directions:
 - polynomial time exact algorithm
 - polynomial time approximation algorithm
 - sub-routines for the branch-and-bound method for integer programming
 - other algorithmic models: online algorithm, distributed algorithms, dynamic algorithms, fast algorithms

Vertex Cover



Vertex Cover

Instance: An undirected graph $G(V, E)$.
Find the smallest $C \subseteq V$ that intersects all edges.



Vertex Cover

Instance: An undirected graph $G(V, E)$.
Find the smallest $C \subseteq V$ that intersects all edges.

- **NP**-hard
- $\ln(n)$ -approximation by greedy set cover
- 2-approximation algorithm:

Find a *maximal matching*;
return the *matched* vertices;

- [Khot, Regev 2008] Assuming the *unique games conjecture*, there is no poly-time $(2 - \epsilon)$ -approximation algorithm.

Vertex Cover

Instance: An undirected graph $G(V, E)$.
Find the smallest $C \subseteq V$ that intersects all edges.

- Integer Linear Program (ILP) for vertex cover:

$$\begin{array}{ll} \text{minimize} & \sum_{v \in V} x_v \quad \text{linear objective function} \\ \text{subject to} & \sum_{v \in e} x_v \geq 1, \quad e \in E \quad \text{linear constraints} \\ & x_v \in \{0, 1\}, \quad v \in V \quad \text{integer domains} \end{array}$$

- Solving integer linear program is **NP**-hard.

Vertex Cover

Instance: An undirected graph $G(V, E)$.
Find the smallest $C \subseteq V$ that intersects all edges.

- Linear Program (LP) *relaxation*:

$$\begin{array}{ll} \text{minimize} & \sum_{v \in V} x_v \\ \text{subject to} & \sum_{v \in e} x_v \geq 1, \quad e \in E \\ & x_v \in [0, 1], \quad v \in V \end{array}$$

*fractional
domains*

- linear programs are solvable in polynomial time!

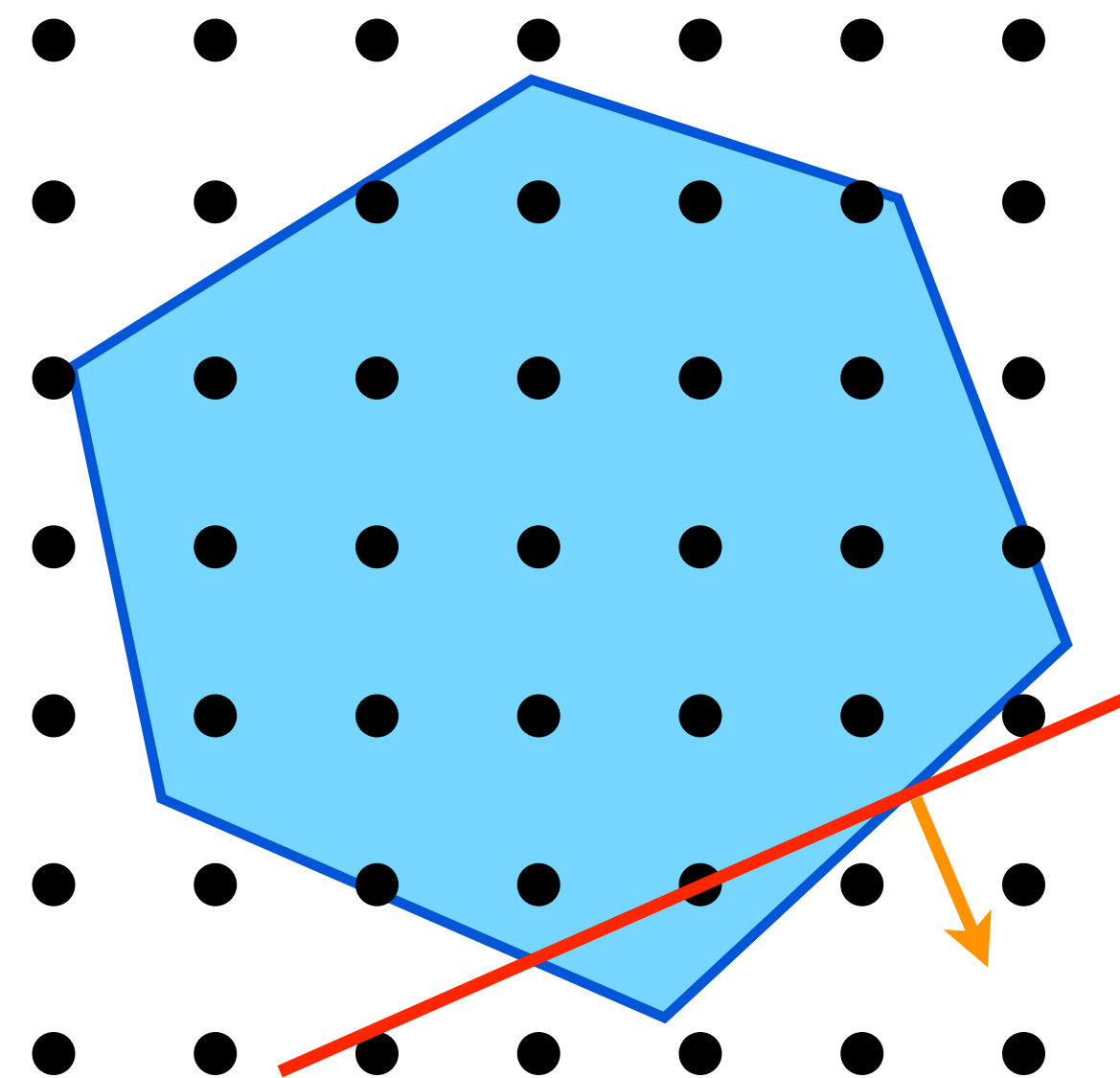
Integrality

$$\min \quad c^T x$$

$$\text{s.t.} \quad Ax \geq b$$

~~$$x \in \mathbb{Z}^n$$~~

LP-relaxation



Vertex Cover

Instance: An undirected graph $G(V, E)$.
Find the smallest $C \subseteq V$ that intersects all edges.

- Integer Linear Program (ILP) for vertex cover:

$$\begin{array}{ll} \text{minimize} & \sum_{v \in V} x_v \\ \text{subject to} & \sum_{v \in e} x_v \geq 1, \quad e \in E \\ & x_v \in \{0, 1\}, \quad v \in V \end{array}$$

- **LP relaxation** for Minimum vertex cover of $G(V, E)$

$$\begin{array}{ll}\text{minimize} & \sum_{v \in V} x_v \\ \text{subject to} & \sum_{v \in e} x_v \geq 1, \quad e \in E \\ & x_v \in [0, 1], \quad v \in V\end{array}$$

LP Relaxation & Rounding:

find **optimal** solution $x^* \in [0, 1]^V$ of LP relaxation;
round x^* to an **feasible integral** solution $\hat{x} \in \{0, 1\}^V$:

$$\hat{x}_v = \begin{cases} 1 & \text{if } x_v^* \geq 0.5 \\ 0 & \text{otherwise} \end{cases}$$

$$\begin{array}{ll}
\min & \sum_{v \in V} x_v \\
\text{s.t.} & \sum_{v \in e} x_v \geq 1, \quad e \in E \\
& x_v \in [0,1], \quad v \in V
\end{array}$$

LP Relax & Round:

find **OPT** $x^* \in [0,1]^V$;

round x^* to **feasible integral** \hat{x} :

$$\hat{x}_v = \left\{ \begin{array}{ll} 1 & \text{if } x_v^* \geq 0.5 \\ 0 & \text{otherwise} \end{array} \right\} \leq 2x_v^*$$

- Soundness of rounded solution \hat{x} (as a **vertex cover**):

$$\sum_{v \in e} x_v^* \geq 1 \implies \sum_{v \in e} \hat{x}_v \geq 1$$

- Approximation ratio:

$$OPT = OPT_{Int} \geq OPT_{LP} = \sum_{v \in V} x_v^*$$

$$SOL = \sum_{v \in V} \hat{x}_v \leq 2 \sum_{v \in V} x_v^* \leq 2OPT$$

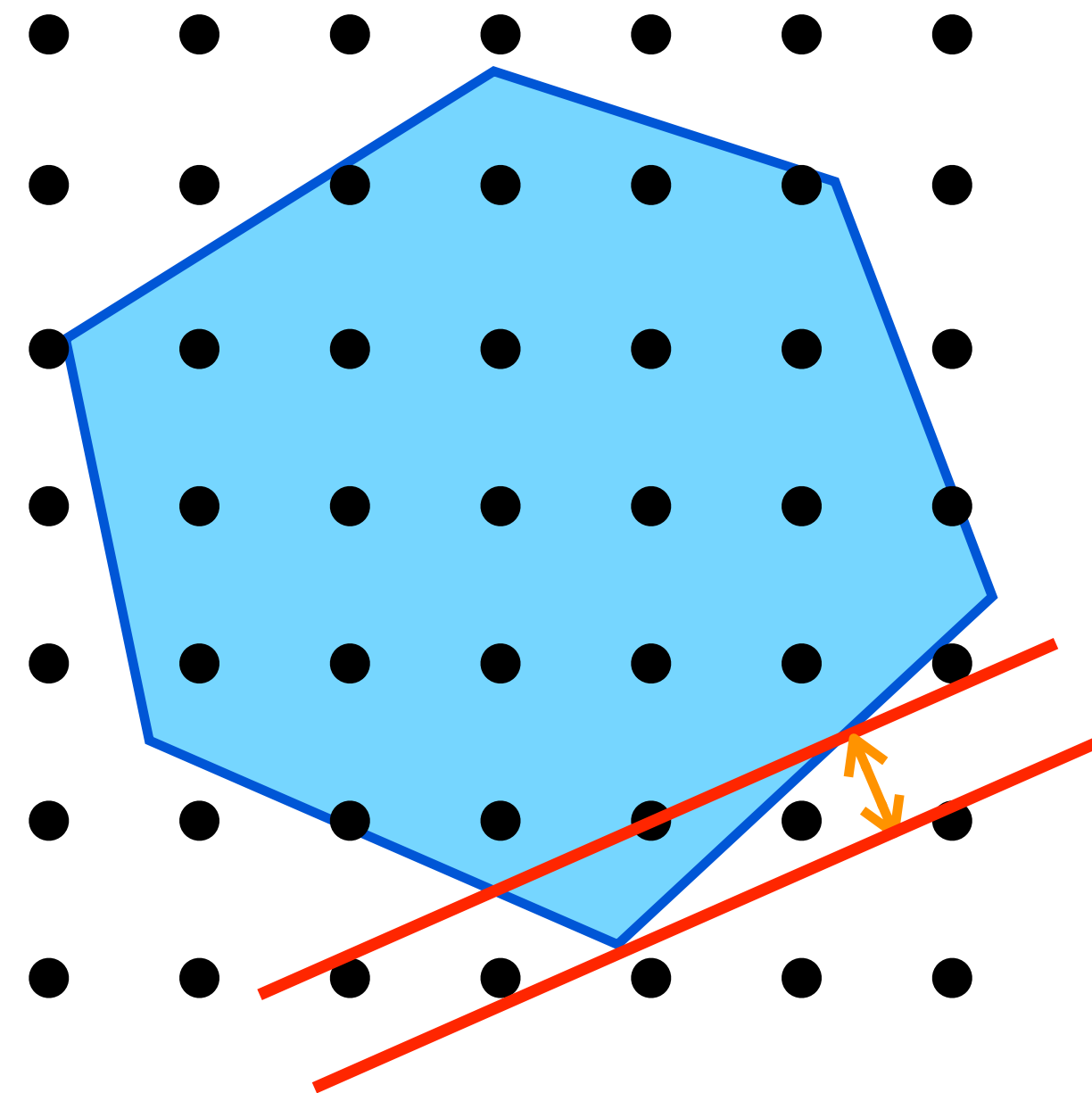
LP Relaxation & Rounding

- **Modeling**: Express the optimization problem as an Integer Linear Program (ILP).
- **Relaxation**: Relax the ILP to a Linear Program (LP).
- **Solving**: Find the *optimal* solution by an efficient LP solver.
- **Rounding**: Round the optimal solution to a *feasible integral* solution.
- **Analysis**: Prove that the rounded solution is not too far away from the *optimal integral* solution (**usually by comparing with the optimal solution**).

Integrality Gap

$$\begin{array}{ll} \min & c^T x \\ \text{s.t.} & Ax \geq b \end{array}$$

$$\cancel{x \in \mathbb{Z}^n}$$



$$\text{integrality gap} = \sup_I \frac{\text{OPT}(I)}{\text{OPT}_{\text{LP}}(I)}$$

Integrality Gap

- minimum vertex cover of $G(V, E)$:

$$\begin{array}{ll}\text{minimize} & \sum_{v \in V} x_v \\ \text{subject to} & \sum_{v \in e} x_v \geq 1, \quad e \in E \\ & x_v \in \{0, 1\}, \quad v \in V\end{array}$$

$$\text{integrality gap} = \sup_I \frac{\text{OPT}(I)}{\text{OPT}_{\text{LP}}(I)}$$

- The 2-approx. LP-rounding algorithm shows integrality gap ≤ 2

Because the analysis compares the relaxed OPT with an integral feasible solution (output of the algorithm)

Integrality Gap

- minimum vertex cover of $G(V, E)$:

$$\begin{array}{ll} \text{minimize} & \sum_{v \in V} x_v \\ \text{subject to} & \sum_{v \in e} x_v \geq 1, \quad e \in E \\ & x_v \in \{0, 1\}, \quad v \in V \end{array}$$

$$\text{integrality gap} = \sup_I \frac{\text{OPT}(I)}{\text{OPT}_{\text{LP}}(I)}$$

- For LP relaxation of vertex cover: integrality gap = 2
- [Singh '19] int. gap on $G = \left(2 - \frac{2}{\chi^f(G)}\right)$ fractional chromatic number

MAX-SAT

$(x \text{ OR } y \text{ OR } z) \text{ AND } (x \text{ OR } \bar{y} \text{ OR } z) \text{ AND}$
 $(x \text{ OR } y \text{ OR } \bar{z}) \text{ AND } (x \text{ OR } \bar{y} \text{ OR } \bar{z}) \text{ AND}$
 $(\bar{x} \text{ OR } y \text{ OR } z) \text{ AND } (\bar{x} \text{ OR } \bar{y} \text{ OR } \bar{z})$

Max-SAT

Instance: A CNF formula $\Phi = C_1 \wedge C_2 \wedge \dots \wedge C_m$.

Find an assignment $x \in \{T, F\}^n$ that maximizes the number of satisfied clauses.

- **CNF (Conjunctive Normal Form):** conjunction (\wedge) of *clauses*

$$\Phi = (x_1 \vee \neg x_2 \vee x_3) \wedge (x_1 \vee x_2 \vee x_4) \wedge (x_3 \vee \neg x_4 \vee \neg x_5)$$

- **Boolean variables:** $x_1, x_2, \dots, x_n \in \{T, F\}$
- **Clause:** disjunction (\vee) of *literals*
- **literal:** x_i or $\neg x_i$
- **Max-SAT:** **NP**-hard

Random Assignment

Instance: A CNF formula $\Phi = C_1 \wedge C_2 \wedge \dots \wedge C_m$.

Find an assignment $x \in \{T, F\}^n$ that maximizes the number of satisfied clauses.

Random assignment:

each x_i is assigned a value from $\{T, F\}$ uniformly and independently at random

- A clause $C_j = (\ell_1 \vee \dots \vee \ell_k)$ of k_j literals:

$$\Pr[C_j \text{ is satisfied}] = 1 - 2^{-k_j} \geq \frac{1}{2}$$

$$\mathbb{E} [\# \text{ of satisfied clauses}] = \sum_{j=1}^m \Pr[C_j \text{ is satisfied}] \geq \frac{m}{2} \geq \frac{1}{2} OPT$$

Integer Program

Instance: A CNF formula $\Phi = C_1 \wedge C_2 \wedge \dots \wedge C_m$.

$$\Phi = (x_1 \vee \neg x_2 \vee x_3) \wedge (x_1 \vee x_2 \vee x_4) \wedge (x_3 \vee \neg x_4 \vee \neg x_5)$$

- **Boolean variables:** $x_1, \dots, x_n \in \{0, 1\}$ $x_i = 1 \iff x_i = \text{T}$
- **Clauses:** $C_j = (\ell_1 \vee \dots \vee \ell_k) = \bigvee_{i \in S_j^+} x_i \vee \bigvee_{i \in S_j^-} \neg x_i$
- $S_j^+ = \left\{ i \mid x_i \text{ appears in } C_j \right\}$
- $S_j^- = \left\{ i \mid \neg x_i \text{ appears in } C_j \right\}$

$$C_j \text{ is satisfied} \iff \sum_{i \in S_j^+} x_i + \sum_{i \in S_j^-} (1 - x_i) \geq 1$$

Integer Program

Instance: Clauses C_1, \dots, C_m , where for each clause C_j :

$$S_j^+ = \left\{ i \mid x_i \text{ appears in } C_j \right\}, S_j^- = \left\{ i \mid \neg x_i \text{ appears in } C_j \right\}$$

$$\begin{array}{ll} \text{maximize} & \sum_{j=1}^m y_j \\ \text{subject to} & \sum_{i \in S_j^+} x_i + \sum_{i \in S_j^-} (1 - x_i) \geq y_j \quad 1 \leq j \leq m \\ & x_i \in \{0,1\} \quad 1 \leq i \leq n \\ & y_j \in \{0,1\} \quad 1 \leq j \leq m \end{array}$$

LP Relaxation

Instance: Clauses C_1, \dots, C_m , where for each clause C_j :

$$S_j^+ = \left\{ i \mid x_i \text{ appears in } C_j \right\}, \quad S_j^- = \left\{ i \mid \neg x_i \text{ appears in } C_j \right\}$$

$$\begin{aligned} &\text{maximize} && \sum_{j=1}^m y_j \\ &\text{subject to} && \sum_{i \in S_j^+} x_i + \sum_{i \in S_j^-} (1 - x_i) \geq y_j && 1 \leq j \leq m \\ &&& x_i \in [0,1] && 1 \leq i \leq n \\ &&& y_j \in [0,1] && 1 \leq j \leq m \end{aligned}$$

Linear Randomized Rounding

Instance: Clauses C_1, \dots, C_m , where for each clause C_j :

$$S_j^+ = \{i \mid x_i \text{ appears in } C_j\}, S_j^- = \{i \mid \neg x_i \text{ appears in } C_j\}$$

$$\begin{aligned} &\text{maximize} && \sum_{j=1}^m y_j \\ &\text{subject to} && \sum_{i \in S_j^+} x_i + \sum_{i \in S_j^-} (1 - x_i) \geq y_j \quad 1 \leq j \leq m \\ &&& x_i, y_j \in [0,1] \quad \forall i, j \end{aligned}$$

Optimal solution: $\mathbf{x}^* \in [0,1]^n$, $\mathbf{y}^* \in [0,1]^m$

Linear rounding: $\hat{x}_i = \begin{cases} 1 & \text{with prob. } x_i^* \\ 0 & \text{with prob. } 1 - x_i^* \end{cases}$ (independently)

$$\begin{array}{ll}
\text{maximize} & \sum_{j=1}^m y_j \\
\text{subject to} & \sum_{i \in S_j^+} x_i^* + \sum_{i \in S_j^-} (1 - x_i^*) \geq y_j^* \quad 1 \leq j \leq m \\
& x_i, y_j \in [0, 1] \quad \forall i, j
\end{array}$$

Optimal solution: $\mathbf{x}^* \in [0, 1]^n$, $\mathbf{y}^* \in [0, 1]^m$

Linear rounding: $\hat{x}_i = \begin{cases} 1 & \text{with prob. } x_i^* \\ 0 & \text{with prob. } 1 - x_i^* \end{cases}$ (independently)

- $OPT = OPT_{Int} \leq OPT_{LP}$

$$= 1 - \prod_{i \in S_j^+} (1 - x_i^*) \prod_{i \in S_j^-} x_i^* \geq 1 - \left(1 - y_j^*/k_j\right)^{k_j}$$

(C_j has k_j literals)

Jenssen's inequality

$$\begin{aligned}
&\geq \left[1 - \left(1 - 1/k_j\right)^{k_j}\right] y_j^* \\
&\geq (1 - 1/e) y_j^*
\end{aligned}$$

AM-GM

$$\begin{aligned}
& \text{maximize} && \sum_{j=1}^m y_j \\
& \text{subject to} && \sum_{i \in S_j^+} x_i + \sum_{i \in S_j^-} (1 - x_i) \geq y_j \quad 1 \leq j \leq m \\
& && x_i, y_j \in [0,1] \quad \forall i, j
\end{aligned}$$

Optimal solution: $\mathbf{x}^* \in [0,1]^n$, $\mathbf{y}^* \in [0,1]^m$

Linear rounding: $\hat{x}_i = \begin{cases} 1 & \text{with prob. } x_i^* \\ 0 & \text{with prob. } 1 - x_i^* \end{cases}$ (independently)

- $OPT = OPT_{Int} \leq OPT_{LP} = \sum_{j=1}^m y_j^*$
- $\Pr[C_j \text{ is satisfied }] = 1 - \prod_{i \in S_j^+} (1 - x_i^*) \prod_{i \in S_j^-} x_i^* \geq (1 - 1/e) y_j^*$

$$\mathbb{E}[SOL] = \sum_{j=1}^m \Pr[C_j \text{ is satisfied }] \geq (1 - 1/e) \sum_{j=1}^m y_j^* \geq (1 - 1/e) OPT$$

$$\begin{array}{ll}
\text{maximize} & \sum_{j=1}^m y_j \\
\text{subject to} & \sum_{i \in S_j^+} x_i + \sum_{i \in S_j^-} (1 - x_i) \geq y_j \quad 1 \leq j \leq m \\
& x_i, y_j \in [0,1] \quad \forall i, j
\end{array}$$

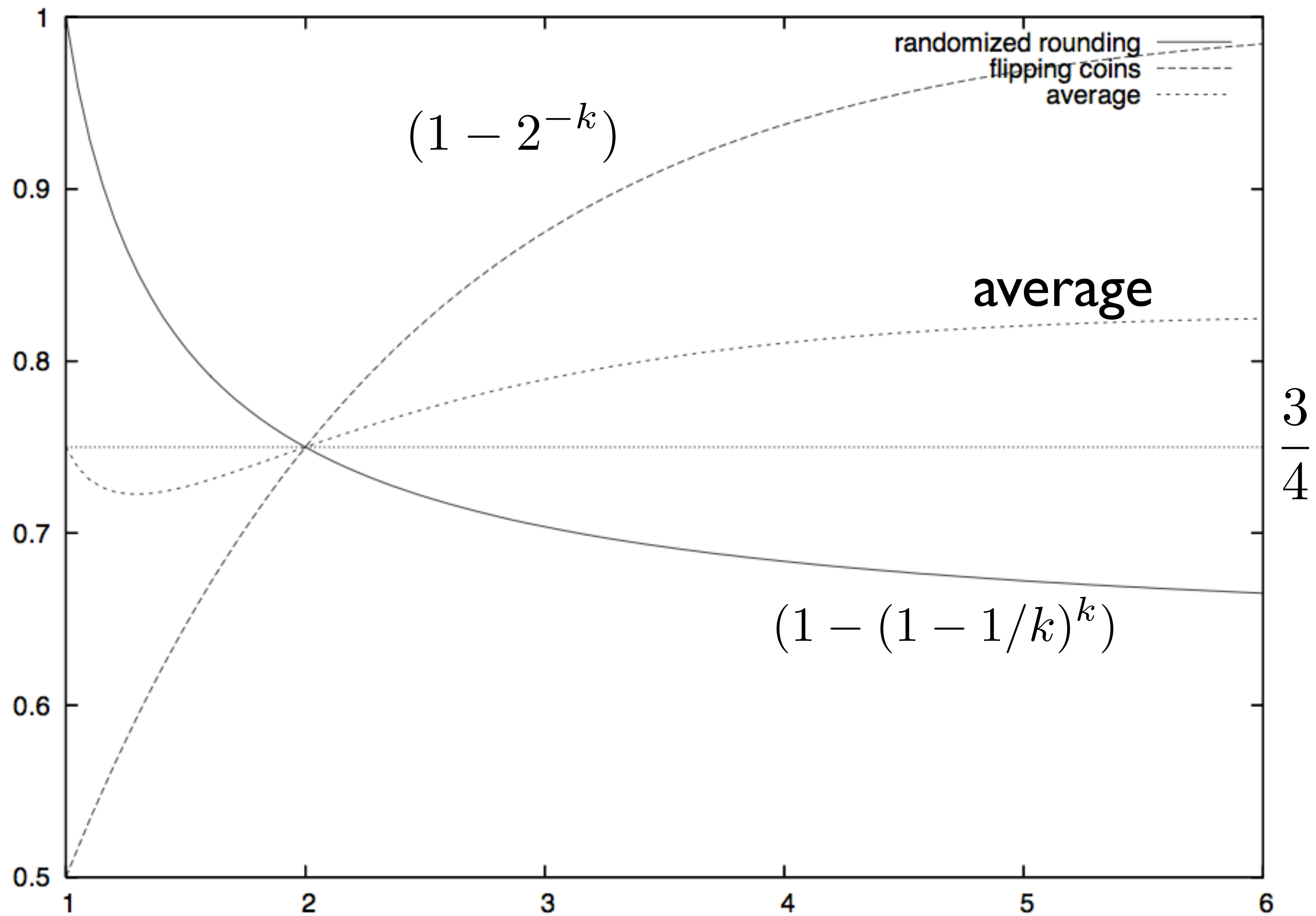
Optimal solution: $\mathbf{x}^* \in [0,1]^n$, $\mathbf{y}^* \in [0,1]^m$

Linear rounding: $\hat{x}_i = \begin{cases} 1 & \text{with prob. } x_i^* \\ 0 & \text{with prob. } 1 - x_i^* \end{cases}$ (independently)

- $\Pr[C_j \text{ is satisfied}] \geq (1 - (1 - 1/k_j)^{k_j}) \cdot y_j^*$

Random assignment: $\hat{x}_i = \begin{cases} 1 & \text{with prob. } 1/2 \\ 0 & \text{with prob. } 1/2 \end{cases}$ (independently)

- $\Pr[C_j \text{ is satisfied}] = 1 - 2^{-k_j} \geq (1 - 2^{-k_j}) \cdot y_j^* \geq (1 - 1/e) \cdot y_j^*$



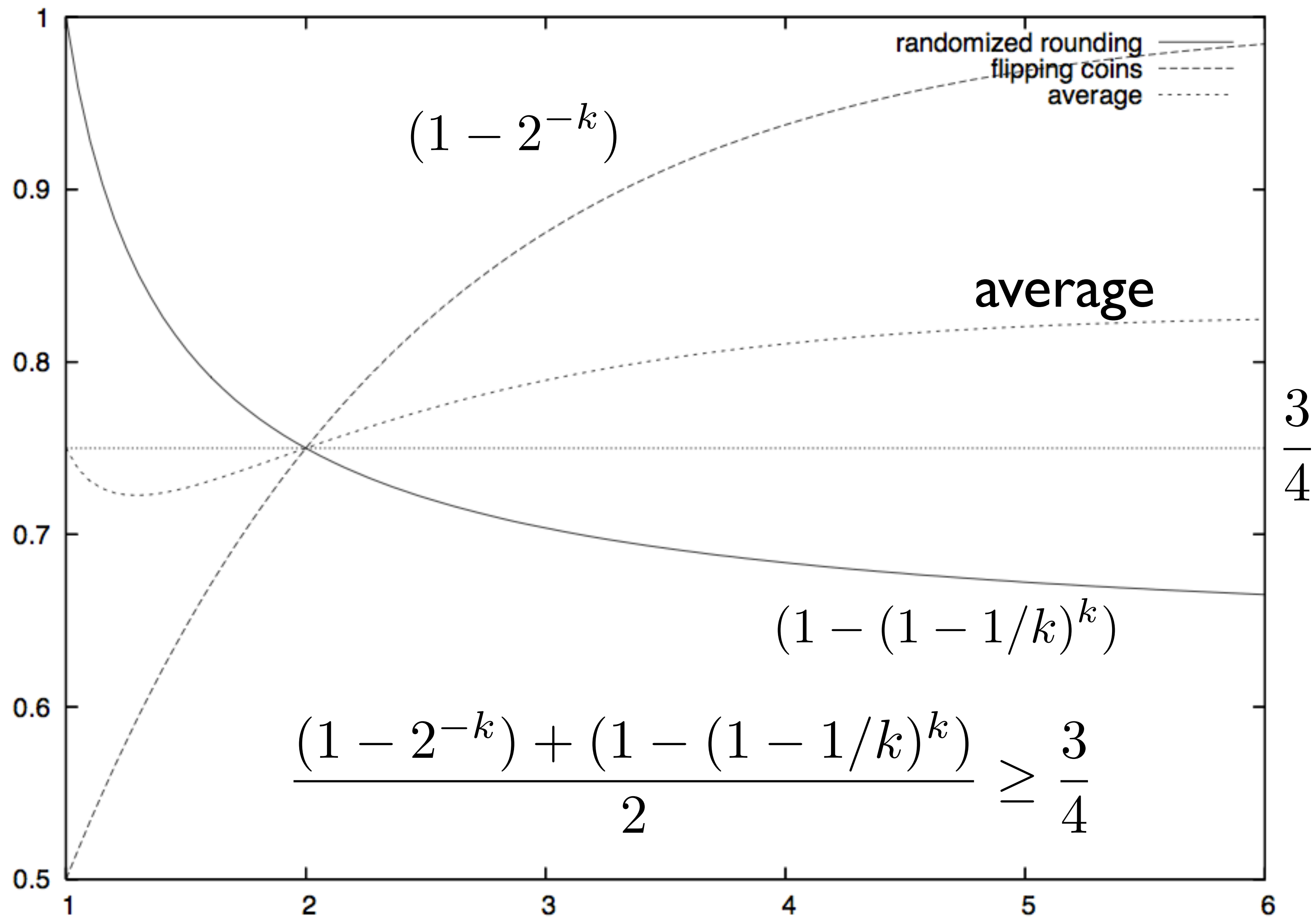
Power of Two Choices

- Use random assignment to satisfy M_1 clauses;
- Use linear rounding of LP relaxation to satisfy M_2 clauses;
- Return the solution with more satisfied clauses.

$$\mathbb{E} \left[\max \{M_1, M_2\} \right] \geq \mathbb{E} \left[\frac{M_1 + M_2}{2} \right]$$

$$\mathbb{E} [M_1] \geq \sum_{j=1}^m \left(1 - 2^{-k_j} \right) \cdot y_j^*$$

$$\mathbb{E} [M_2] \geq \sum_{j=1}^m \left(1 - (1 - 1/k_j)^{k_j} \right) \cdot y_j^*$$



Power of Two Choices

- Use random assignment to satisfy M_1 clauses;
- Use linear rounding of LP relaxation to satisfy M_2 clauses;
- Return the solution with more satisfied clauses.

$$\mathbb{E} \left[\max \{M_1, M_2\} \right] \geq \mathbb{E} \left[\frac{M_1 + M_2}{2} \right] \geq \frac{3}{4} \sum_{j=1}^m y_j^* \geq \frac{3}{4} OPT$$

$$\mathbb{E} [M_1] \geq \sum_{j=1}^m \left(1 - 2^{-k_j} \right) \cdot y_j^*$$

$$\mathbb{E} [M_2] \geq \sum_{j=1}^m \left(1 - (1 - 1/k_j)^{k_j} \right) \cdot y_j^*$$

Max-SAT

Instance: A CNF formula $\Phi = C_1 \wedge C_2 \wedge \dots \wedge C_m$.

Find an assignment $x \in \{T, F\}^n$ that maximizes the number of satisfied clauses.

- Use random assignment to satisfy M_1 clauses;
 - Use linear rounding of LP relaxation to satisfy M_2 clauses;
 - Return the solution with more satisfied clauses.
-
- This combined algorithm outputs a random solution that satisfies $\geq \frac{3}{4}OPT$ clauses in expectation.
 - Can this be achieved by a single algorithm?

Non-Linear Rounding

Instance: Clauses C_1, \dots, C_m , where for each clause C_j :

$$S_j^+ = \left\{ i \mid x_i \text{ appears in } C_j \right\}, \quad S_j^- = \left\{ i \mid \neg x_i \text{ appears in } C_j \right\}$$

$$\begin{aligned} &\text{maximize} && \sum_{j=1}^m y_j \\ &\text{subject to} && \sum_{i \in S_j^+} x_i + \sum_{i \in S_j^-} (1 - x_i) \geq y_j \quad 1 \leq j \leq m \\ &&& x_i, y_j \in [0,1] \quad \forall i, j \end{aligned}$$

Optimal fractional solution: $x^* \in [0,1]^n$, $y^* \in [0,1]^m$

Nonlinear rounding: $\hat{x}_i = \begin{cases} 1 & \text{with prob. } f(x_i^*) \\ 0 & \text{with prob. } 1 - f(x_i^*) \end{cases}$ (independently)

$$\begin{array}{ll}
\text{maximize} & \sum_{j=1}^m y_j \\
\text{subject to} & \sum_{i \in S_j^+} x_i^* + \sum_{i \in S_j^-} (1 - x_i^*) \geq y_j^* \quad 1 \leq j \leq m \\
& x_i, y_j \in [0, 1] \quad \forall i, j
\end{array}$$

Optimal fractional solution: $x^* \in [0, 1]^n$, $y^* \in [0, 1]^m$

Nonlinear rounding: $\hat{x}_i = \begin{cases} 1 & \text{with prob. } f(x_i^*) \\ 0 & \text{with prob. } 1 - f(x_i^*) \end{cases}$ (independently)

$$\Pr[C_j \text{ is unsatisfied}] = \prod_{i \in S_j^+} (1 - f(x_i^*)) \prod_{i \in S_j^-} f(x_i^*)$$

Suppose: for some $c > 1$

$$1 - c^{-x} \leq f(x) \leq c^{x-1}$$

$$\leq c^{-\left(\sum_{i \in S_j^+} x_i^* + \sum_{i \in S_j^-} (1 - x_i^*) \right)}$$

$$\leq c^{-y_j^*}$$

$$\begin{aligned}
& \text{maximize} && \sum_{j=1}^m y_j \\
& \text{subject to} && \sum_{i \in S_j^+} x_i + \sum_{i \in S_j^-} (1 - x_i) \geq y_j \quad 1 \leq j \leq m \\
& && x_i, y_j \in [0,1] \quad \forall i, j
\end{aligned}$$

Optimal fractional solution: $x^* \in [0,1]^n$, $y^* \in [0,1]^m$

Nonlinear rounding: $\hat{x}_i = \begin{cases} 1 & \text{with prob. } f(x_i^*) \\ 0 & \text{with prob. } 1 - f(x_i^*) \end{cases}$ (independently)

Suppose: for some $c > 1$

$$1 - c^{-x} \leq f(x) \leq c^{x-1}$$

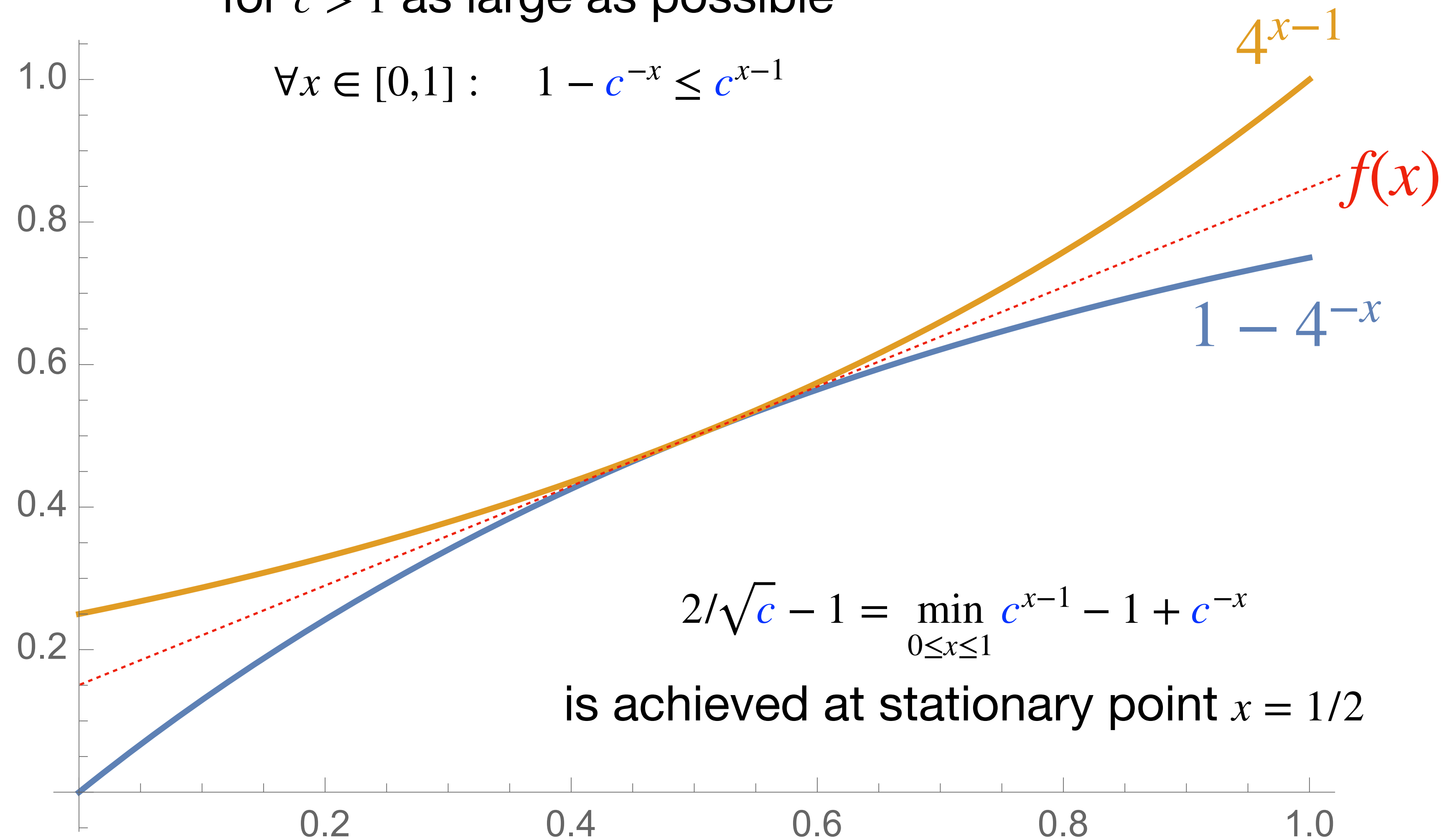
$$\Pr[C_j \text{ is unsatisfied}] \leq c^{-y_j^*}$$

Jenssen's inequality

$$\mathbb{E}[SOL] \geq \sum_{j=1}^m \left(1 - c^{-y_j^*} \right) \geq (1 - 1/c) \sum_{j=1}^m y_j^* \geq (1 - 1/c) OPT$$

for $c > 1$ as large as possible

$$\forall x \in [0,1] : \quad 1 - c^{-x} \leq c^{x-1}$$



$$\begin{aligned}
& \text{maximize} && \sum_{j=1}^m y_j \\
& \text{subject to} && \sum_{i \in S_j^+} x_i + \sum_{i \in S_j^-} (1 - x_i) \geq y_j \quad 1 \leq j \leq m \\
& && x_i, y_j \in [0,1] \quad \forall i, j
\end{aligned}$$

Optimal fractional solution: $x^* \in [0,1]^n$, $y^* \in [0,1]^m$

Nonlinear rounding: $\hat{x}_i = \begin{cases} 1 & \text{with prob. } f(x_i^*) \\ 0 & \text{with prob. } 1 - f(x_i^*) \end{cases}$ (independently)

Suppose:

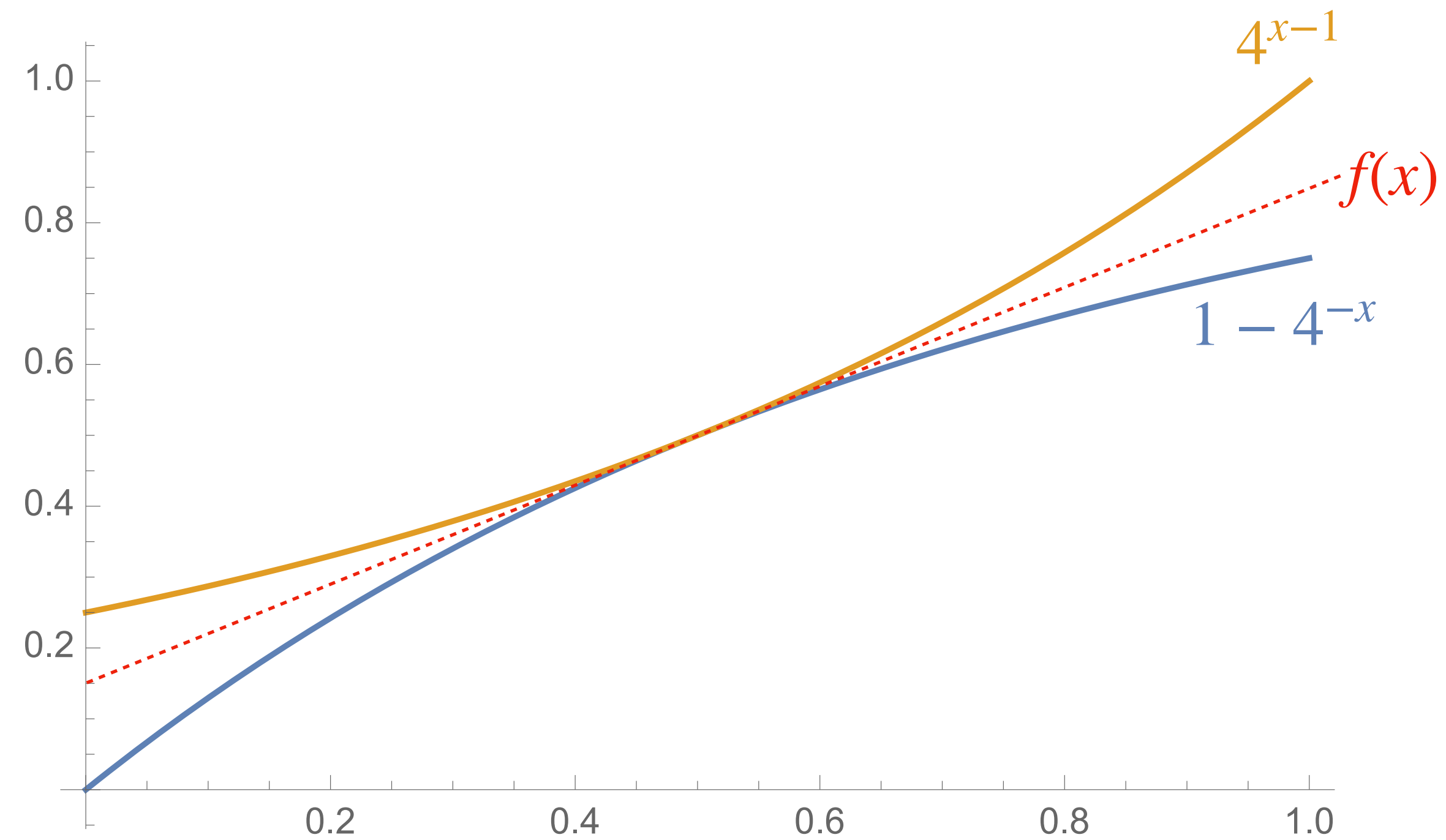
$$1 - 4^{-x} \leq f(x) \leq 4^{x-1}$$

$$\Pr[C_j \text{ is unsatisfied}] \leq 4^{-y_j^*}$$

Jenssen's inequality

$$\mathbb{E}[SOL] \geq \sum_{j=1}^m \left(1 - 4^{-y_j^*} \right) \geq \frac{3}{4} \sum_{j=1}^m y_j^* \geq \frac{3}{4} OPT$$

Max-SAT



- Nonlinear rounding of LP-relaxation for Max-SAT satisfies $\geq \frac{3}{4}OPT$ clauses in expectation.
- Easy-to-compute *explicit* rounding $f(\cdot)$?

Power of Two Choices

- Use random assignment to satisfy M_1 clauses;
- Use linear rounding of LP relaxation to satisfy M_2 clauses;
- Return the solution with more satisfied clauses.

$$\mathbb{E} \left[\max \{ M_1, M_2 \} \right] \geq \mathbb{E} \left[\frac{M_1 + M_2}{2} \right] \geq \frac{3}{4} \sum_{j=1}^m y_j^* \geq \frac{3}{4} OPT$$

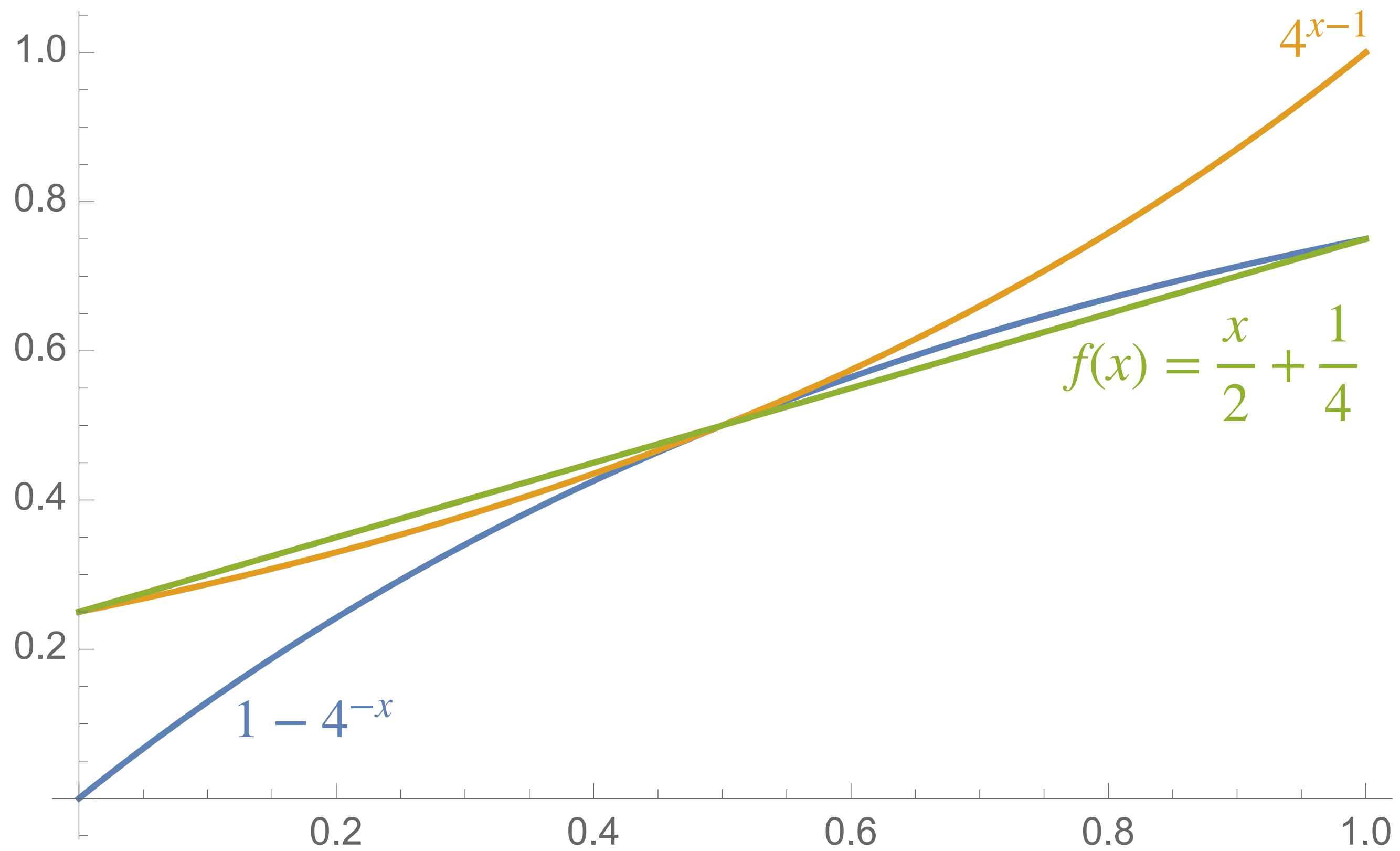
average of two
rounding schemes

$$\begin{aligned}
& \text{maximize} && \sum_{j=1}^m y_j \\
& \text{subject to} && \sum_{i \in S_j^+} x_i + \sum_{i \in S_j^-} (1 - x_i) \geq y_j \quad 1 \leq j \leq m \\
& && x_i, y_j \in [0,1] \quad \forall i, j
\end{aligned}$$

Optimal solution: $\mathbf{x}^* \in [0,1]^n$, $\mathbf{y}^* \in [0,1]^m$

$$\hat{x}_i = \begin{cases} \begin{cases} 1 & \text{w.p. } \frac{1}{2} \\ 0 & \text{w.p. } \frac{1}{2} \end{cases} & \text{w.p. } \frac{1}{2} \quad (\text{random assign.}) \\ \begin{cases} 1 & \text{w.p. } x_i^* \\ 0 & \text{w.p. } 1 - x_i^* \end{cases} & \text{w.p. } \frac{1}{2} \quad (\text{linear rounding}) \end{cases}$$

maximize $\sum_{j=1}^m y_j$



$i \leq m$

$$\begin{aligned}
& \text{maximize} && \sum_{j=1}^m y_j \\
& \text{subject to} && \sum_{i \in S_j^+} x_i^* + \sum_{i \in S_j^-} (1 - x_i^*) \geq y_j^* \quad 1 \leq j \leq m \\
& && x_i, y_j \in [0, 1] \quad \forall i, j
\end{aligned}$$

Optimal solution: $\mathbf{x}^* \in [0, 1]^n$, $\mathbf{y}^* \in [0, 1]^m$

$$\hat{x}_i = \begin{cases} 1 & \text{with prob. } \frac{1}{2}x_i^* + \frac{1}{4} = f(x_i^*) \\ 0 & \text{with prob. } -\frac{1}{2}x_i^* + \frac{3}{4} = 1 - f(x_i^*) \end{cases}$$

$$\Pr[C_j \text{ is unsatisfied}] = \prod_{i \in S_j^+} \left(-\frac{1}{2}x_i^* + \frac{3}{4} \right) \prod_{i \in S_j^-} \left(\frac{1}{2}x_i^* + \frac{1}{4} \right)$$

$$\begin{aligned}
& \text{AM} \\
& \text{GM} \leq \left(\frac{1}{k_j} \left(\sum_{i \in S_j^+} \left(-\frac{1}{2}x_i^* + \frac{3}{4} \right) + \sum_{i \in S_j^-} \left(\frac{1}{2}x_i^* + \frac{1}{4} \right) \right) \right)^{k_j} \leq \left(\frac{3}{4} - \frac{y_j^*}{2k_j} \right)^{k_j}
\end{aligned}$$

$$\begin{array}{ll}
\text{maximize} & \sum_{j=1}^m y_j \\
\text{subject to} & \sum_{i \in S_j^+} x_i + \sum_{i \in S_j^-} (1 - x_i) \geq y_j \quad 1 \leq j \leq m \\
& x_i, y_j \in [0,1] \quad \forall i, j
\end{array}$$

Optimal solution: $\mathbf{x}^* \in [0,1]^n$, $\mathbf{y}^* \in [0,1]^m$

$$\hat{x}_i = \begin{cases} 1 & \text{with prob. } \frac{1}{2}x_i^* + \frac{1}{4} = f(x_i^*) \\ 0 & \text{with prob. } -\frac{1}{2}x_i^* + \frac{3}{4} = 1 - f(x_i^*) \end{cases}$$

$$\Pr[C_j \text{ is satisfied}] \geq 1 - \left(\frac{3}{4} - \frac{y_j^*}{2k_j} \right)^{k_j} \geq \left(1 - \left(\frac{3}{4} - \frac{1}{2k_j} \right)^{k_j} \right) y_j^* \geq \frac{3}{4} y_j^*$$

Jenssen's inequality

$$\mathbb{E}[SOL] \geq \frac{3}{4} \sum_{j=1}^m y_j^* \geq \frac{3}{4} OPT$$

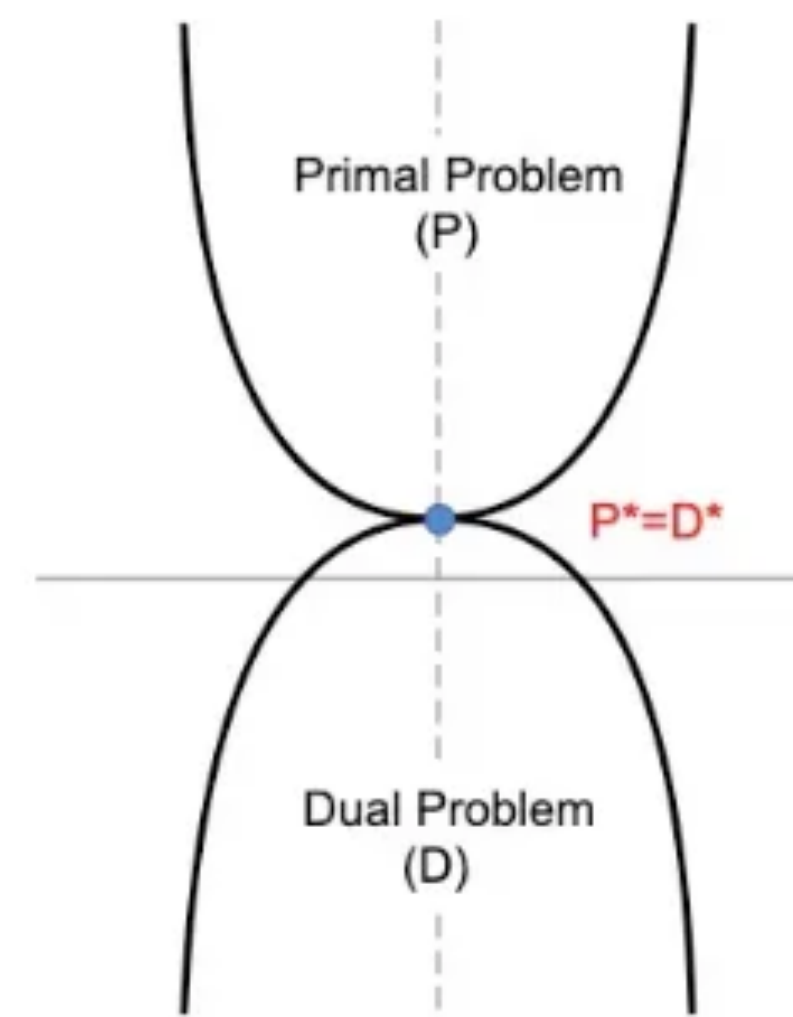
Max-SAT

Instance: A CNF formula $\Phi = C_1 \wedge C_2 \wedge \dots \wedge C_m$.

Find an assignment $x \in \{T, F\}^n$ that maximizes the number of satisfied clauses.

- Randomized rounding of LP-relaxation for Max-SAT satisfies $\geq \frac{3}{4}OPT$ clauses in expectation.
- It can be *de-randomized* via **conditional expectation**.
- The **integrality gap** of the LP-relaxation for Max-SAT is 3/4.
- Max-3SAT has a 7/8-approximation algorithm by rounding semidefinite programming (**SDP**) relaxation, and cannot have $>7/8$ -approximation ratio in poly-time unless **NP=P**.

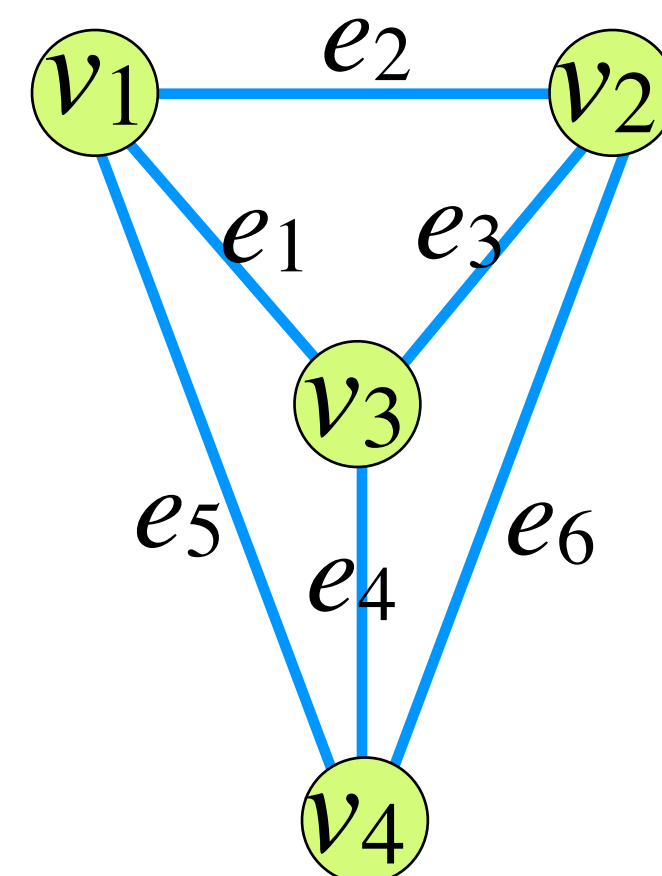
Duality



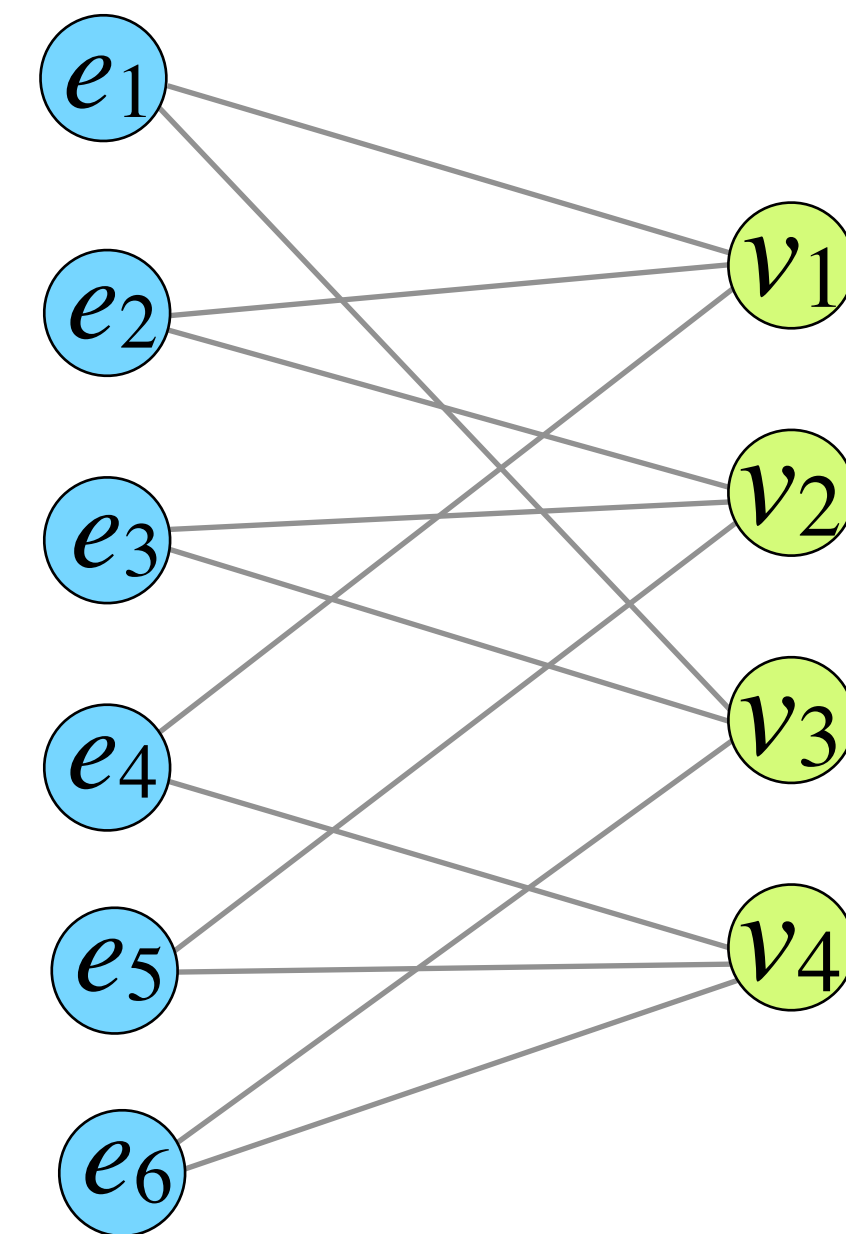
Vertex Cover

Instance: An undirected graph $G(V,E)$

Find the smallest $C \subseteq V$ that every edge has at least one endpoint in C .



incidence
graph



instance of set cover
with frequency = 2

Instance: An undirected graph $G(V,E)$

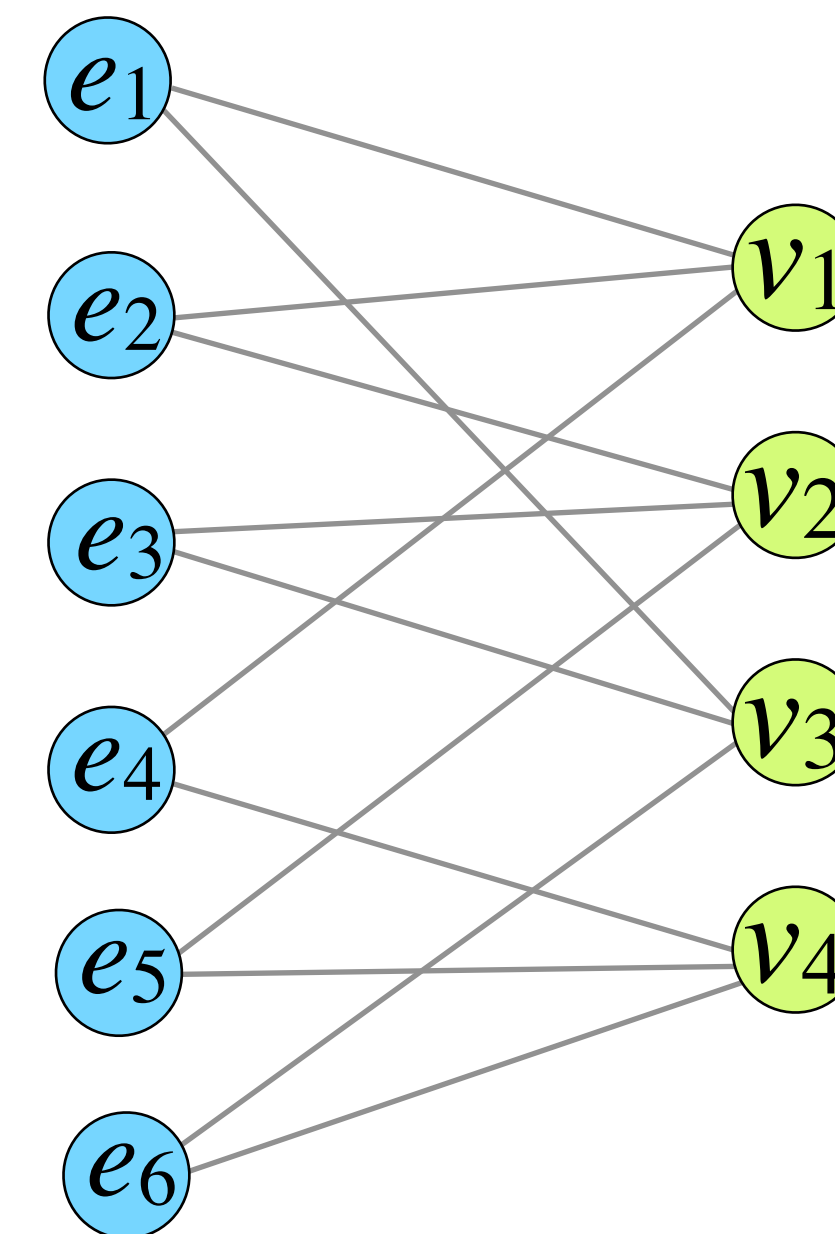
Find the smallest $C \subseteq V$ that every edge has at least one endpoint in C .

Find a *maximal matching* M ;
return the set $C = \{v: (u,v) \in M\}$ of *matched* vertices;

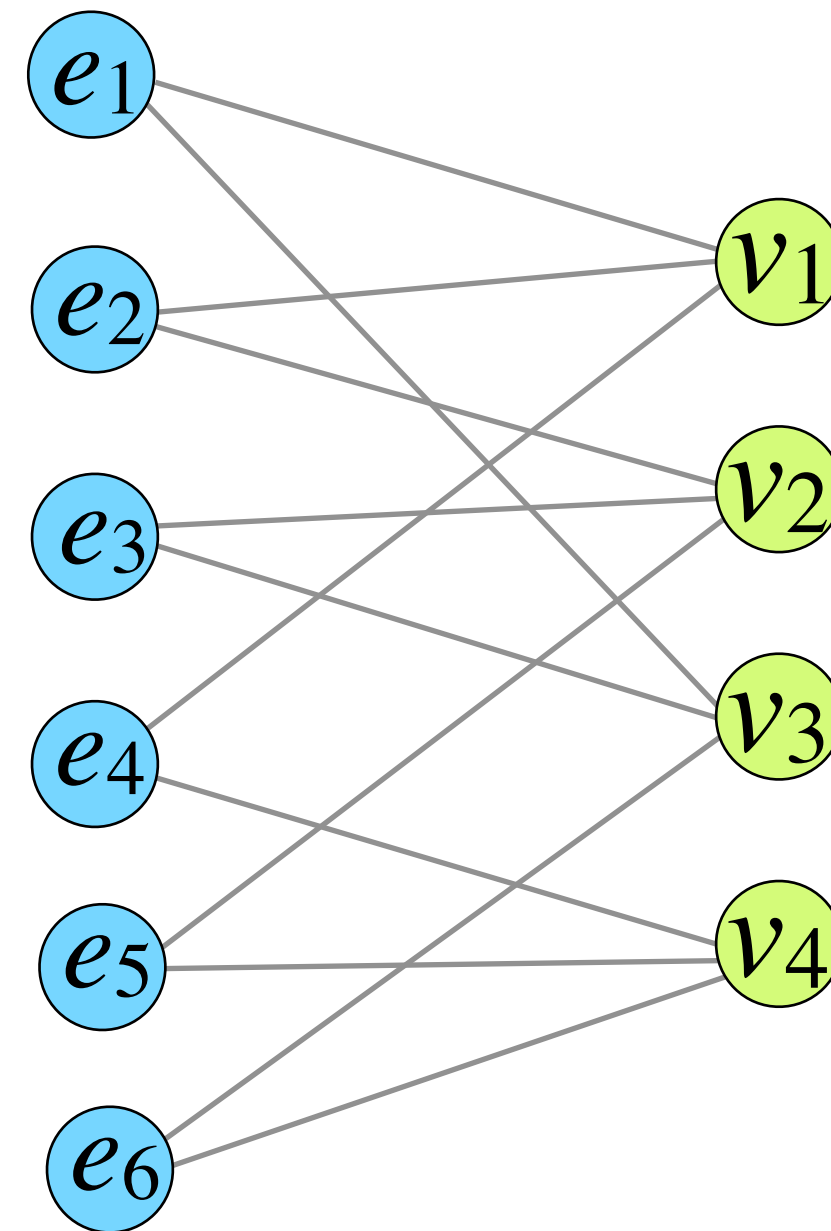
maximality $\Rightarrow C$ is vertex cover

matching $\Rightarrow |M| \leq \text{OPT}_{\text{VC}}$
(*weak duality*)

$$|C| \leq 2|M| \leq 2\text{OPT}$$



Duality



vertex cover:

constraints

$$\sum_{v \in e} x_v \geq 1$$

variables

$$x_v \in \{0, 1\}$$

matching:

variables

$$y_e \in \{0, 1\}$$

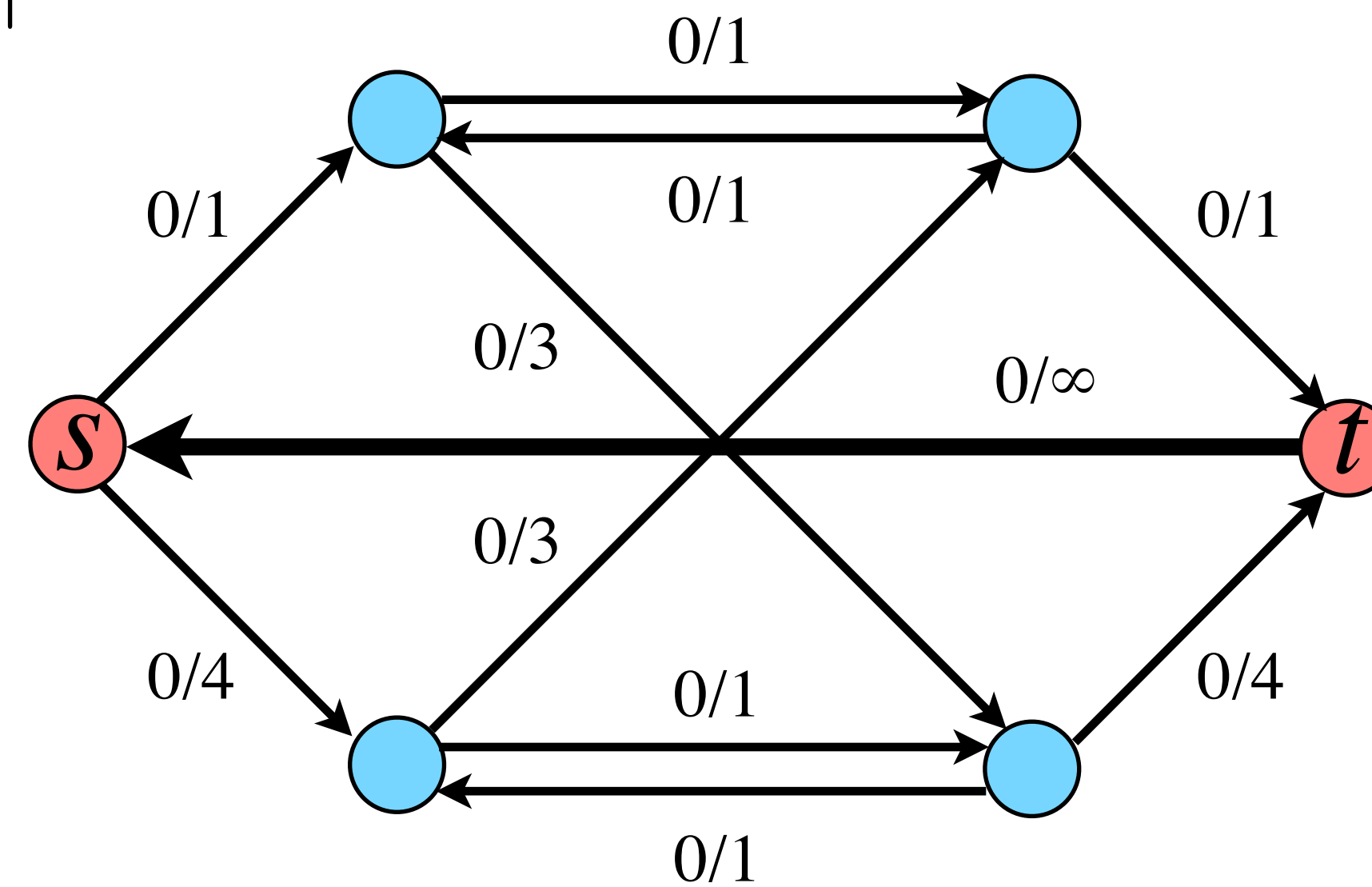
constraints

$$\sum_{e \ni v} y_e \leq 1$$

Max-Flow

digraph: $G = (V, E)$ source: s sink: t

capacity: $c : E \rightarrow \mathbb{R}^+$



max f_{ts}

d_{uv} **s.t.** $0 \leq f_{uv} \leq c_{uv} \quad \forall (u, v) \in E$

p_u $\sum_{w:(w,u) \in E} f_{wu} - \sum_{v:(u,v) \in E} f_{uv} \geq 0 \quad \forall u \in V$

Dual-LP

digraph: $G = (V, E)$

source: s sink: t

capacity: $c : E \rightarrow \mathbb{R}^+$

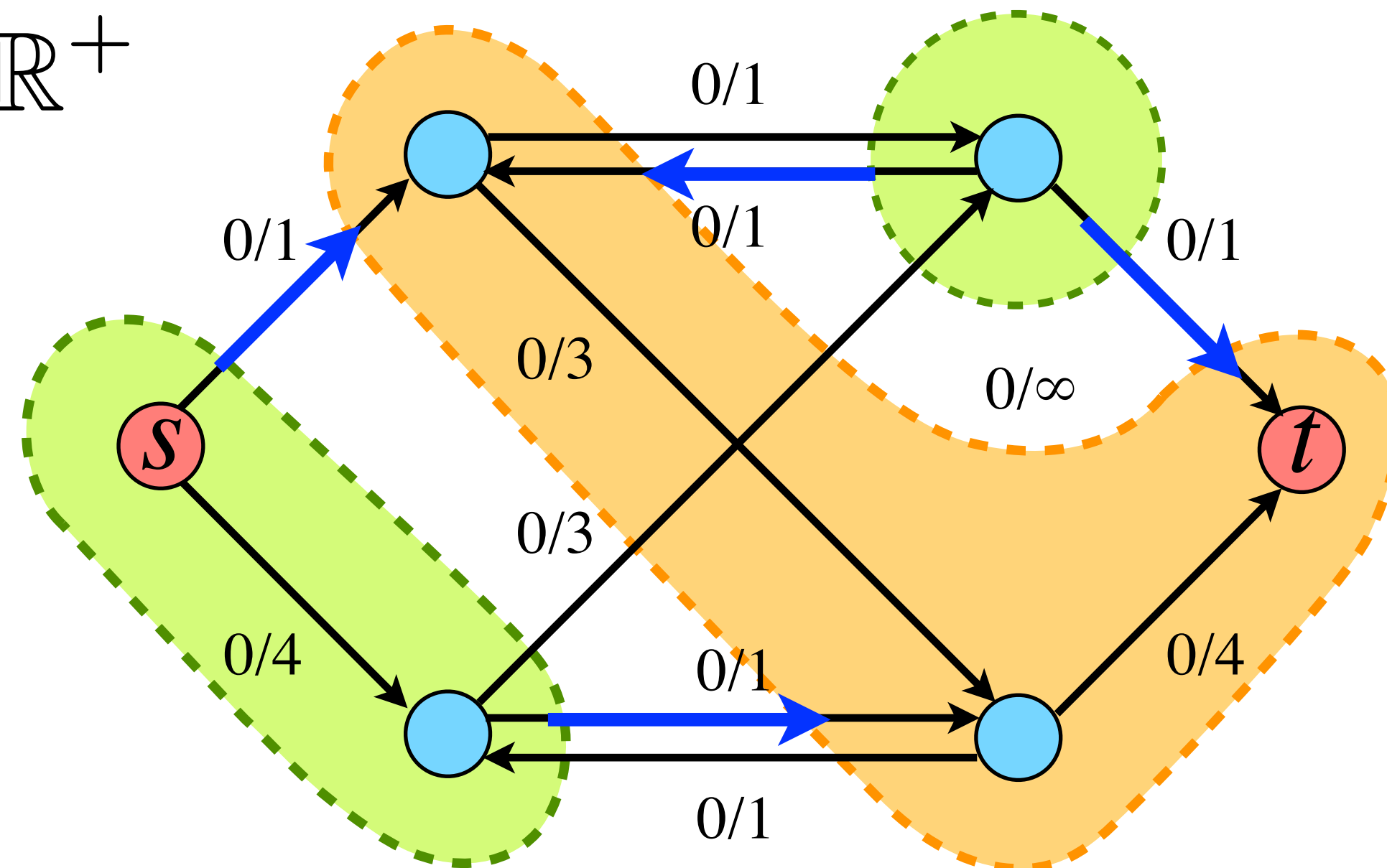
$$\min \sum_{(u,v) \in E} c_{uv} d_{uv}$$

$$\text{s.t.} \quad d_{uv} - p_u + p_v \geq 0$$

$$p_s - p_t \geq 1$$

$$d_{uv} \geq 0, p_u \in [0, 1] \quad \forall (u,v) \in E$$

$$\forall (u,v) \in E \quad \forall u \in V$$



Duality

Instance: graph $G(V,E)$

primal: minimize $\sum_{v \in V} x_v$
subject to $\sum_{v \in e} x_v \geq 1, \quad \forall e \in E$
 $x_v \in \{0, 1\}, \quad \forall v \in V$

vertex
covers

dual: maximize $\sum_{e \in E} y_e$
subject to $\sum_{e \ni v} y_e \leq 1, \quad \forall v \in V$
 $y_e \in \{0, 1\}, \quad \forall e \in E$

matchings

Duality for LP-Relaxation

Instance: graph $G(V,E)$

primal: minimize $\sum_{v \in V} x_v$

subject to $\sum_{v \in e} x_v \geq 1, \quad \forall e \in E$

$x_v \geq 0, \quad \forall v \in V$

dual: maximize $\sum_{e \in E} y_e$

subject to $\sum_{e \ni v} y_e \leq 1, \quad \forall v \in V$

$y_e \geq 0, \quad \forall e \in E$

Estimate the Optima

minimize

$$7x_1 + x_2 + 5x_3$$

VI

subject to

$$x_1 - x_2 + 3x_3 \geq 10$$

+

+

$$5x_1 + 2x_2 - x_3 \geq 6$$

||

$$x_1, x_2, x_3 \geq 0 \quad 16$$

$$16 \leq \text{OPT} \leq 30 \text{ by feasible solution}$$

$$x = (2, 1, 3)$$

Estimate the Optima

minimize

$$7x_1 + x_2 + 5x_3$$

VI

subject to

$$x_1 - x_2 + 3x_3 \geq 10 y_1$$

+

+

$$5x_1 + 2x_2 - x_3 \geq 6 y_2$$

$$x_1, x_2, x_3 \geq 0$$

$$10y_1 + 6y_2 \leq \text{OPT}$$

for any

$$\begin{array}{rclcl} y_1 & + & 5y_2 & \leq & 7 \\ -y_1 & + & 2y_2 & \leq & 1 \\ 3y_1 & - & y_2 & \leq & 5 \end{array} \quad y_1, y_2 \geq 0$$

Primal-Dual

Primal

$$\begin{array}{ll}\text{min} & 7x_1 + x_2 + 5x_3 \\ \text{s.t.} & x_1 - x_2 + 3x_3 \geq 10 \\ & 5x_1 + 2x_2 - x_3 \geq 6 \\ & x_1, x_2, x_3 \geq 0\end{array}$$

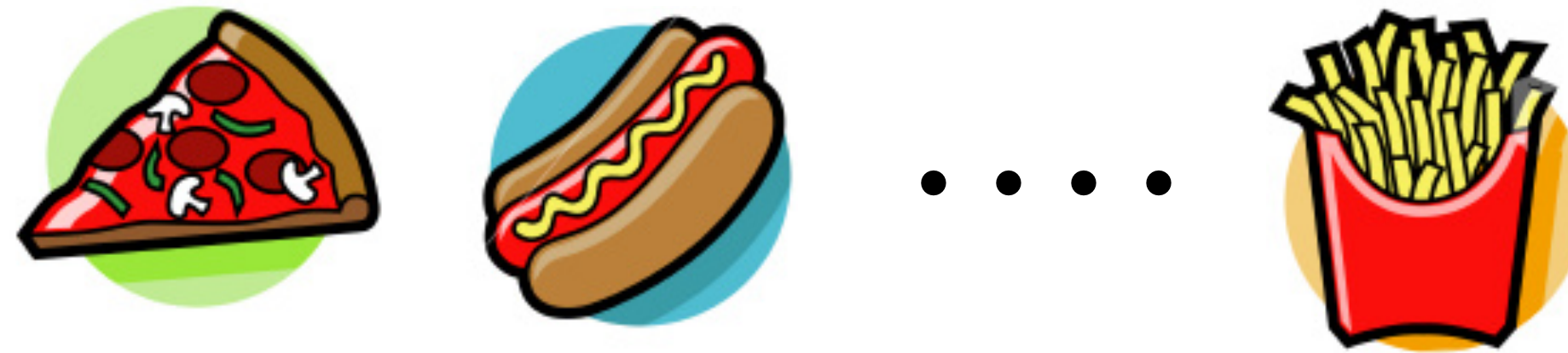
Dual

$$\begin{array}{ll}\text{max} & 10y_1 + 6y_2 \\ \text{s.t.} & y_1 + 5y_2 \leq 7 \\ & -y_1 + 2y_2 \leq 1 \\ & 3y_1 - y_2 \leq 5 \\ & y_1, y_2 \geq 0\end{array}$$

\forall dual feasible
 \leq primal OPT

LP \in **NP** \cap **coNP**

Surviving Problem



price
vitamin 1
⋮
vitamin m

c_1	c_2	⋯	c_n
a_{11}	a_{12}	⋯	a_{1n}
⋮	⋮		⋮
a_{m1}	a_{m2}	⋯	a_{mn}

healthy

$$\geq b_1$$

⋮

$$\geq b_m$$

solution: x_1 x_2 ⋯ x_n

minimize the total price while keeping healthy

Surviving Problem

$$\min \quad c^T x$$

$$\text{s.t.} \quad Ax \geq b$$

$$x \geq 0$$

price	c_1	c_2	\dots	c_n	healthy
vitamin 1	a_{11}	a_{12}	\dots	a_{1n}	$\geq b_1$
\vdots	\vdots	\vdots		\vdots	\vdots
vitamin m	a_{m1}	a_{m2}	\dots	a_{mn}	$\geq b_m$

solution: $x_1 \quad x_2 \quad \dots \quad x_n$

minimize the total price while keeping healthy

LP Duality

Primal:

$$\min \mathbf{c}^T \mathbf{x}$$

$$\text{s.t. } \mathbf{Ax} \geq \mathbf{b}$$

$$\mathbf{x} \geq \mathbf{0}$$

Dual:

$$\max \mathbf{b}^T \mathbf{y}$$

$$\text{s.t. } \mathbf{y}^T \mathbf{A} \leq \mathbf{c}^T$$

$$\mathbf{y} \geq \mathbf{0}$$

dual
solution: **price**
 y_1 **vitamin 1**
 \vdots
 y_m **vitamin m**

c_1	c_2	\dots	c_n
a_{11}	a_{12}	\dots	a_{1n}
\vdots	\vdots		\vdots
a_{m1}	a_{m2}	\dots	a_{mn}

healthy

b_1
 \vdots
 b_m

m types of vitamin pills, design a pricing system
competitive to n natural foods, max the total price

LP Duality

Primal:

$$\min \mathbf{c}^T \mathbf{x} \quad \geq$$

$$\text{s.t. } \mathbf{Ax} \geq \mathbf{b}$$

$$\mathbf{x} \geq \mathbf{0}$$

Dual:

$$\max \mathbf{b}^T \mathbf{y}$$

$$\text{s.t. } \mathbf{y}^T \mathbf{A} \leq \mathbf{c}^T$$

$$\mathbf{y} \geq \mathbf{0}$$

Monogamy: $\text{dual}(\text{dual}(\text{LP})) = \text{LP}$

Weak Duality:

\forall feasible primal solution \mathbf{x} and dual solution \mathbf{y}

$$\mathbf{y}^T \mathbf{b} \leq \mathbf{y}^T \mathbf{A} \mathbf{x} \leq \mathbf{c}^T \mathbf{x}$$

LP Duality

Primal:

$$\min \quad c^T x \quad \geq$$

$$\text{s.t.} \quad Ax \geq b$$

$$x \geq 0$$

Dual:

$$\max \quad b^T y$$

$$\text{s.t.} \quad y^T A \leq c^T$$

$$y \geq 0$$

Weak Duality Theorem:

\forall feasible primal solution x and dual solution y

$$y^T b \leq c^T x$$

LP Duality

Primal:

$$\min \quad c^T x$$

$$\text{s.t.} \quad Ax \geq b$$

$$x \geq 0$$

Dual:

$$\max \quad b^T y$$

$$\text{s.t.} \quad y^T A \leq c^T$$

$$y \geq 0$$

Strong Duality Theorem:

Primal LP has finite optimal solution x^*
iff dual LP has finite optimal solution y^* .

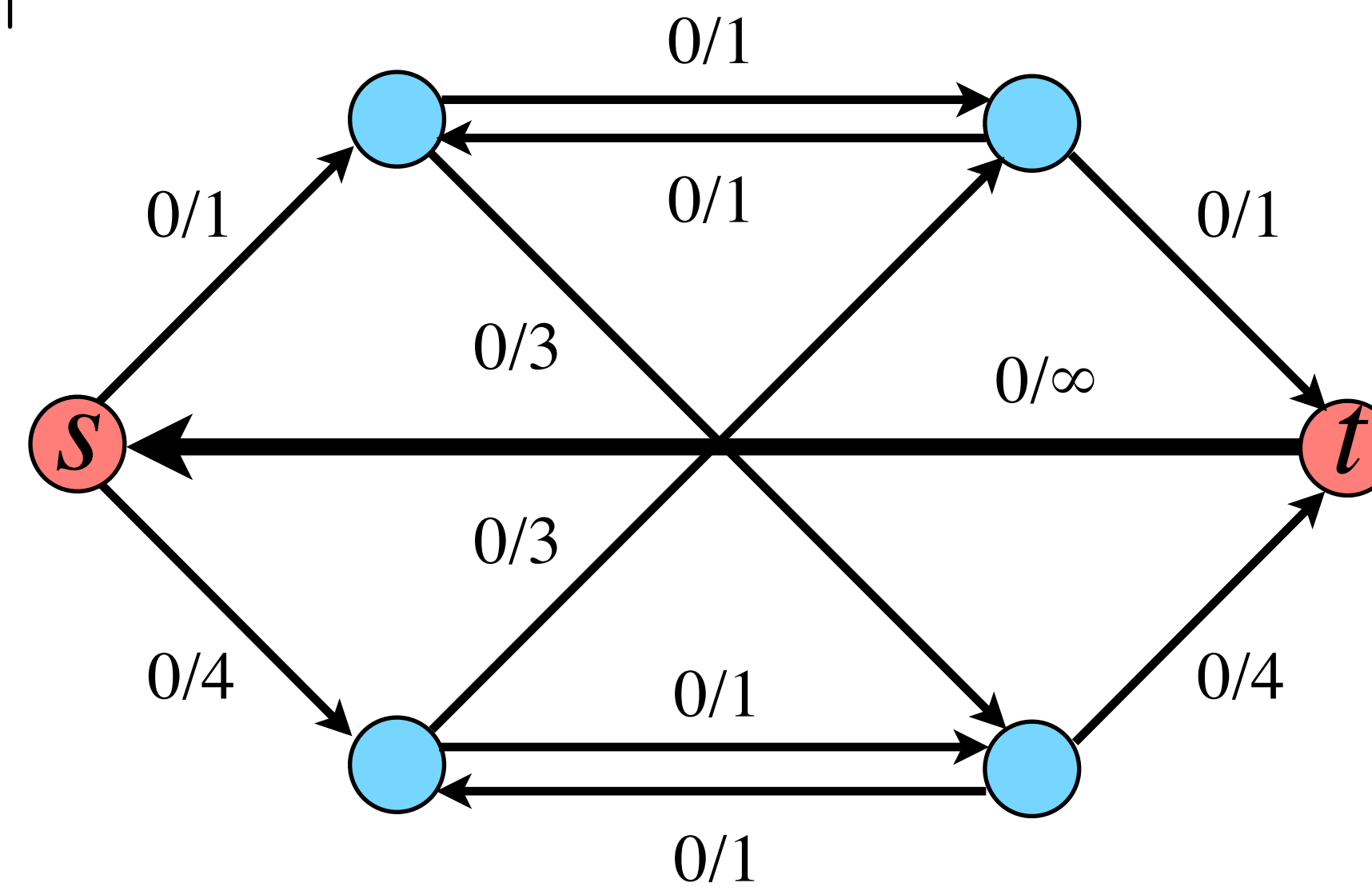
$$y^{*T} b = c^T x^*$$

Max-Flow

digraph: $D(V, E)$

source: s sink: t

capacity: $c : E \rightarrow \mathbb{R}^+$



max f_{ts}

d_{uv} **s.t.** $0 \leq f_{uv} \leq c_{uv} \quad \forall (u, v) \in E$

p_u $\sum_{w:(w,u) \in E} f_{wu} - \sum_{v:(u,v) \in E} f_{uv} \leq 0 \quad \forall u \in V$

Dual-LP

digraph: $D(V, E)$

source: s sink: t

capacity: $c : E \rightarrow \mathbb{R}^+$

$$\min \sum_{(u,v) \in E} c_{uv} d_{uv}$$

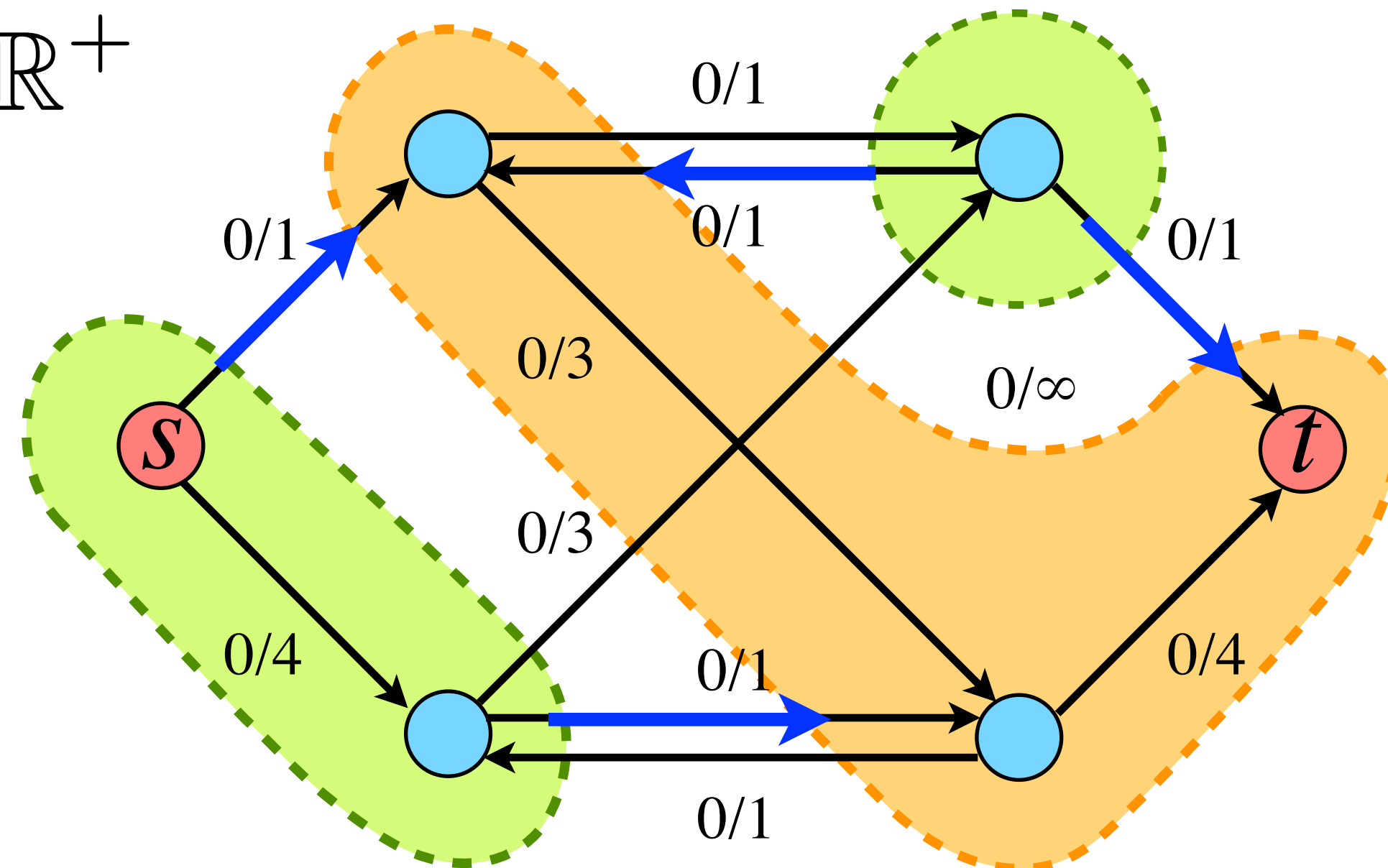
$$\text{s.t.} \quad d_{uv} - p_u + p_v \geq 0$$

$$p_s - p_t \geq 1$$

$$d_{uv}, p_u \in \{0, 1\}$$

$$\forall (u, v) \in E$$

$$\forall (u, v) \in E \quad \forall u \in V$$



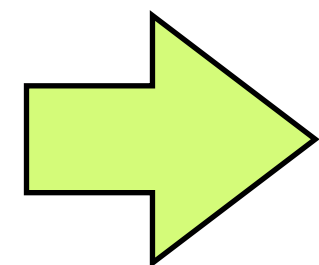
Primal: $\min \mathbf{c}^T \mathbf{x}$
s.t. $A\mathbf{x} \geq \mathbf{b}$
 $\mathbf{x} \geq \mathbf{0}$

Dual: $\max \mathbf{b}^T \mathbf{y}$
s.t. $\mathbf{y}^T A \leq \mathbf{c}^T$
 $\mathbf{y} \geq \mathbf{0}$

\forall feasible primal solution \mathbf{x} and dual solution \mathbf{y}

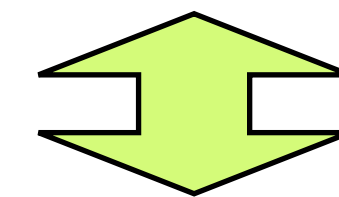
$$\mathbf{y}^T \mathbf{b} \leq \mathbf{y}^T A \mathbf{x} \leq \mathbf{c}^T \mathbf{x}$$

**Strong Duality
Theorem**

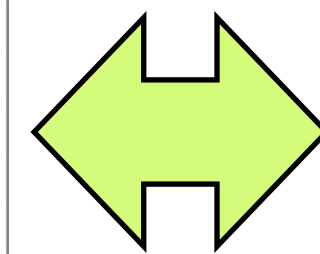


\mathbf{x} and \mathbf{y} are both optimal iff

$$\mathbf{y}^T \mathbf{b} = \mathbf{y}^T A \mathbf{x} = \mathbf{c}^T \mathbf{x}$$



$\forall i$: either $A_{i \cdot} \mathbf{x} = b_i$ or $y_i = 0$
 $\forall j$: either $\mathbf{y}^T A_{\cdot j} = c_j$ or $x_j = 0$



$$\sum_{i=1}^m b_i y_i = \sum_{i=1}^m \left(\sum_{j=1}^n a_{ij} x_j \right) y_i$$

$$\sum_{j=1}^n c_j x_j = \sum_{j=1}^n \left(\sum_{i=1}^m a_{ij} y_i \right) x_j$$

Complementary Slackness

Primal:	$\min \quad c^T x$	Dual:	$\max \quad b^T y$
	s.t. $Ax \geq b$		s.t. $y^T A \leq c^T$
	$x \geq 0$		$y \geq 0$

Complementary Slackness Conditions:

\forall feasible primal solution x and dual solution y
 x and y are both optimal iff

$$\begin{aligned} \forall i: & \text{ either } A_{i \cdot} x = b_i \text{ or } y_i = 0 \\ \forall j: & \text{ either } y^T A_{\cdot j} = c_j \text{ or } x_j = 0 \end{aligned}$$

Relaxed

Complementary Slackness

Primal: $\min \mathbf{c}^T \mathbf{x}$
s.t. $A\mathbf{x} \geq \mathbf{b}$
 $\mathbf{x} \geq \mathbf{0}$

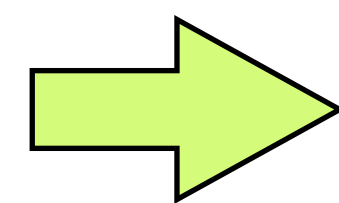
Dual: $\max \mathbf{b}^T \mathbf{y}$
s.t. $\mathbf{y}^T A \leq \mathbf{c}^T$
 $\mathbf{y} \geq \mathbf{0}$

\forall feasible primal solution \mathbf{x} and dual solution \mathbf{y}

for $\alpha, \beta \geq 1$:

$\forall i$: either $A_{i\cdot} \mathbf{x} \leq \alpha b_i$ or $y_i = 0$

$\forall j$: either $\mathbf{y}^T A_{\cdot j} \geq c_j / \beta$ or $x_j = 0$



$$\mathbf{c}^T \mathbf{x} \leq \alpha \beta \mathbf{b}^T \mathbf{y} \leq \alpha \beta \text{OPT}_{\text{LP}}$$

$$\sum_{j=1}^n c_j x_j \leq \sum_{j=1}^n \left(\beta \sum_{i=1}^m a_{ij} y_i \right) x_j = \beta \sum_{i=1}^m \left(\sum_{j=1}^n a_{ij} x_j \right) y_i \leq \alpha \beta \sum_{i=1}^m b_i y_i$$

Primal-Dual Schema

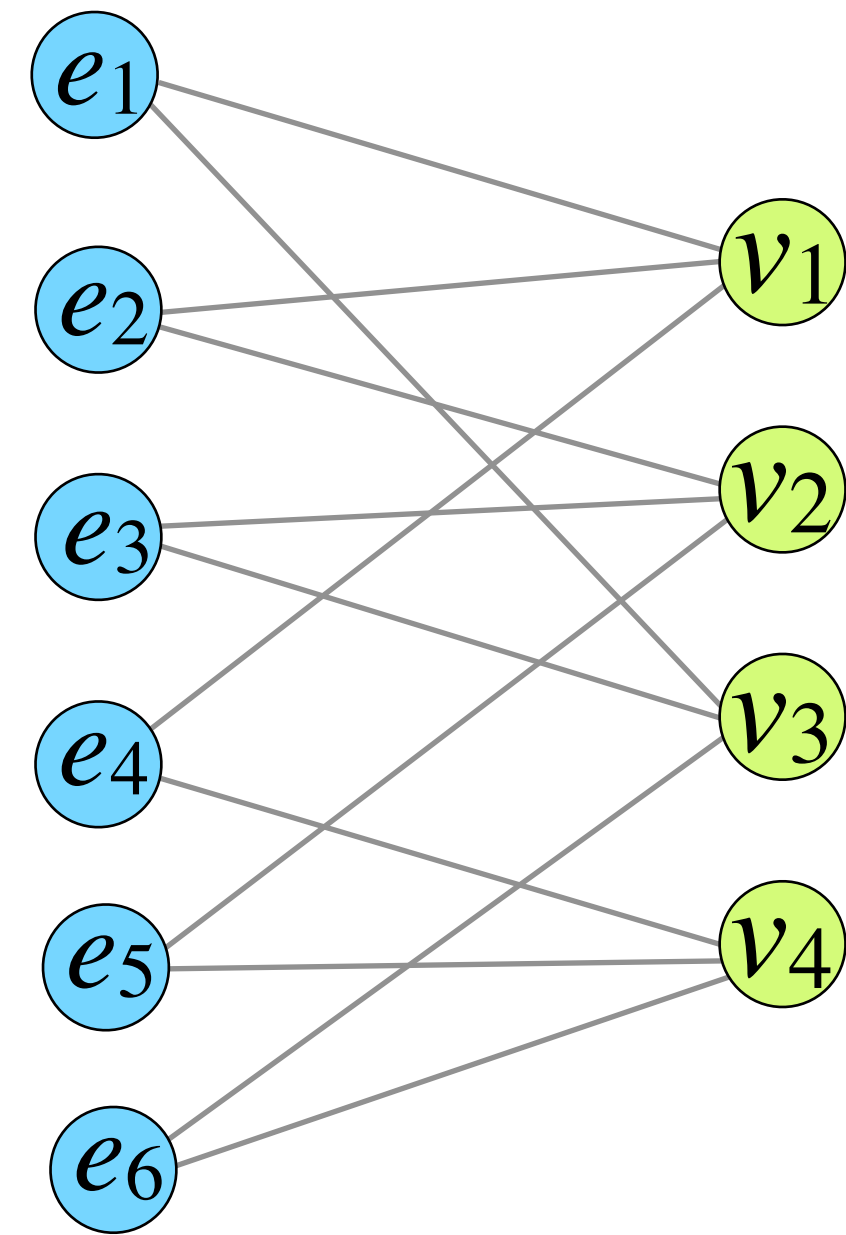
	Dual
Primal IP: $\min \mathbf{c}^T \mathbf{x}$	LP-relax: $\max \mathbf{b}^T \mathbf{y}$
s.t. $A\mathbf{x} \geq \mathbf{b}$	s.t. $\mathbf{y}^T A \leq \mathbf{c}^T$
$\mathbf{x} \in \mathbb{Z}_{\geq 0}$	$\mathbf{y} \geq \mathbf{0}$

Find a primal *integral* solution \mathbf{x} and a dual solution \mathbf{y}

for $\alpha, \beta \geq 1$:

$$\begin{aligned} \forall i: & \text{ either } A_{i \cdot} \mathbf{x} \leq \alpha b_i \text{ or } y_i = 0 \\ \forall j: & \text{ either } \mathbf{y}^T A_{\cdot j} \geq c_j / \beta \text{ or } x_j = 0 \end{aligned}$$

 $\mathbf{c}^T \mathbf{x} \leq \alpha \beta \mathbf{b}^T \mathbf{y} \leq \alpha \beta \text{OPT}_{\text{LP}} \leq \alpha \beta \text{OPT}_{\text{IP}}$



vertex cover:

constraints
 $\sum_{v \in e} x_v \geq 1$

variables
 $x_v \in \{0,1\}$

matching:

variables
 $y_e \in \{0,1\}$

constraints
 $\sum_{e \ni v} y_e \leq 1$

primal:

$$\min \sum_{v \in V} x_v$$

$$\text{s.t. } \sum_{v \in e} x_v \geq 1, \quad \forall e \in E$$

$$x_v \in \{0,1\}, \quad \forall v \in V$$

dual-relax:

$$\min \sum_{e \in E} y_e$$

$$\text{s.t. } \sum_{e \ni v} y_e \leq 1, \quad \forall v \in V$$

$$y_e \geq 0, \quad \forall e \in E$$

feasible (x, y) such that:

$\forall e:$	either $\sum_{v \in e} x_v = 1$ or $y_e = 0$
$\forall v:$	either $\sum_{e \ni v} y_e = 1$ or $x_v = 0$

primal:

$$\begin{array}{ll} \min & \sum_{v \in V} x_v \\ \text{s.t.} & \sum_{v \in e} x_v \geq 1, \quad \forall e \in E \\ & x_v \in \{0, 1\}, \quad \forall v \in V \end{array}$$

dual-relax:

$$\begin{array}{ll} \min & \sum_{e \in E} y_e \\ \text{s.t.} & \sum_{e \ni v} y_e \leq 1, \quad \forall v \in V \\ & y_e \geq 0, \quad \forall e \in E \end{array}$$

event: “ v is *tight (saturated)*” $\longleftrightarrow \sum_{e \ni v} y_e = 1$

Initially $x = \mathbf{0}$, $y = \mathbf{0}$;

while $E \neq \emptyset$

 pick an $e \in E$ and raise y_e ^{to 1} ~~until some v goes tight;~~

 set $x_v = 1$ for those ~~tight v~~ ^{$v \in e$} and delete all $e \ni v$ from E ;

every deleted e is incident to a v that $x_v = 1$ } $\Rightarrow \forall e \in E: \sum_{v \in e} x_v \geq 1$
all edges are eventually deleted } x is **feasible**

*relaxed
complementary
slackness:*

$\forall e$: either $\sum_{v \in e} x_v \leq 2$ or $y_e = 0$
 $\forall v$: either $\sum_{e \ni v} y_e = 1$ or $x_v = 0$

$$\Rightarrow \sum_{v \in V} x_v \leq 2 \cdot OPT$$

Initially $x = 0, y = 0$;
 while $E \neq \emptyset$
 pick an $e \in E$ and raise y_e ~~until some v goes tight;~~ ^{to 1}
 set $x_v = 1$ for those ~~tight v~~ ^{$v \in e$} and delete all $e \ni v$ from E ;

Find a *maximal matching*;
 return the set of *matched* vertices;

the returned set is a vertex cover

$$SOL \leq 2 OPT$$

The Primal-Dual Schema

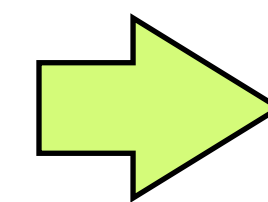
- Write down an LP-relaxation and its dual.

$$\begin{array}{ll} \min & c^T x \\ \text{s.t.} & Ax \geq b \\ & x \in \mathbb{Z}_{\geq 0} \end{array}$$

$$\begin{array}{ll} \max & b^T y \\ \text{s.t.} & y^T A \leq c^T \\ & y \geq 0 \end{array}$$

- Start with a **primal infeasible** solution x and a **dual feasible** solution y (usually $x=0, y=0$).
- Raise x and y until x is feasible:
 - raise y until some dual constraints gets **tight** $y^T A_{\cdot j} = c_j$;
 - raise x_j (integrally) corresponding to the **tight** dual constraints.
- Show the **complementary slackness conditions**:

$$\begin{array}{l} \forall i: \text{ either } A_{i \cdot} x \leq \alpha b_i \text{ or } y_i = 0 \\ \forall j: \text{ either } y^T A_{\cdot j} = c_j \text{ or } x_j = 0 \end{array}$$



$$\begin{array}{l} c^T x \leq \alpha b^T y \\ \leq \alpha \text{ OPT} \end{array}$$