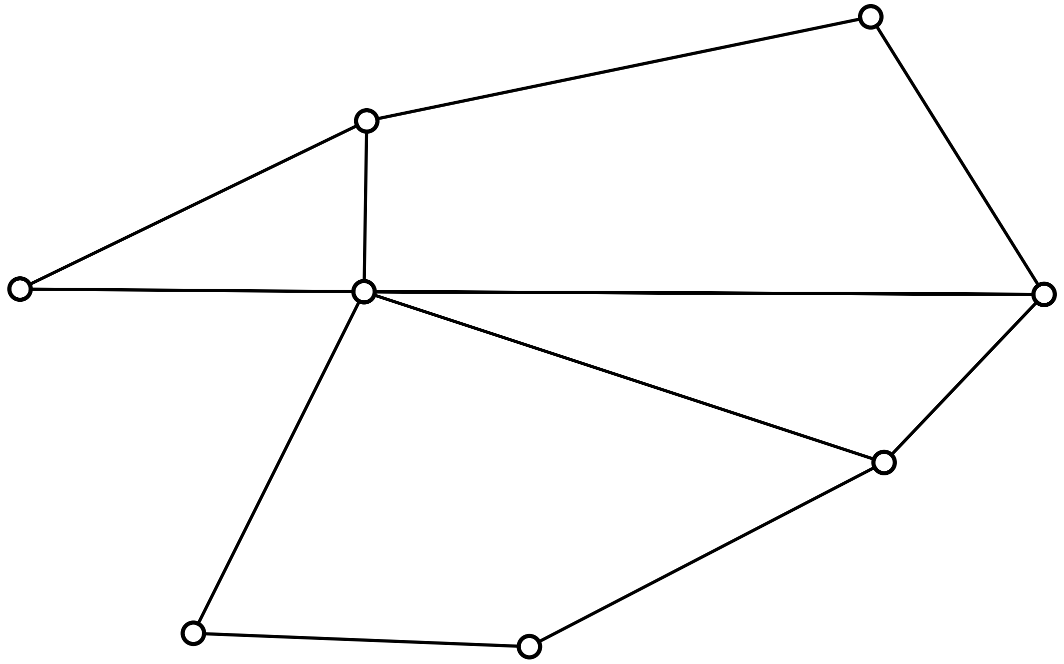
Applications of Duality

Total Unimodularity

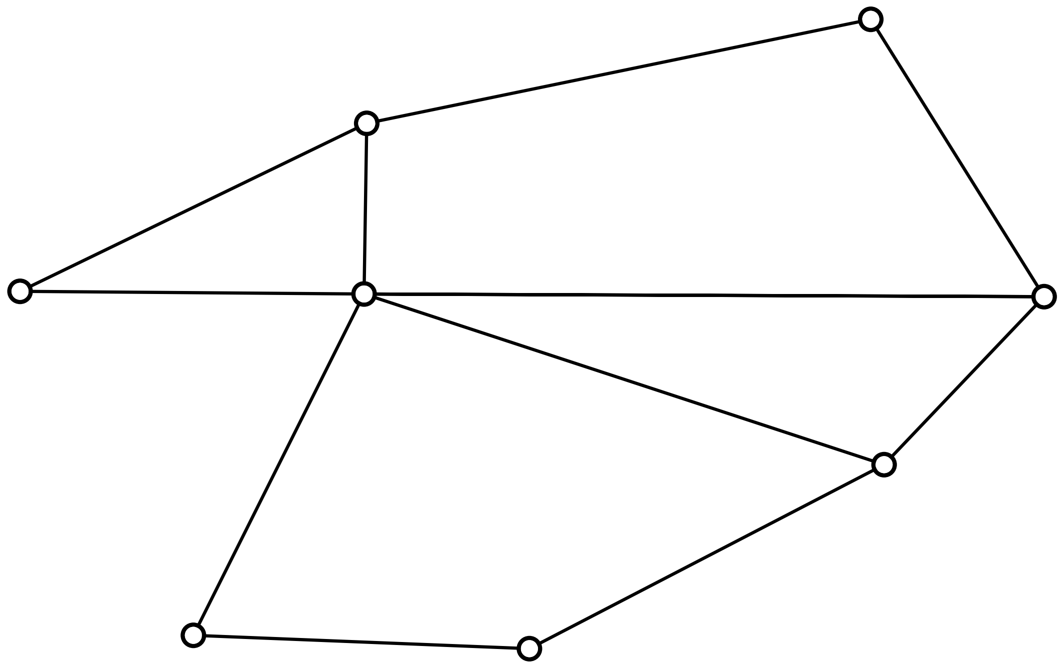
Matchings, Flows, and Shortest paths

Matchings



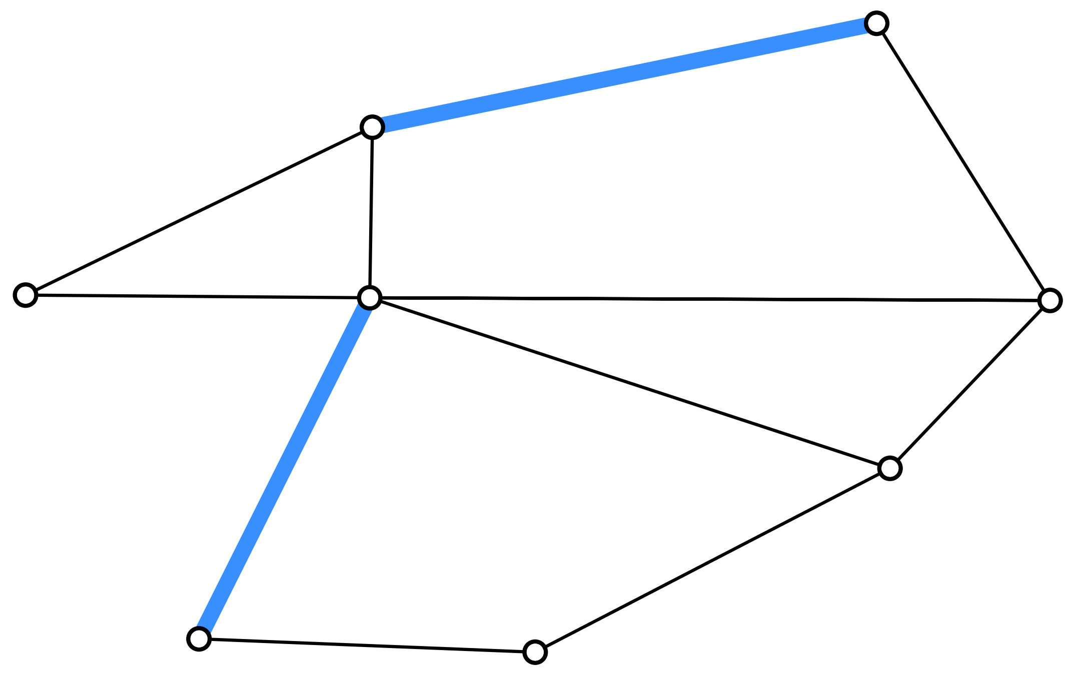
Matchings

An edge set $M \subseteq E$ of a graph $G = (V, E)$ is a matching if no two edges of M are adjacent (i.e., share an end vertex).



Matchings

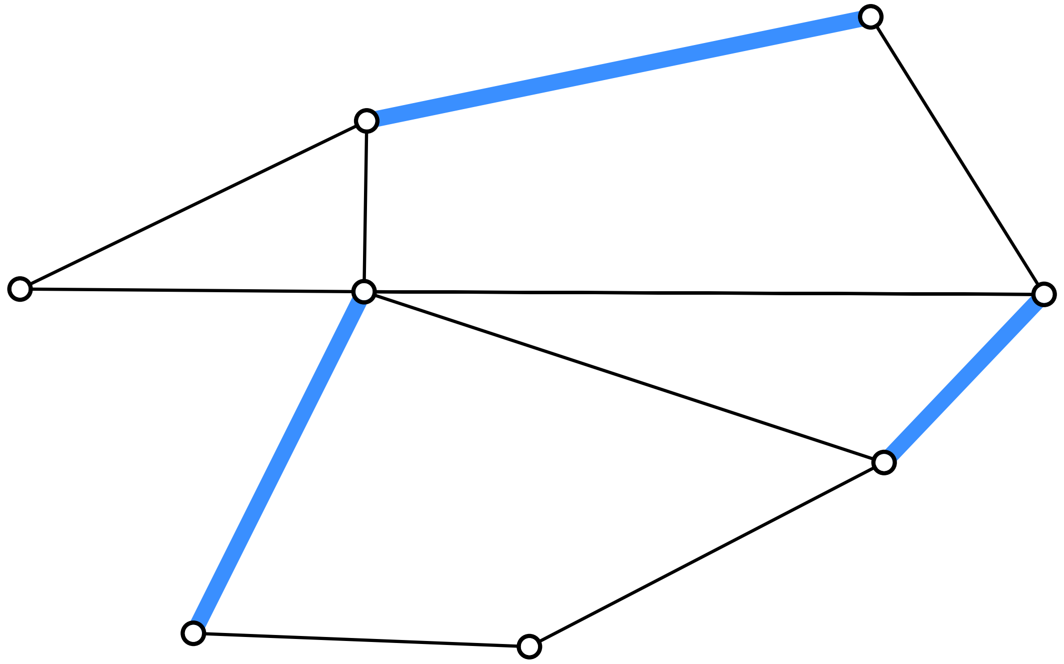
An edge set $M \subseteq E$ of a graph $G = (V, E)$ is a matching if no two edges of M are adjacent (i.e., share an end vertex).



Matchings

An edge set $M \subseteq E$ of a graph $G = (V, E)$ is a matching if no two edges of M are adjacent (i.e., share an end vertex).

M is maximal if there is no matching M' with $M' \supsetneq M$.



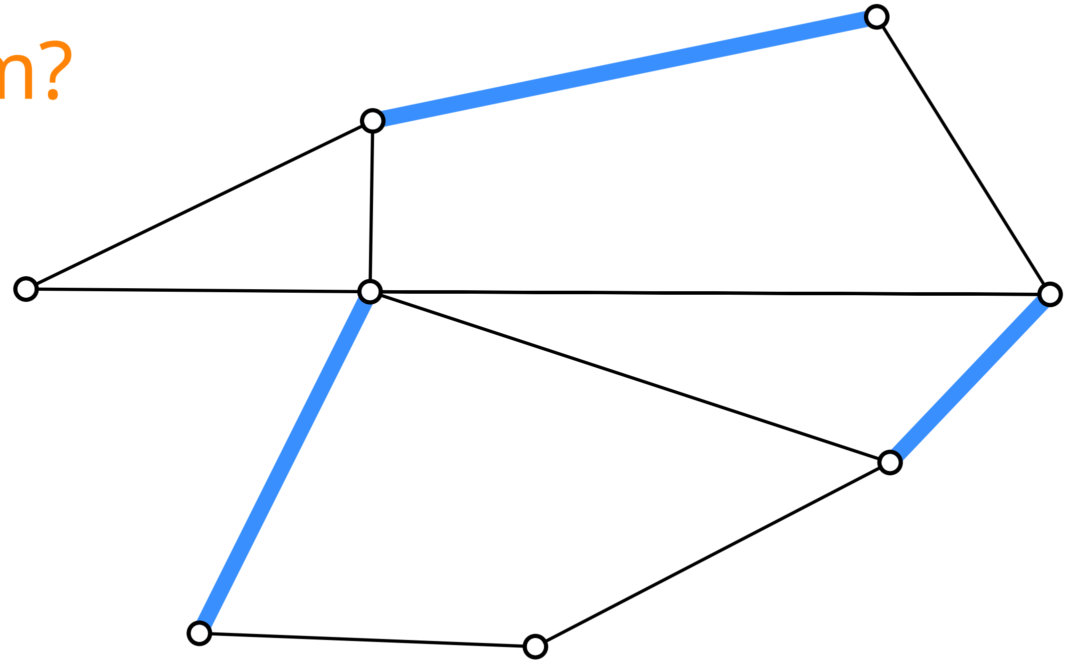
Matchings

An edge set $M \subseteq E$ of a graph $G = (V, E)$ is a matching if no two edges of M are adjacent (i.e., share an end vertex).

M is maximal if there is no matching M' with $M' \supsetneq M$.

M is maximum if it has the largest number of edges among all matchings.

Is this maximum?

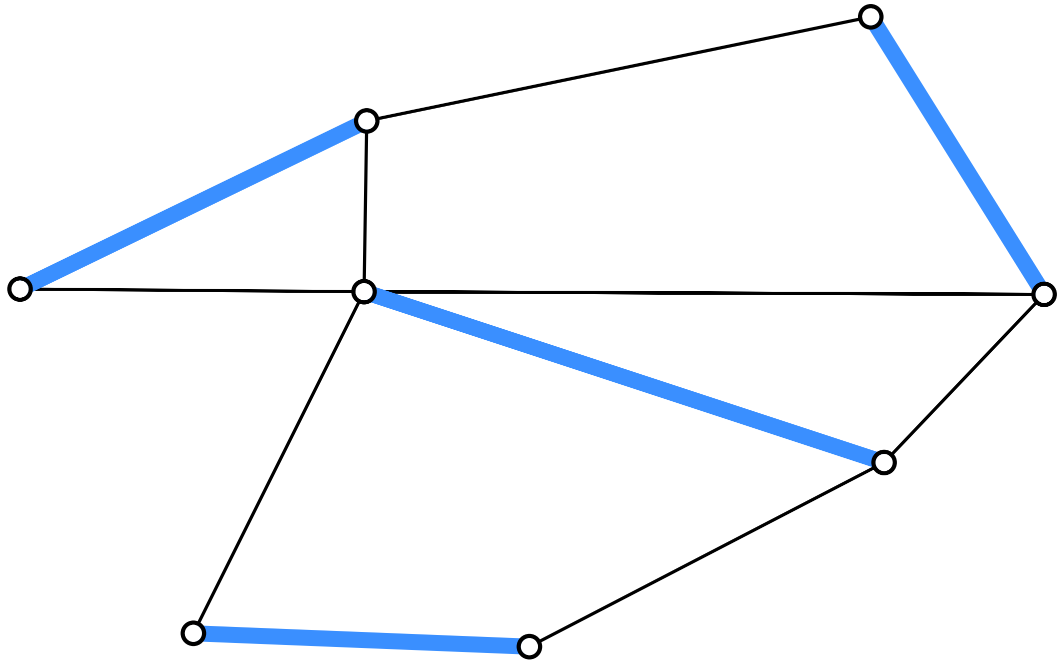


Matchings

An edge set $M \subseteq E$ of a graph $G = (V, E)$ is a matching if no two edges of M are adjacent (i.e., share an end vertex).

M is maximal if there is no matching M' with $M' \supsetneq M$.

M is maximum if it has the largest number of edges among all matchings.



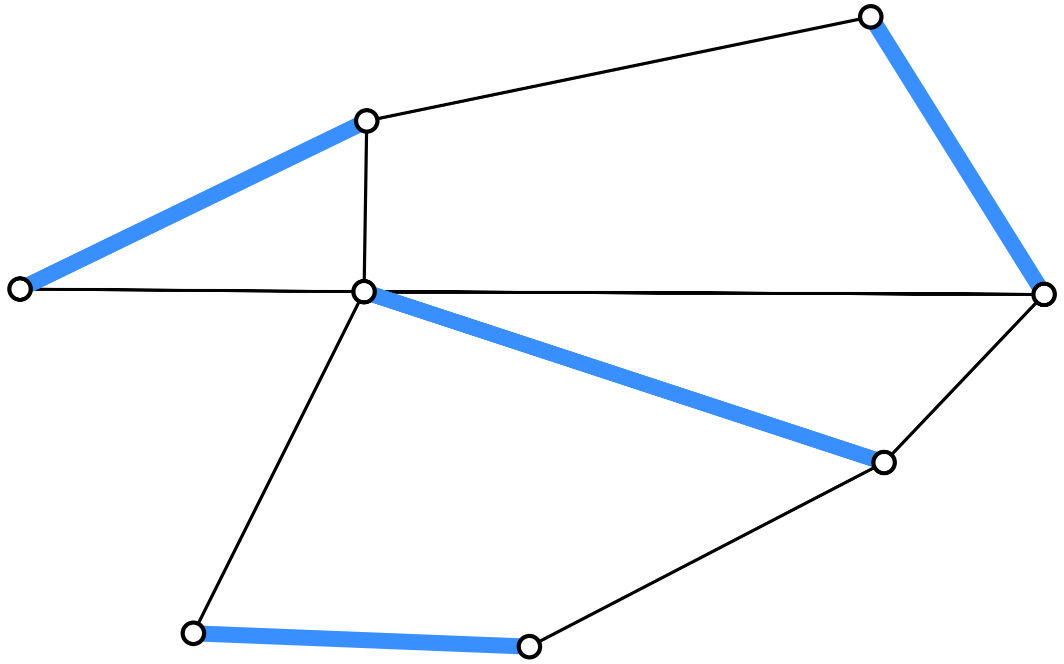
Matchings

An edge set $M \subseteq E$ of a graph $G = (V, E)$ is a matching if no two edges of M are adjacent (i.e., share an end vertex).

M is maximal if there is no matching M' with $M' \supsetneq M$.

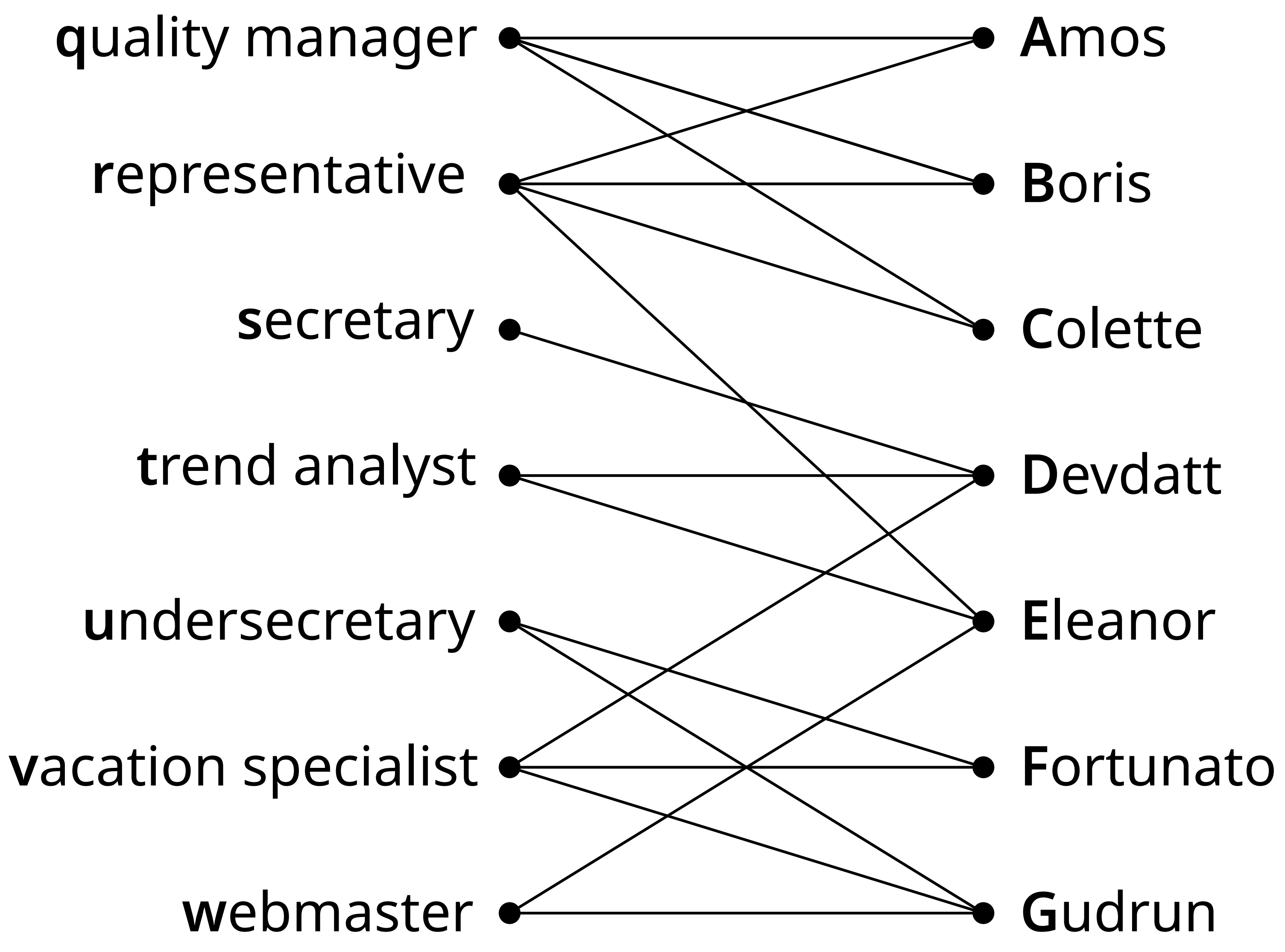
M is maximum if it has the largest number of edges among all matchings.

Remark: Not every matching can be extended to a maximum one.



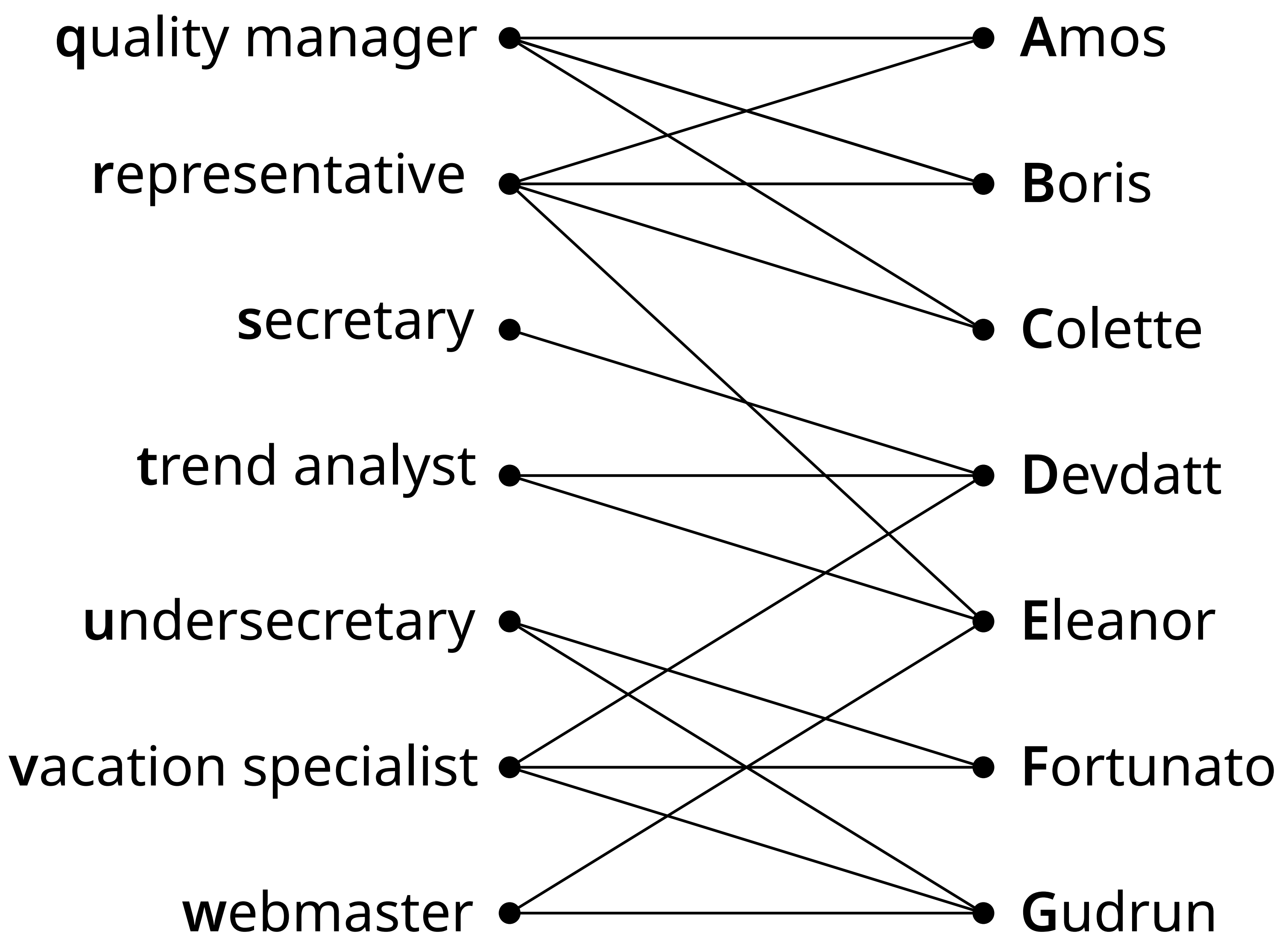
Perfect matching

A company is assigning workers to jobs. In the bipartite graph an edge connects a worker to a job they are willing to take.



Perfect matching

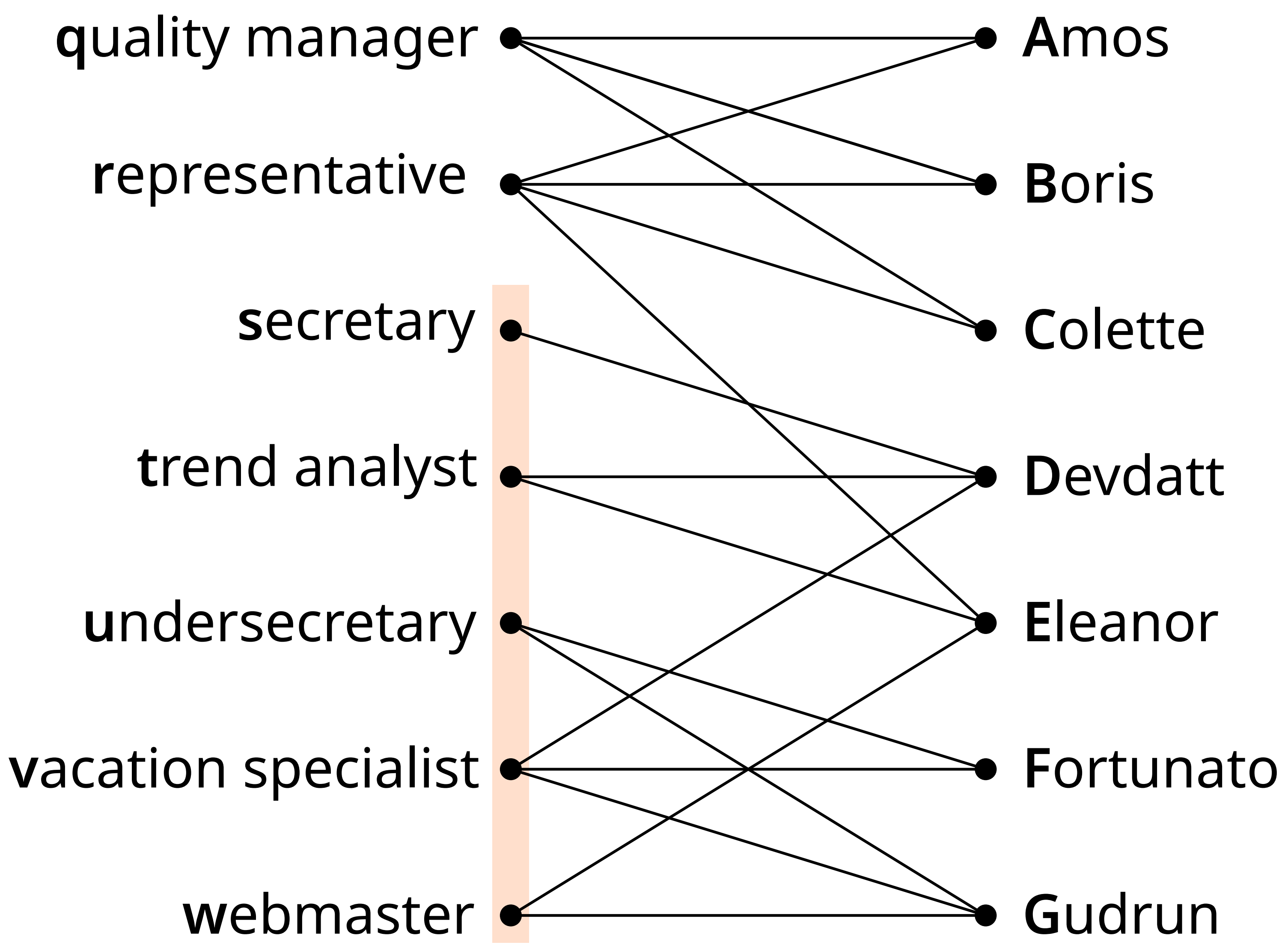
A company is assigning workers to jobs. In the bipartite graph an edge connects a worker to a job they are willing to take.



Can every person be matched to a job?

Perfect matching

A company is assigning workers to jobs. In the bipartite graph an edge connects a worker to a job they are willing to take.

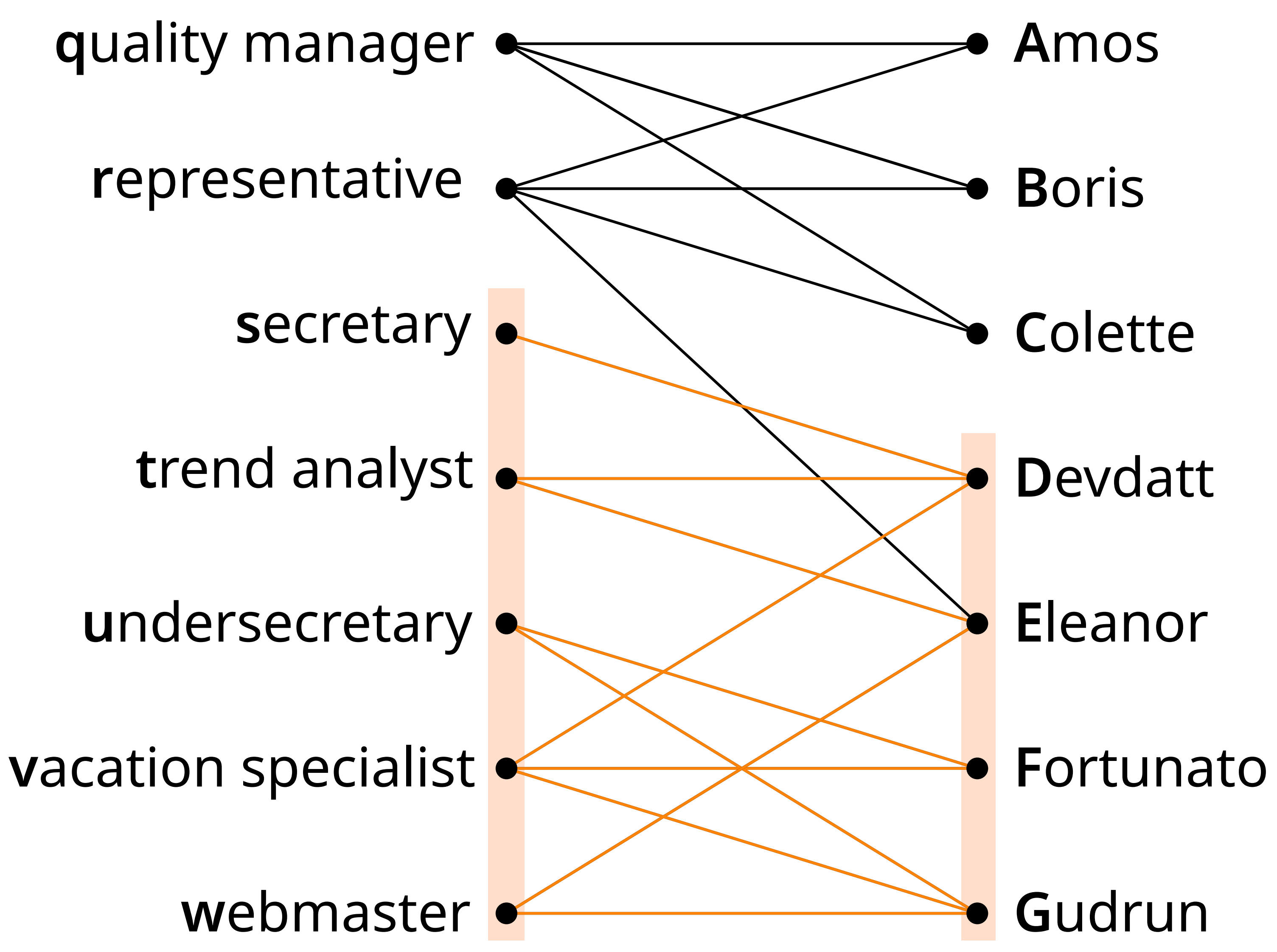


Neighborhood$\{s, t, u, v, w\} =$

Can every person be matched to a job? No! Why not?

Perfect matching

A company is assigning workers to jobs. In the bipartite graph an edge connects a worker to a job they are willing to take.



Neighborhood$\{s, t, u, v, w\} =$

$\{D, E, F, G\}$

Can every person be matched to a job? No! Why not?

Perfect matching

Hall's Theorem: Let G be a bipartite graph with bipartition $V = X \cup Y$.
Then G has a matching covering X $\Leftrightarrow$ $|\text{Neighborhood}(X')| \geq |X'|$

for all $X' \subseteq X$

due to Hall, in 1935.

Perfect matching

Hall's Theorem: Let G be a bipartite graph with bipartition $V = X \cup Y$.
Then G has a matching covering X $\Leftrightarrow$ $|\text{Neighborhood}(X')| \geq |X'|$

for all $X' \subseteq X$

due to Hall, in 1935.

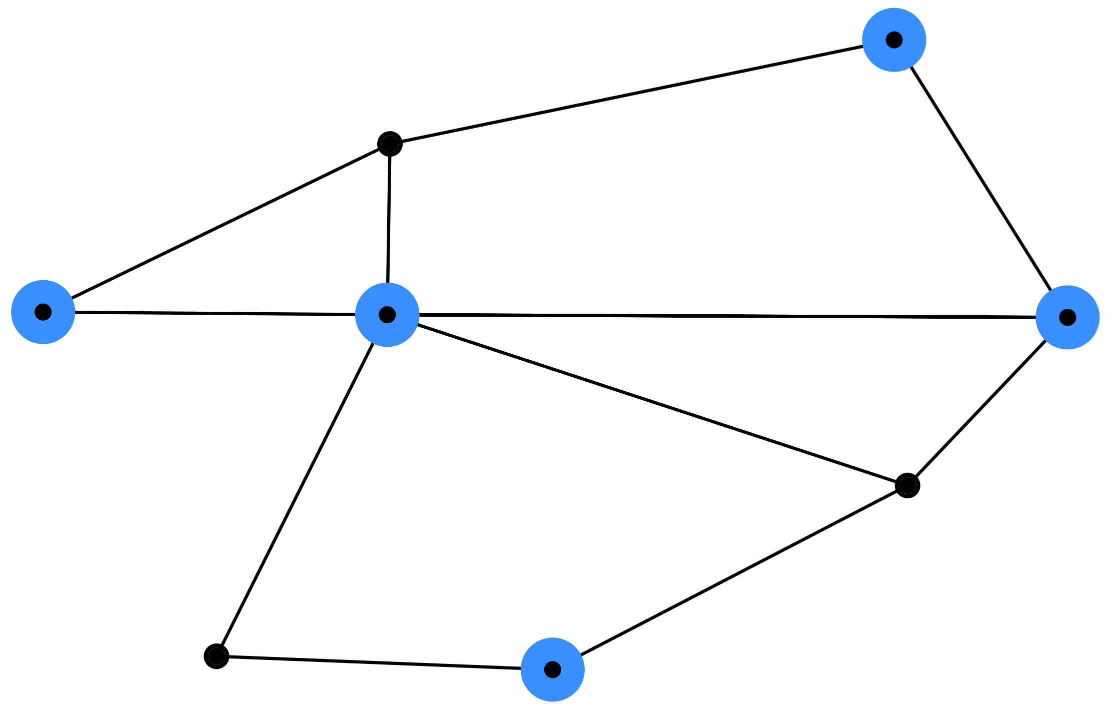
Proof: $\Rightarrow$ is clear.

$\Leftarrow$ will follow from König's Theorem.

Vertex Cover and König’s Theorem

König’s Theorem: In a bipartite graph the size of a minimum vertex cover equals the size of a maximum matching.

due to König, and independently, Egarvary, in 1931.

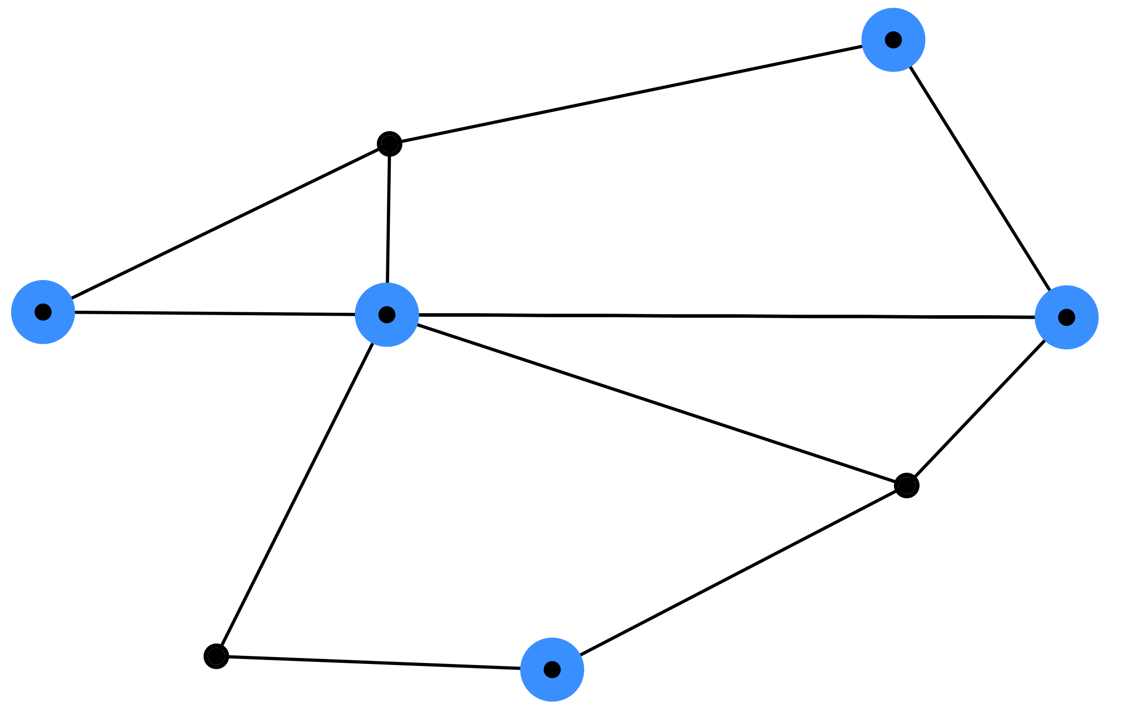


Vertex Cover and König's Theorem

König's Theorem: In a bipartite graph the size of a minimum vertex cover equals the size of a maximum matching.

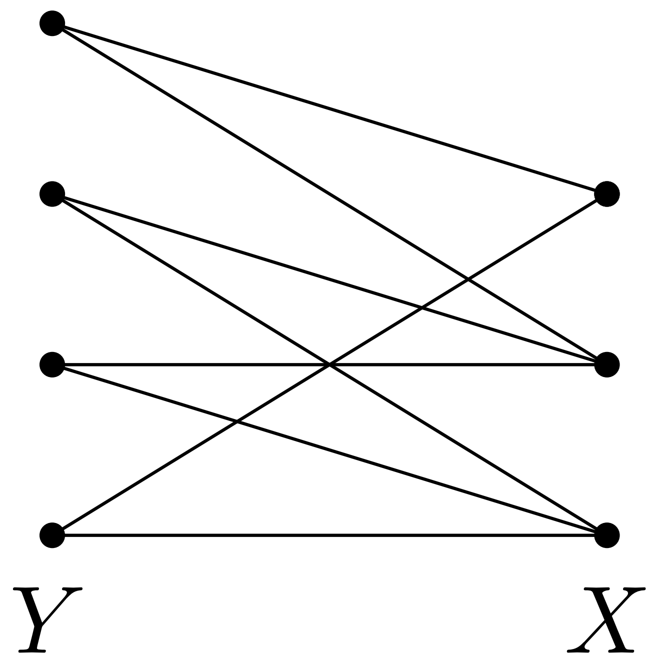
due to König, and independently, Egarvary, in 1931.

We will proof this using duality!



Proof of Hall's Theorem from König's Theorem

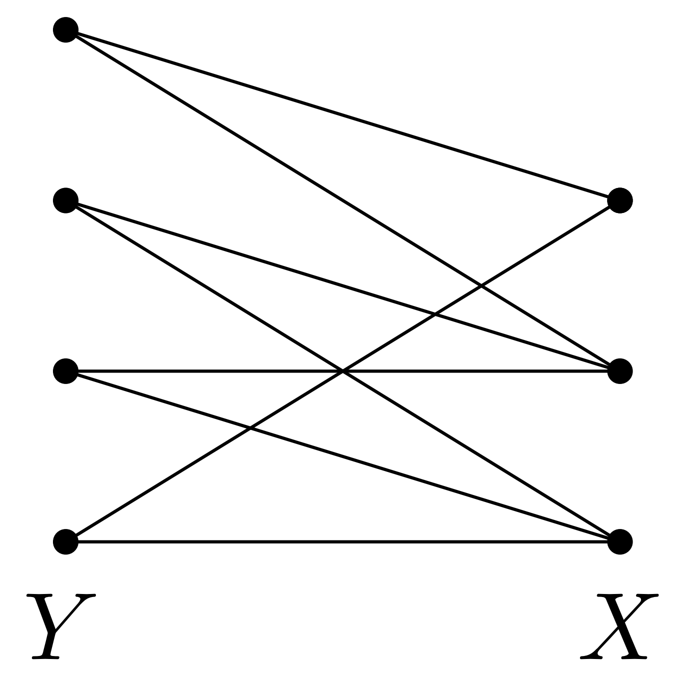
Let $G = (V, E)$ be a bipartite graph with bipartition $V = X \cup Y$.



Proof of Hall's Theorem from König's Theorem

Let $G = (V, E)$ be a bipartite graph with bipartition $V = X \cup Y$.

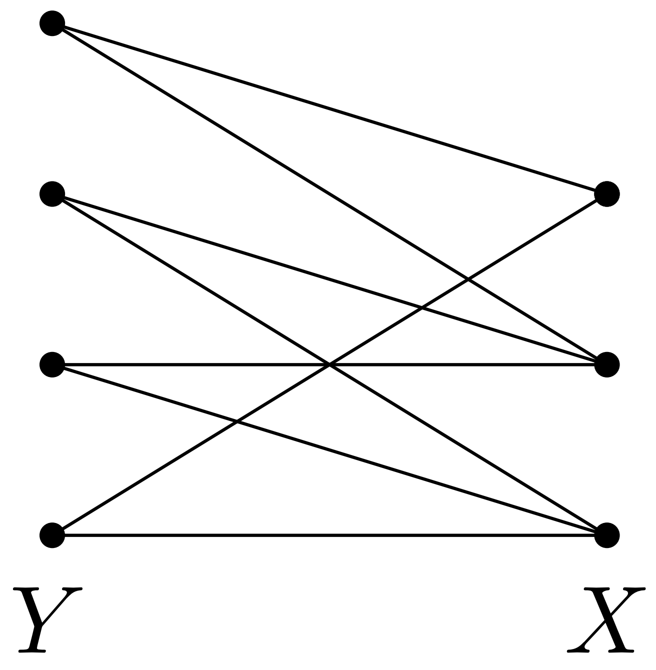
If $\left|\text{Neighborhood}(X')\right| \geq |X'|$ for all $X' \subseteq X$, we must show G has a matching covering X.



Proof of Hall's Theorem from König's Theorem

Let $G = (V, E)$ be a bipartite graph with bipartition $V = X \cup Y$.

If $|\text{Neighborhood}(X')| \geq |X'|$ for all $X' \subseteq X$, we must show G has a matching covering X. We'll show any vertex cover has size at least $|X|$, which by König's Theorem gives a matching of size $|X|$, which obviously covers X.



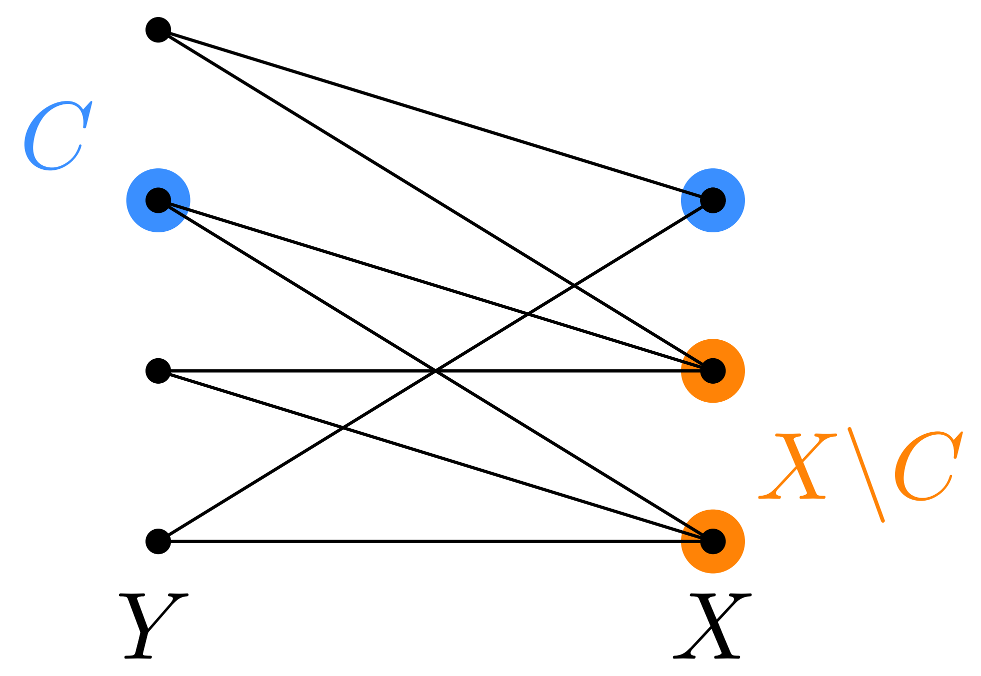
Proof of Hall's Theorem from König's Theorem

Let $G = (V, E)$ be a bipartite graph with bipartition $V = X \cup Y$.

If $|\text{Neighborhood}(X')| \geq |X'|$ for all $X' \subseteq X$, we must show G has a matching covering X. We'll show any vertex cover has size at least $|X|$, which by König's Theorem gives a matching of size $|X|$, which obviously covers X.

For a contradiction, suppose there is a vertex cover C with

- k vertices from X
- $< |X| - k$ vertices from Y



Proof of Hall's Theorem from König's Theorem

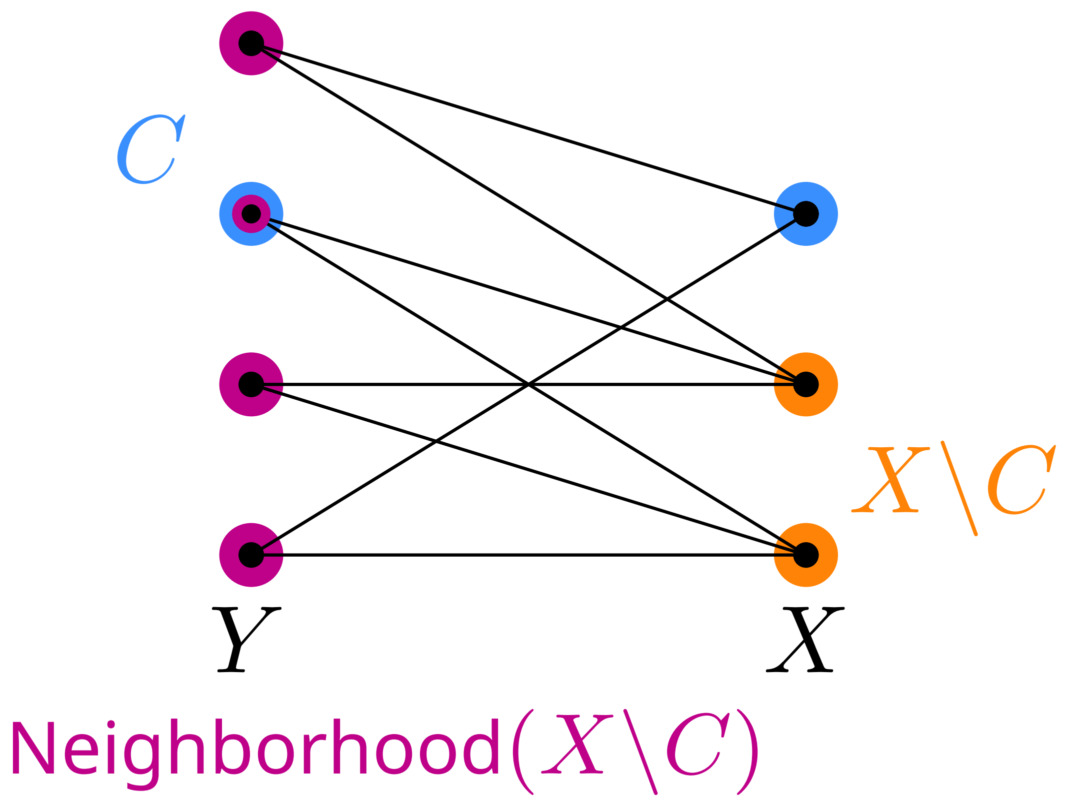
Let $G = (V, E)$ be a bipartite graph with bipartition $V = X \cup Y$.

If $|\text{Neighborhood}(X')| \geq |X'|$ for all $X' \subseteq X$, we must show G has a matching covering X. We'll show any vertex cover has size at least $|X|$, which by König's Theorem gives a matching of size $|X|$, which obviously covers X.

For a contradiction, suppose there is a vertex cover C with

- k vertices from X
- $< |X| - k$ vertices from Y

Then $X \backslash C$ has size $|X| - k$ and satisfies $|\text{Neighborhood}(X \backslash C)| \geq |X| - k$ by assumption.



Proof of Hall's Theorem from König's Theorem

Let $G = (V, E)$ be a bipartite graph with bipartition $V = X \cup Y$.

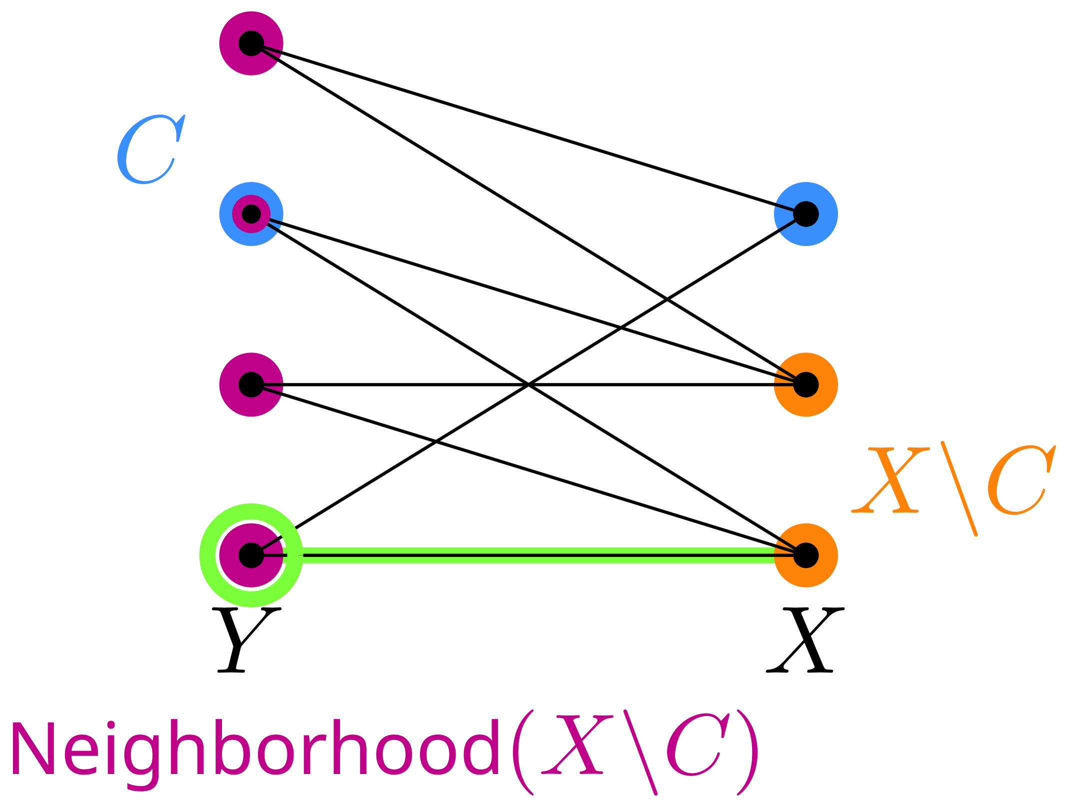
If $|\text{Neighborhood}(X')| \geq |X'|$ for all $X' \subseteq X$, we must show G has a matching covering X. We'll show any vertex cover has size at least $|X|$, which by König's Theorem gives a matching of size $|X|$, which obviously covers X.

For a contradiction, suppose there is a vertex cover C with

- k vertices from X
- $< |X| - k$ vertices from Y

Then $X \backslash C$ has size $|X| - k$ and satisfies $|\text{Neighborhood}(X \backslash C)| \geq |X| - k$ by assumption.

This gives a vertex in $\text{Neighborhood}(X \backslash C)$ that is not in $C \cap Y$ – showing C is not a cover.



Total Unimodular Matrices

A matrix is totally unimodular if every square submatrix has determinant $0, 1$, or -1.

Example

What does this matrix represent?

$$\begin{bmatrix} -1 & -1 & 0 & 0 & 0 & 1 \\ 1 & 0 & -1 & -1 & 0 & 0 \\ 0 & 1 & 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 & 1 & -1 \end{bmatrix}$$

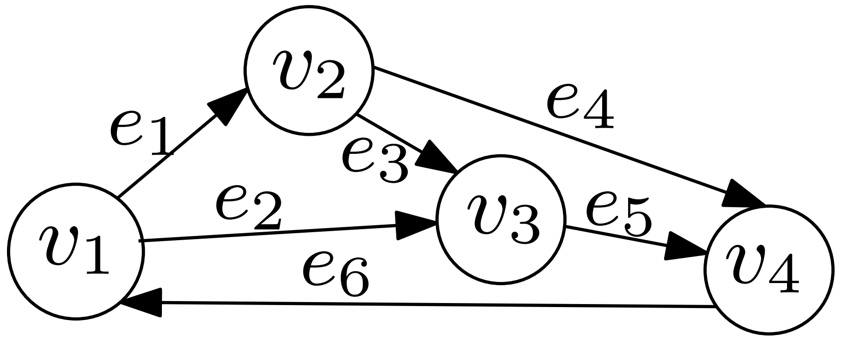
Total Unimodular Matrices

A matrix is totally unimodular if every square submatrix has determinant $0, 1$, or -1.

Example

$$\begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \end{array} \begin{array}{c} \begin{array}{cccccc} e_1 & e_2 & e_3 & e_4 & e_5 & e_6 \end{array} \\ \begin{bmatrix} -1 & -1 & 0 & 0 & 0 & 1 \\ 1 & 0 & -1 & -1 & 0 & 0 \\ 0 & 1 & 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 & 1 & -1 \end{bmatrix} \end{array}$$

What does this matrix represent?



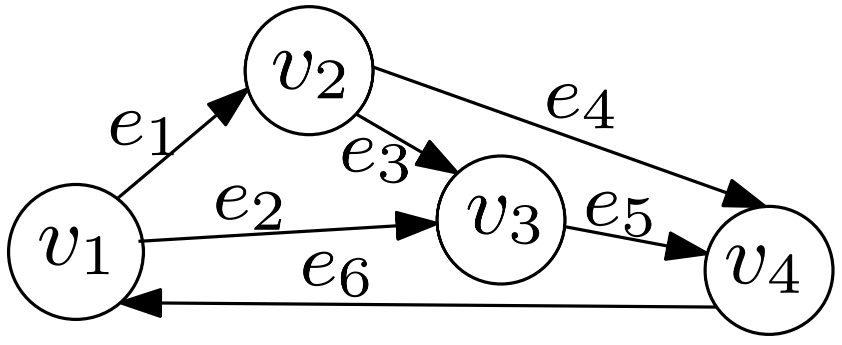
Total Unimodular Matrices

A matrix is totally unimodular if every square submatrix has determinant $0, 1$, or -1.

Example

$$\begin{array}{c c} & \begin{array}{cccccc} e_1 & e_2 & e_3 & e_4 & e_5 & e_6 \end{array} \\ \begin{array}{c} v_1 \\ v_2 \\ v_3 \\ v_4 \end{array} & \begin{bmatrix} -1 & -1 & 0 & 0 & 0 & 1 \\ 1 & 0 & -1 & -1 & 0 & 0 \\ 0 & 1 & 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 & 1 & -1 \end{bmatrix} \end{array}$$

What does this matrix represent?



In particular, each entry must be 0, -1, or 1.

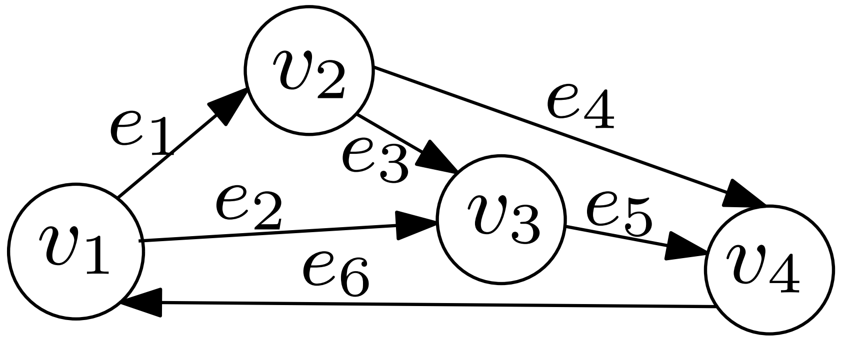
Total Unimodular Matrices

A matrix is totally unimodular if every square submatrix has determinant $0, 1$, or -1.

Example

$$\begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \end{array} \begin{array}{c} \begin{array}{cccccc} e_1 & e_2 & e_3 & e_4 & e_5 & e_6 \end{array} \\ \begin{bmatrix} -1 & -1 & 0 & 0 & 0 & 1 \\ 1 & 0 & -1 & -1 & 0 & 0 \\ 0 & 1 & 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 & 1 & -1 \end{bmatrix} \end{array}$$

What does this matrix represent?



In particular, each entry must be 0, -1, or 1. Why?

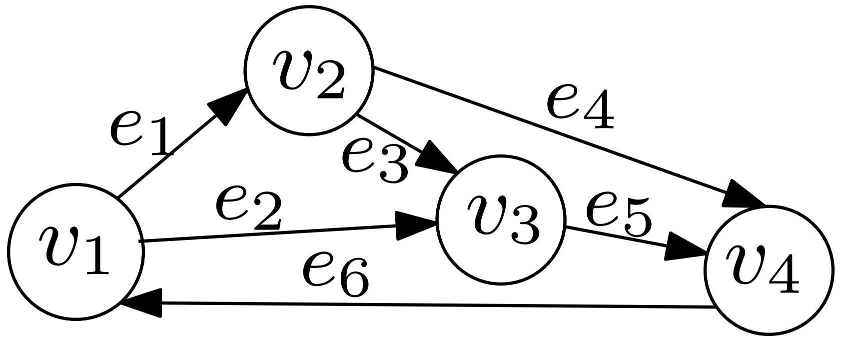
Total Unimodular Matrices

A matrix is totally unimodular if every square submatrix has determinant $0, 1$, or -1.

Example

$$\begin{array}{c|cccccc} & e_1 & e_2 & e_3 & e_4 & e_5 & e_6 \\ \hline v_1 & -1 & -1 & 0 & 0 & 0 & 1 \\ v_2 & 1 & 0 & -1 & -1 & 0 & 0 \\ v_3 & 0 & 1 & 1 & 0 & -1 & 0 \\ v_4 & 0 & 0 & 0 & 1 & 1 & -1 \end{array}$$

What does this matrix represent?



In particular, each entry must be $0, -1$, or 1.

Non-Example

A matrix of this form is not totally unimodular $\begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix}$

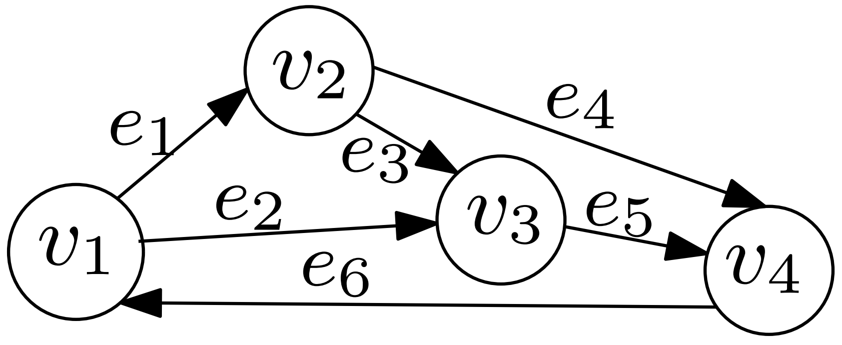
Total Unimodular Matrices

A matrix is totally unimodular if every square submatrix has determinant $0, 1$, or -1.

Example

$$\begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \end{array} \begin{array}{c} \begin{array}{cccccc} e_1 & e_2 & e_3 & e_4 & e_5 & e_6 \end{array} \\ \begin{bmatrix} -1 & -1 & 0 & 0 & 0 & 1 \\ 1 & 0 & -1 & -1 & 0 & 0 \\ 0 & 1 & 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 & 1 & -1 \end{bmatrix} \end{array}$$

What does this matrix represent?



In particular, each entry must be $0, -1$, or 1.

Non-Example A matrix of this form is not totally unimodular $\begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix}$

Example If matrix A is totally unimodular, then so is $\left[\begin{array}{cc} A & \begin{matrix} 0 \\ 0 \\ 1 \\ 0 \end{matrix} \end{array}\right]$

Total Unimodular Matrices

Theorem: Consider the linear program $\max c^T x$ subject to $Ax \leq b, x \geq 0$. If A is totally unimodular, if $b \in \mathbb{Z}^m$, and if there is an optimal solution, then there is an optimal integral solution $x^\star \in \mathbb{Z}^n$.

Total Unimodular Matrices

Theorem: Consider the linear program $\max c^T x$ subject to $Ax \leq b, x \geq 0$. If A is totally unimodular, if $b \in \mathbb{Z}^m$, and if there is an optimal solution, then there is an optimal integral solution $x^\star \in \mathbb{Z}^n$.

Proof: Bring in equational form: $\bar{A}\bar{x} = b$ with $\bar{A} = (A|I)$ and $\bar{x} \in \mathbb{R}^{n+m}$.

Total Unimodular Matrices

Theorem: Consider the linear program $\max c^T x$ subject to $Ax \leq b, x \geq 0$. If A is totally unimodular, if $b \in \mathbb{Z}^m$, and if there is an optimal solution, then there is an optimal integral solution $x^\star \in \mathbb{Z}^n$.

Proof: Bring in equational form: $\bar{A}\bar{x} = b$ with $\bar{A} = (A|I)$ and $\bar{x} \in \mathbb{R}^{n+m}$.

Solve via simplex method to find an optimal bfs $\bar{x}^\star$ corresponding to basis $B \subseteq \{1, 2, \ldots, n+m\}$ with nonzero entries solving $\bar{x}^\star_B = \bar{A}_B^{-1} b$.

Total Unimodular Matrices

Theorem: Consider the linear program $\max c^T x$ subject to $Ax \leq b, x \geq 0$. If A is totally unimodular, if $b \in \mathbb{Z}^m$, and if there is an optimal solution, then there is an optimal integral solution $x^\star \in \mathbb{Z}^n$.

Proof: Bring in equational form: $\bar{A}\bar{x} = b$ with $\bar{A} = (A|I)$ and $\bar{x} \in \mathbb{R}^{n+m}$.

Solve via simplex method to find an optimal bfs $\bar{x}^\star$ corresponding to basis $B \subseteq \{1, 2, \ldots, n+m\}$ with nonzero entries solving $\bar{x}^\star_B = \bar{A}_B^{-1} b$.

$\bar{A}$ totally unimodular
$\bar{A}$ nonsingular

What do we know about $\det(\bar{A})$?

Total Unimodular Matrices

Theorem: Consider the linear program $\max c^T x$ subject to $Ax \leq b, x \geq 0$. If A is totally unimodular, if $b \in \mathbb{Z}^m$, and if there is an optimal solution, then there is an optimal integral solution $x^\star \in \mathbb{Z}^n$.

Proof: Bring in equational form: $\bar{A}\bar{x} = b$ with $\bar{A} = (A|I)$ and $\bar{x} \in \mathbb{R}^{n+m}$.

Solve via simplex method to find an optimal bfs $\bar{x}^\star$ corresponding to basis $B \subseteq \{1, 2, \ldots, n+m\}$ with nonzero entries solving $\bar{x}^\star_B = \bar{A}_B^{-1} b$.

$\bar{A}$ totally unimodular $\Rightarrow$ $\det(\bar{A}_B) \in \{-1, 0, 1\}$.
$\bar{A}$ nonsingular $\Rightarrow$ $\det(\bar{A}_B) \in \{-1, 1\}$.

Total Unimodular Matrices

Theorem: Consider the linear program $\max c^T x$ subject to $Ax \leq b, x \geq 0$. If A is totally unimodular, if $b \in \mathbb{Z}^m$, and if there is an optimal solution, then there is an optimal integral solution $x^\star \in \mathbb{Z}^n$.

Proof: Bring in equational form: $\bar{A}\bar{x} = b$ with $\bar{A} = (A|I)$ and $\bar{x} \in \mathbb{R}^{n+m}$.

Solve via simplex method to find an optimal bfs $\bar{x}^\star$ corresponding to basis $B \subseteq \{1, 2, \ldots, n+m\}$ with nonzero entries solving $\bar{x}^\star_B = \bar{A}_B^{-1} b$.

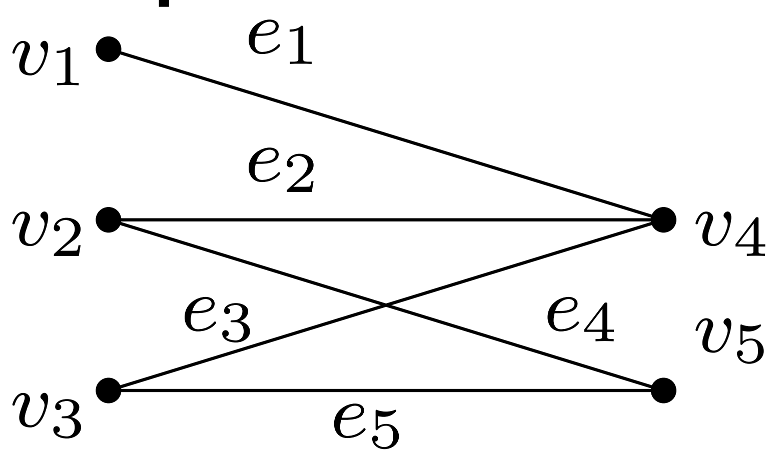
$\bar{A}$ totally unimodular $\Rightarrow$ det$(\bar{A}_B) \in \{-1, 0, 1\}$.
$\bar{A}$ nonsingular $\Rightarrow$ det$(\bar{A}_B) \in \{-1, 1\}$.

By Cramer's rule the coefficients of $\bar{x}^\star$ are rational numbers with denominator det(A_B), i.e., integer numbers! That is $\bar{x}^\star \in \mathbb{Z}^{n+m}$ and hence $x^\star \in \mathbb{Z}^n$.

Total Unimodularity and König’s Theorem

Lemma: The incidence matrix A of a bipartite graph $G = (X \cup Y, E)$ is totally unimodular.

Example



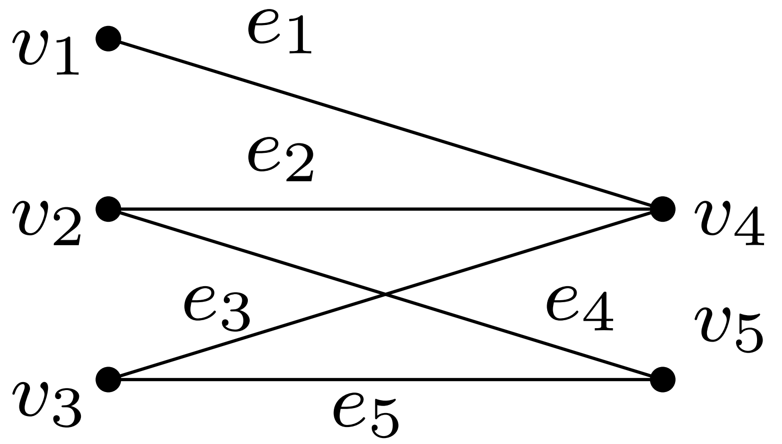
$$\begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \end{array} \begin{array}{c} \begin{array}{ccccc} e_1 & e_2 & e_3 & e_4 & e_5 \end{array} \\ \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} \end{array}$$

Total Unimodularity and König's Theorem

Lemma: The incidence matrix A of a bipartite graph $G = (X \cup Y, E)$ is totally unimodular.

Proof: We must show that ever $\ell \times \ell$ submatrix X of A has determinant $0, 1$, or -1. We show this by induction(ℓ).

Example



$$\begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \end{array} \begin{array}{c} \begin{array}{ccccc} e_1 & e_2 & e_3 & e_4 & e_5 \end{array} \\ \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} \end{array}$$

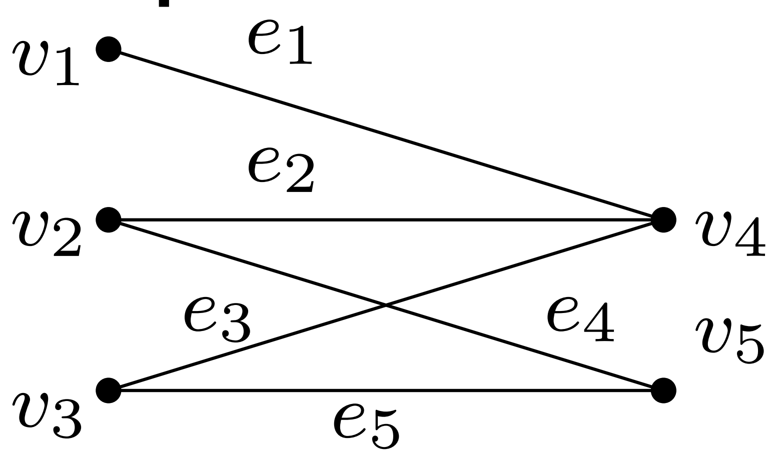
Total Unimodularity and König’s Theorem

Lemma: The incidence matrix A of a bipartite graph $G = (X \cup Y, E)$ is totally unimodular.

Proof: We must show that ever $\ell \times \ell$ submatrix X of A has determinant $0, 1$, or -1. We show this by induction(ℓ).

Base case $\ell = 1$: Clear by definition of incidence matrix.

Example



$$\begin{array}{c c}
 & \begin{array}{ccccc} e_1 & e_2 & e_3 & e_4 & e_5 \end{array} \\
\begin{array}{c} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \end{array} & \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix}
\end{array}$$

Total Unimodularity and König’s Theorem

Lemma: The incidence matrix A of a bipartite graph $G = (X \cup Y, E)$ is totally unimodular.

Proof: We must show that ever $\ell \times \ell$ submatrix X of A has determinant $0, 1$, or -1. We show this by induction(ℓ).

<u>Base case</u> $\ell = 1$: Clear by definition of incidence matrix.

<u>Inductive step:</u> Assume true for $\ell - 1$.

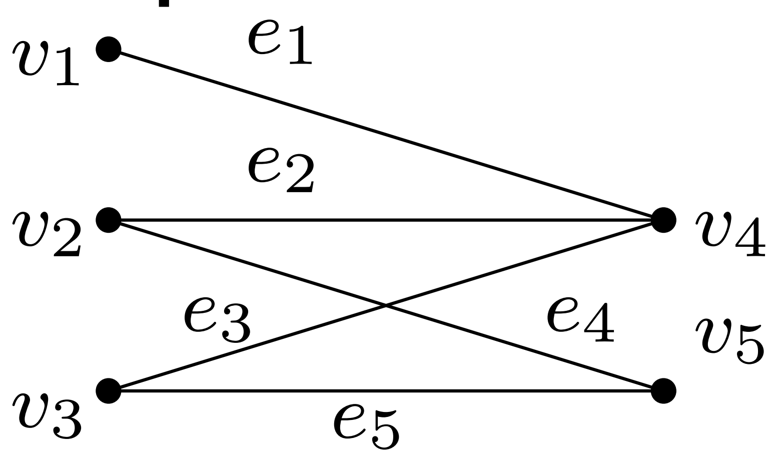
Let Q be an $\ell \times \ell$ submatrix.

If any column of Q is all zero, determinant is zero.

If any column of Q has one 1, true by induction.

Else all columns of Q have two 1.

Example



$$
\begin{array}{c c}
 & \begin{array}{ccccc} e_1 & e_2 & e_3 & e_4 & e_5 \end{array} \\
\begin{array}{c} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \end{array} &
\begin{bmatrix}
1 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 1 & 0 \\
0 & 0 & 1 & 0 & 1 \\
1 & 1 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 & 1
\end{bmatrix}
\end{array}
$$

Total Unimodularity and König's Theorem

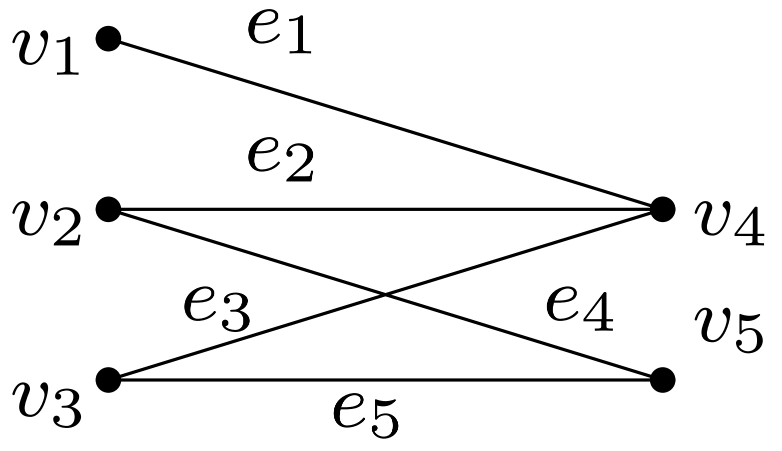
Lemma: The incidence matrix A of a bipartite graph $G = (X \cup Y, E)$ is totally unimodular.

Proof: We must show that ever $\ell \times \ell$ submatrix X of A has determinant $0, 1$, or -1. We show this by induction(ℓ).

Base case $\ell = 1$: Clear by definition of incidence matrix.

Inductive step: Assume true for $\ell - 1$.
Let Q be an $\ell \times \ell$ submatrix.
If any column of Q is all zero, determinant is zero.
If any column of Q has one 1, true by induction.
Else all columns of Q have two 1.
The sum of all rows for vertices in X gives $(1, 1, \ldots, 1)$.
The sum of all rows for vertices in Y gives $(1, 1, \ldots, 1)$.
Hence the rows of Q are linearly dep. and so $\det(Q) = 0$.

Example



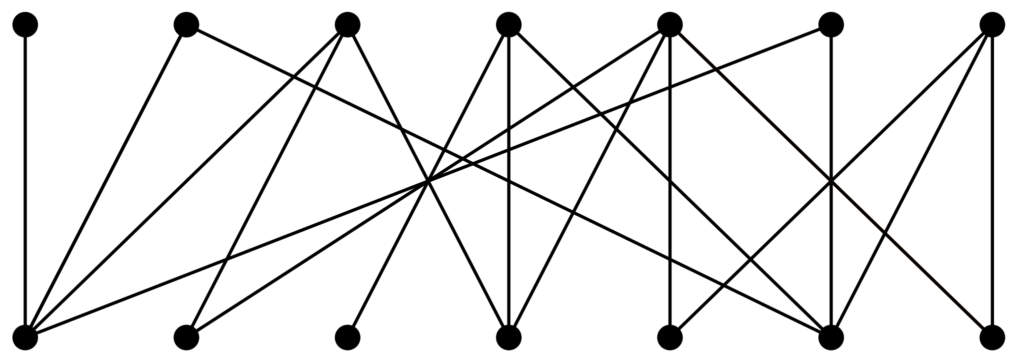
$$\begin{array}{c c} & \begin{array}{ccccc} e_1 & e_2 & e_3 & e_4 & e_5 \end{array} \\ \begin{array}{c} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \end{array} & \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} \end{array}$$

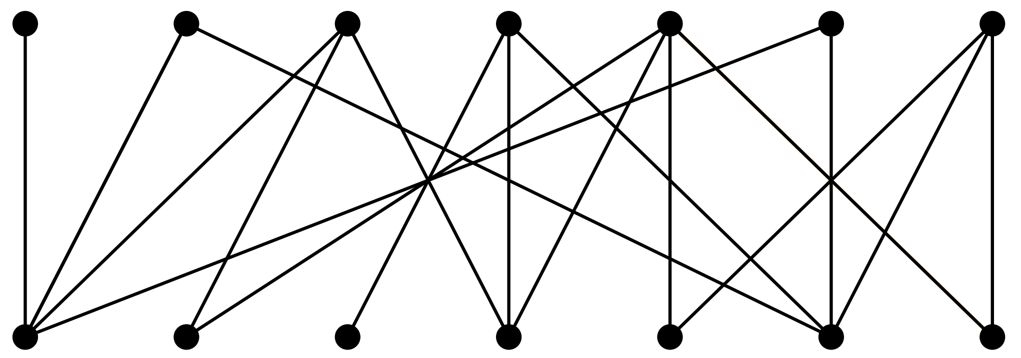
König’s Theorem

Theorem: In a bipartite graph the size of a minimum vertex cover equals the size of a maximum matching.

König's Theorem

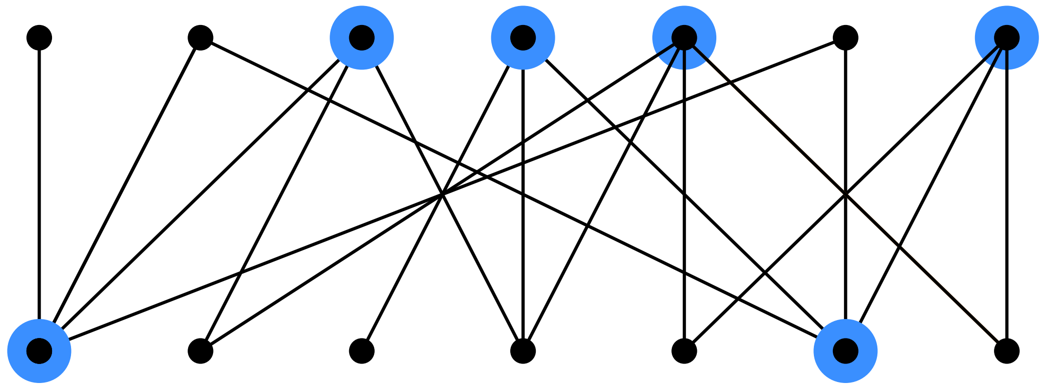
Theorem: In a bipartite graph the size of a minimum vertex cover equals the size of a maximum matching.



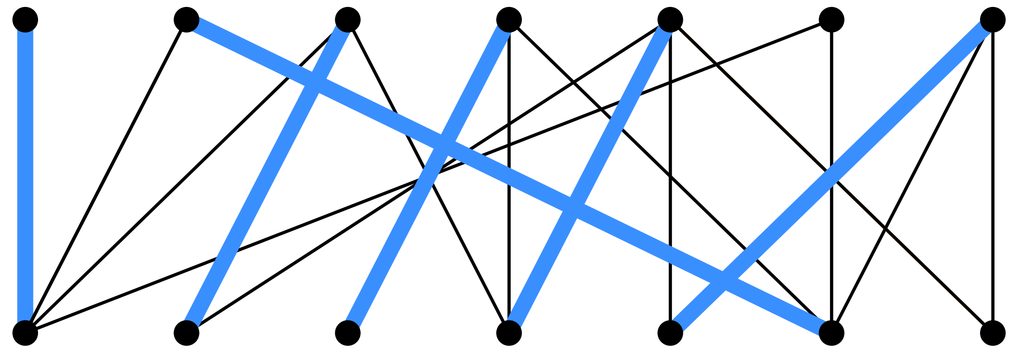


König’s Theorem

Theorem: In a bipartite graph the size of a minimum vertex cover equals the size of a maximum matching.



minimum vertex cover



maximum matching

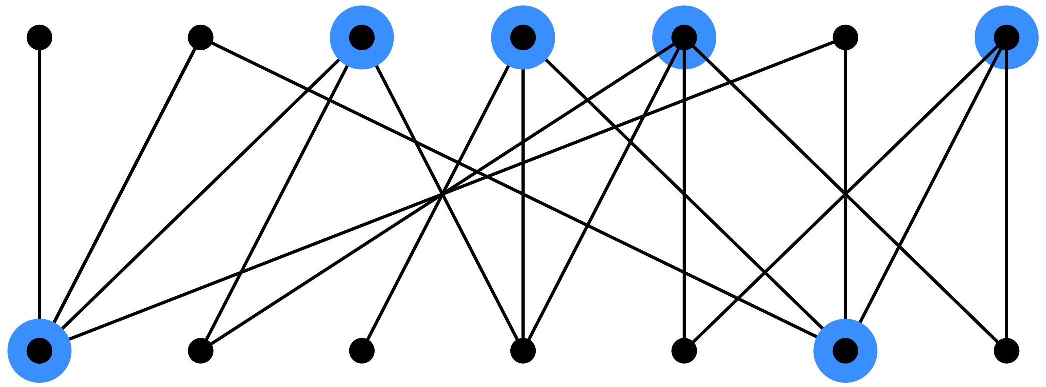
König’s Theorem

Theorem: In a bipartite graph the size of a minimum vertex cover equals the size of a maximum matching.

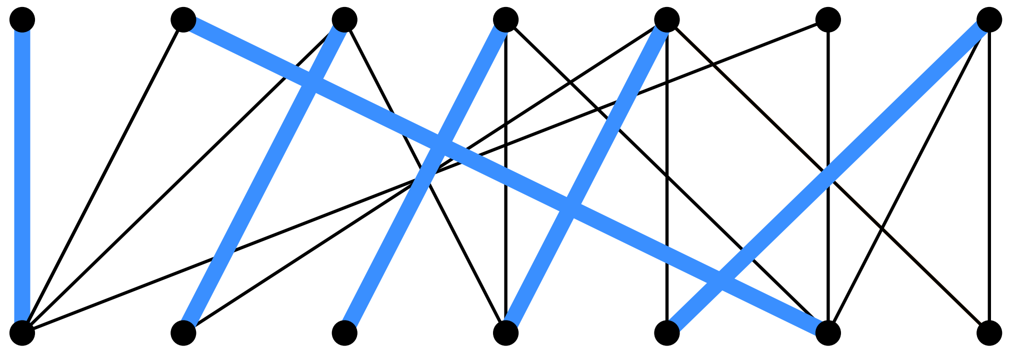
Proof:

Let A be the incidence matrix of the bipartite graph.

What are their LP’s?



minimum vertex cover



maximum matching

König's Theorem

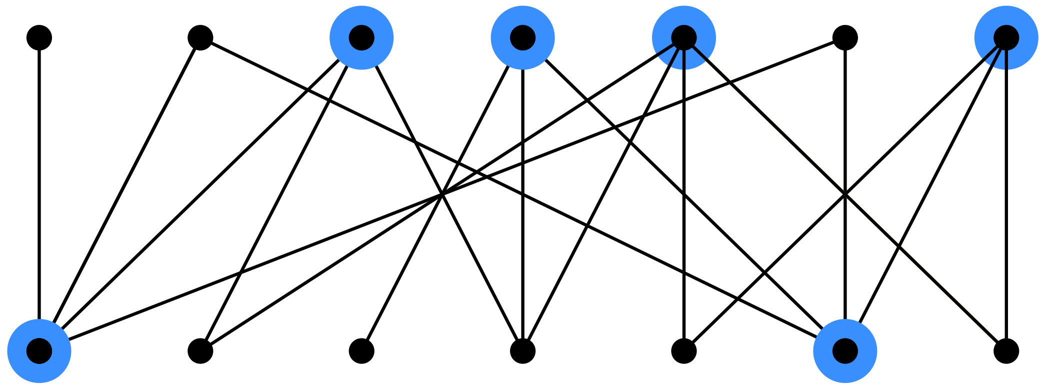
Theorem: In a bipartite graph the size of a minimum vertex cover equals the size of a maximum matching.

Proof:

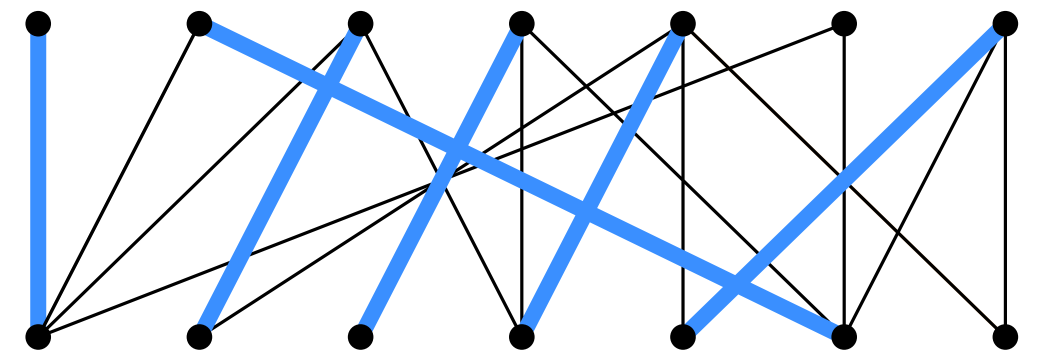
Let A be the incidence matrix of the bipartite graph.

Cover		Matching	
min	$\sum y_i$	max	$\sum x_j$
subject to	$A^T y \geq 1$	subject to	$Ax \leq 1$
	$y \geq 0$		$y \geq 0$

optionally: $y_i, x_j \in \{0, 1\}$



minimum vertex cover



maximum matching

König's Theorem

Theorem: In a bipartite graph the size of a minimum vertex cover equals the size of a maximum matching.

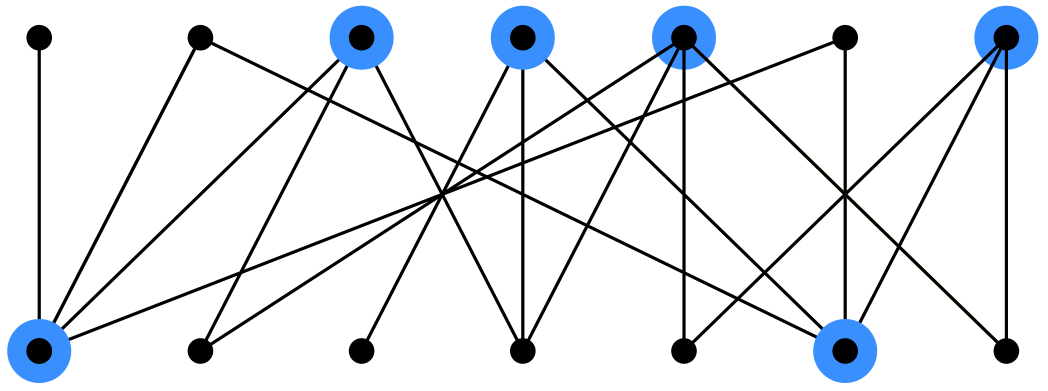
Proof:

Let A be the incidence matrix of the bipartite graph.

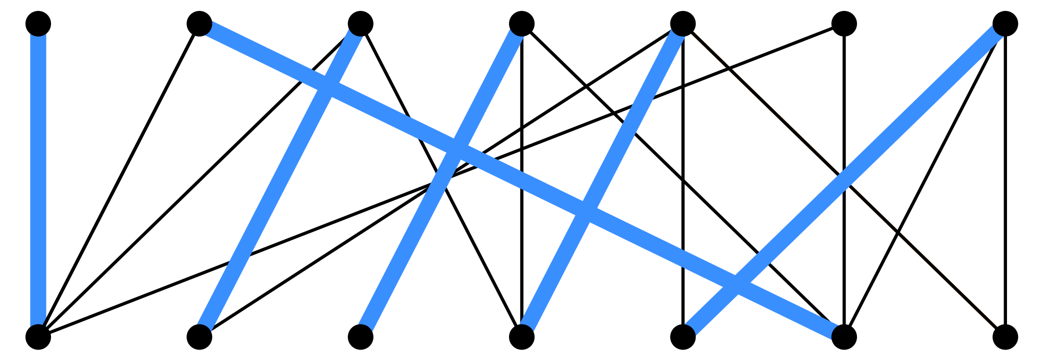
Cover		Matching	
min	$\sum y_i$	max	$\sum x_j$
subject to	$A^T y \geq 1$	subject to	$Ax \leq 1$
	$y \geq 0$		$y \geq 0$

optionally: $y_i, x_j \in \{0, 1\}$

We can drop the integrality constraints since A is totally unimodular.



minimum vertex cover



maximum matching

König's Theorem

Theorem: In a bipartite graph the size of a minimum vertex cover equals the size of a maximum matching.

Proof:

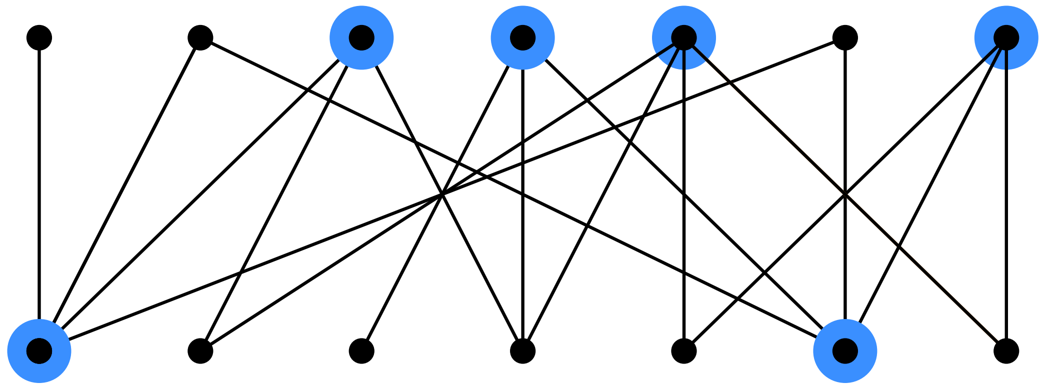
Let A be the incidence matrix of the bipartite graph.

Cover		Matching	
min	$\sum y_i$	max	$\sum x_j$
subject to	$A^T y \geq 1$	subject to	$Ax \leq 1$
	$y \geq 0$		$y \geq 0$

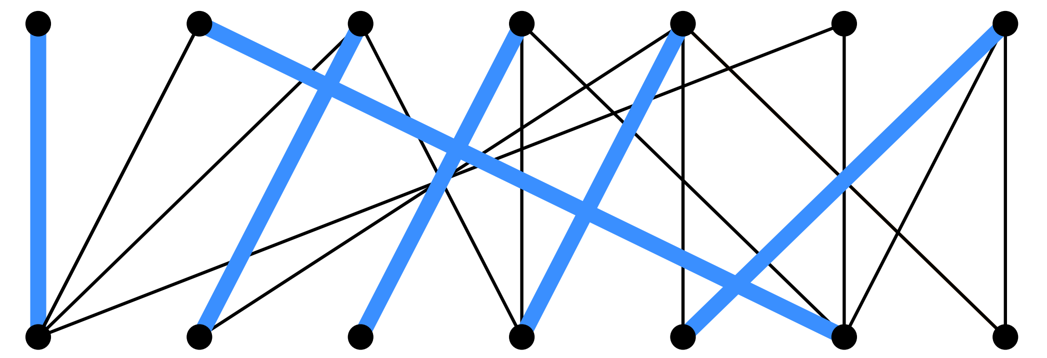
optionally: $y_i, x_j \in \{0, 1\}$

We can drop the integrality constraints since A is totally unimodular.

The duality of these linear programs then proves the theorem.



minimum vertex cover

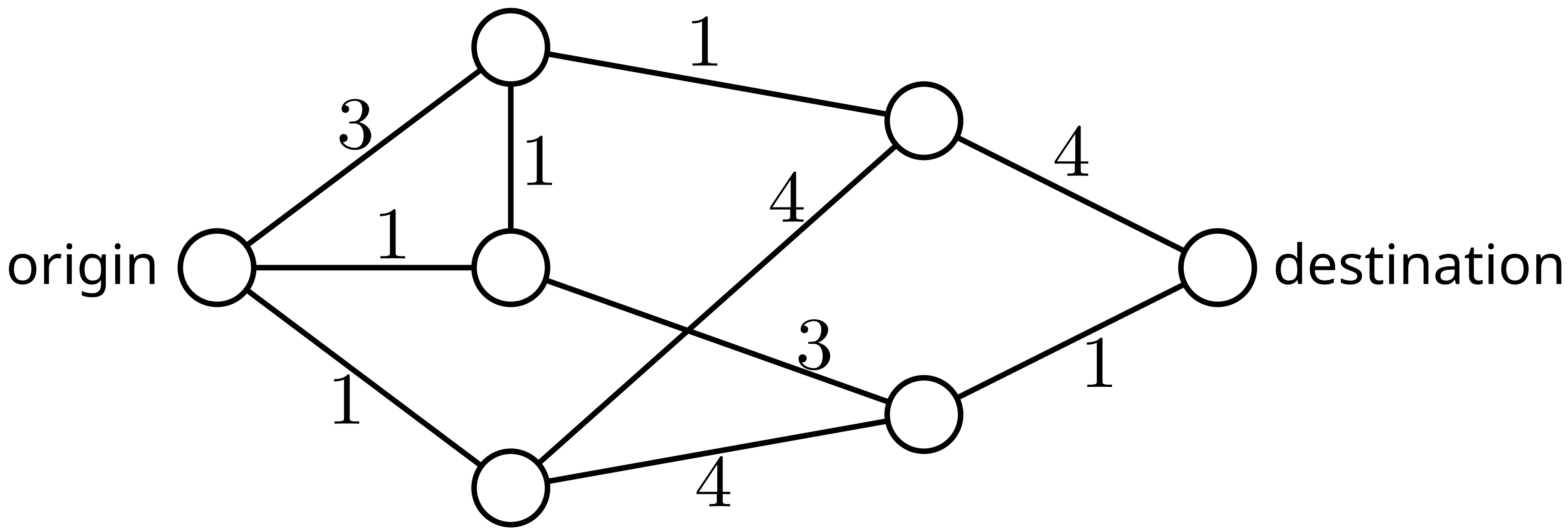


maximum matching

Duality shows Max Flow = Min Cut

Recall: Flow in a Network

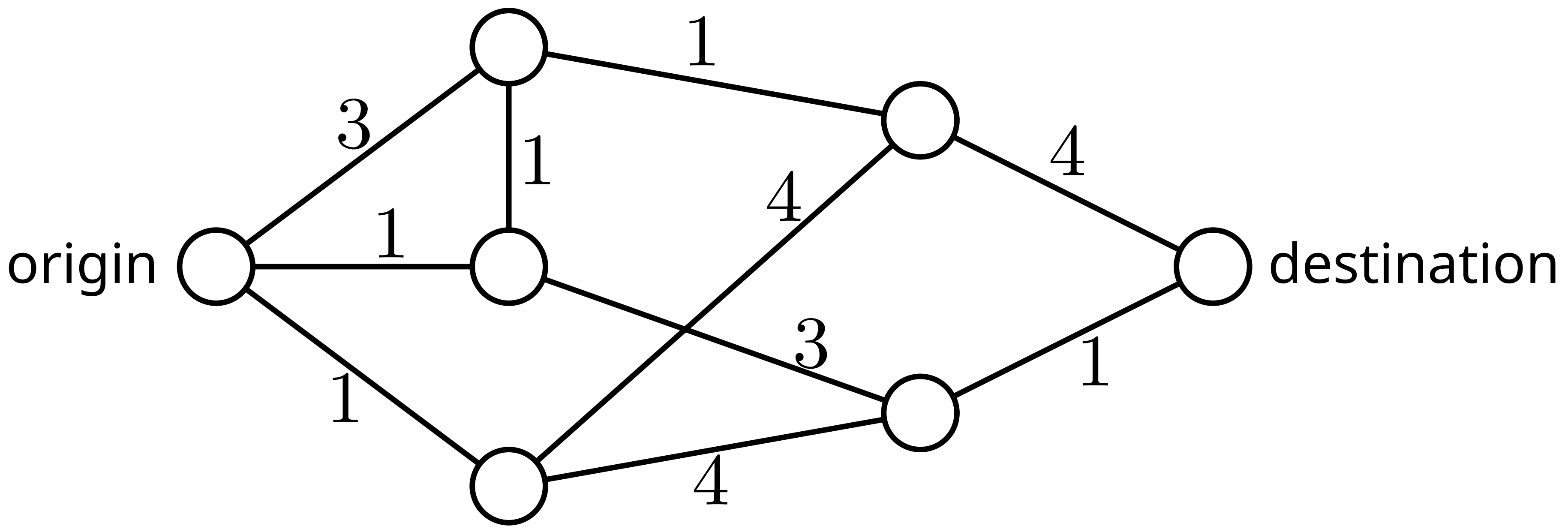
How to send as much data as possible over a local network?



nodes cannot store data and links can transport in only one direction

Recall: Flow in a Network

How to send as much data as possible over a local network?

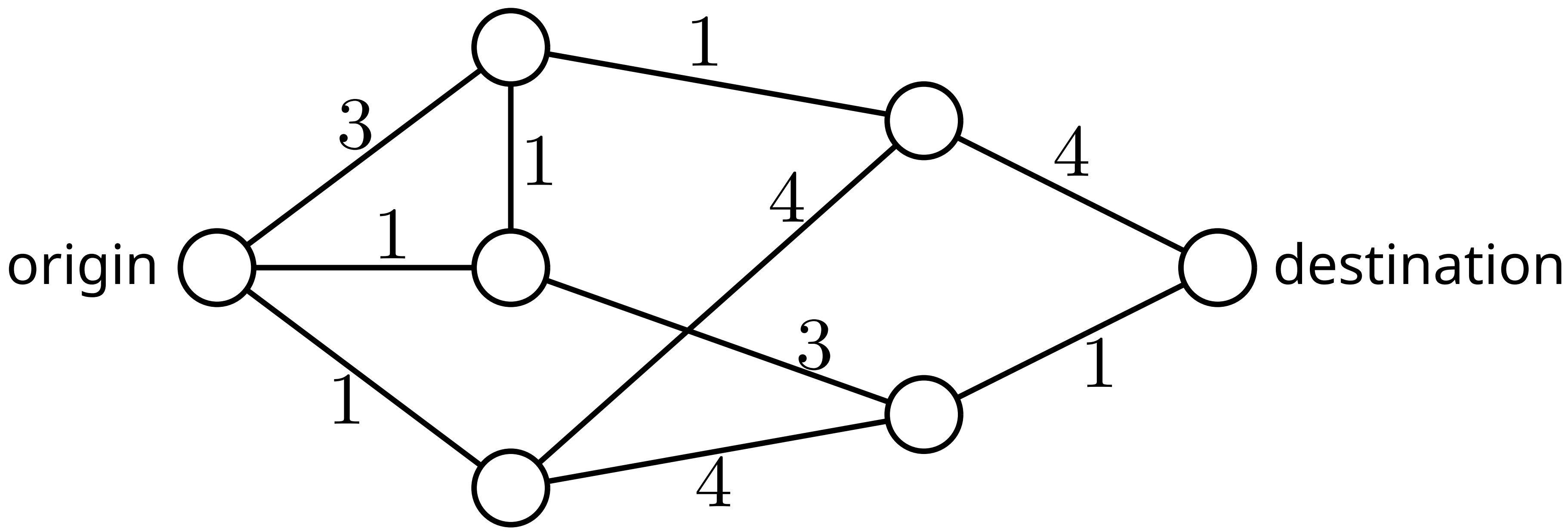


nodes cannot store data and links can transport in only one direction

$\rightarrow$ need to determine orientation and amount per edge (with direction)

Recall: Flow in a Network

How to send as much data as possible over a local network?



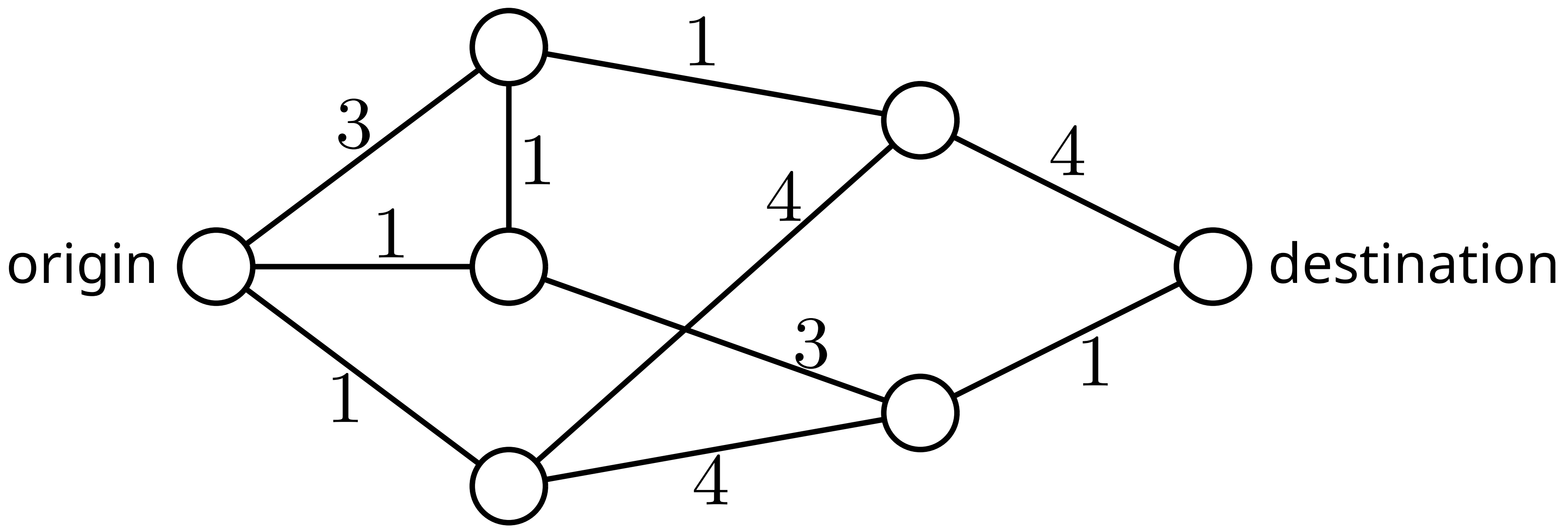
nodes cannot store data and links can transport in only one direction

$\rightarrow$ need to determine orientation and amount per edge (with direction)

LP formulation?

Recall: Flow in a Network

How to send as much data as possible over a local network?



nodes cannot store data and links can transport in only one direction

$\rightarrow$ need to determine orientation and amount per edge (with direction)

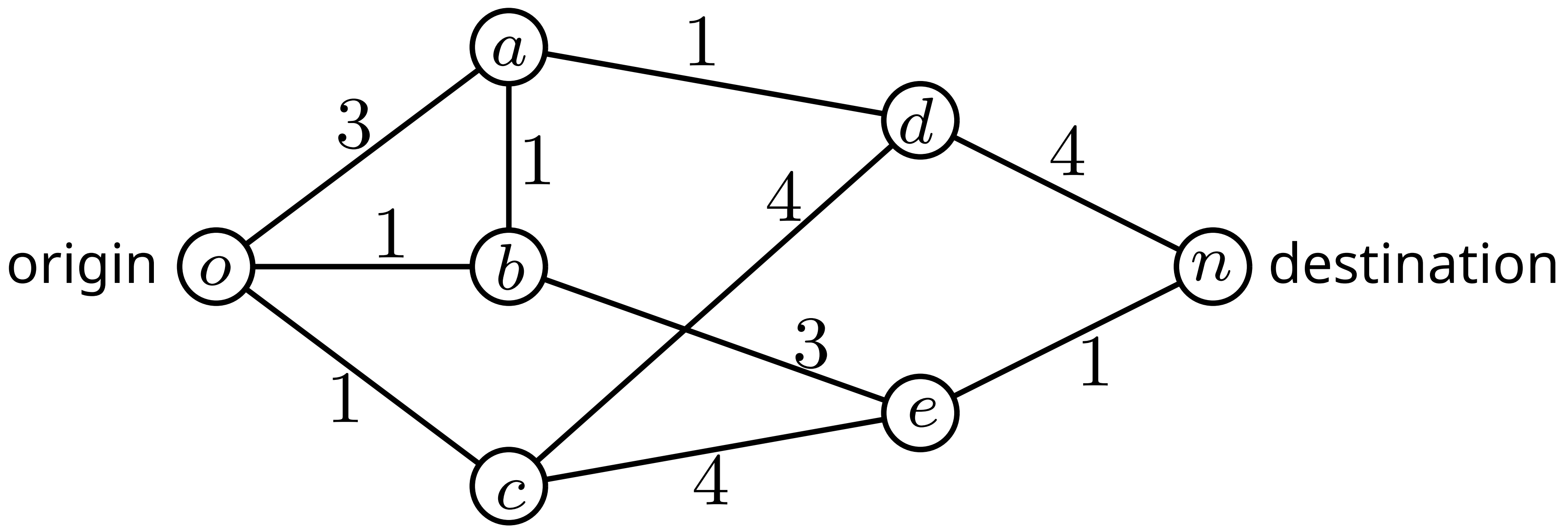
LP formulation?

$\rightarrow$ introduce variable x_{uv} for each edge (u, v) and require

1. flow $\leq$ capacities on edges
2. inflow = outflow on all nodes (except origin, destination)

Recall: Flow in a Network

How to send as much data as possible over a local network?



nodes cannot store data and links can transport in only one direction

$\rightarrow$ need to determine orientation and amount per edge (with direction)

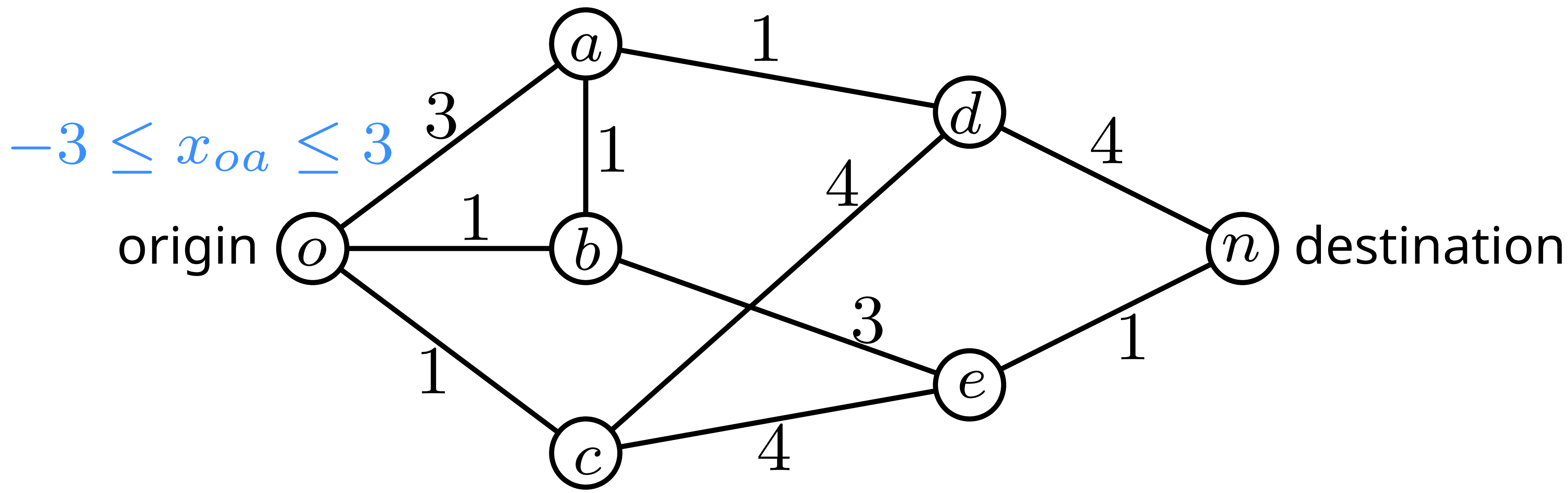
LP formulation?

$\rightarrow$ introduce variable x_{uv} for each edge (u, v) and require

1. flow $\leq$ capacities on edges
2. inflow = outflow on all nodes (except origin, destination)

Recall: Flow in a Network

How to send as much data as possible over a local network?



nodes cannot store data and links can transport in only one direction

$\to$ need to determine orientation and amount per edge (with direction)

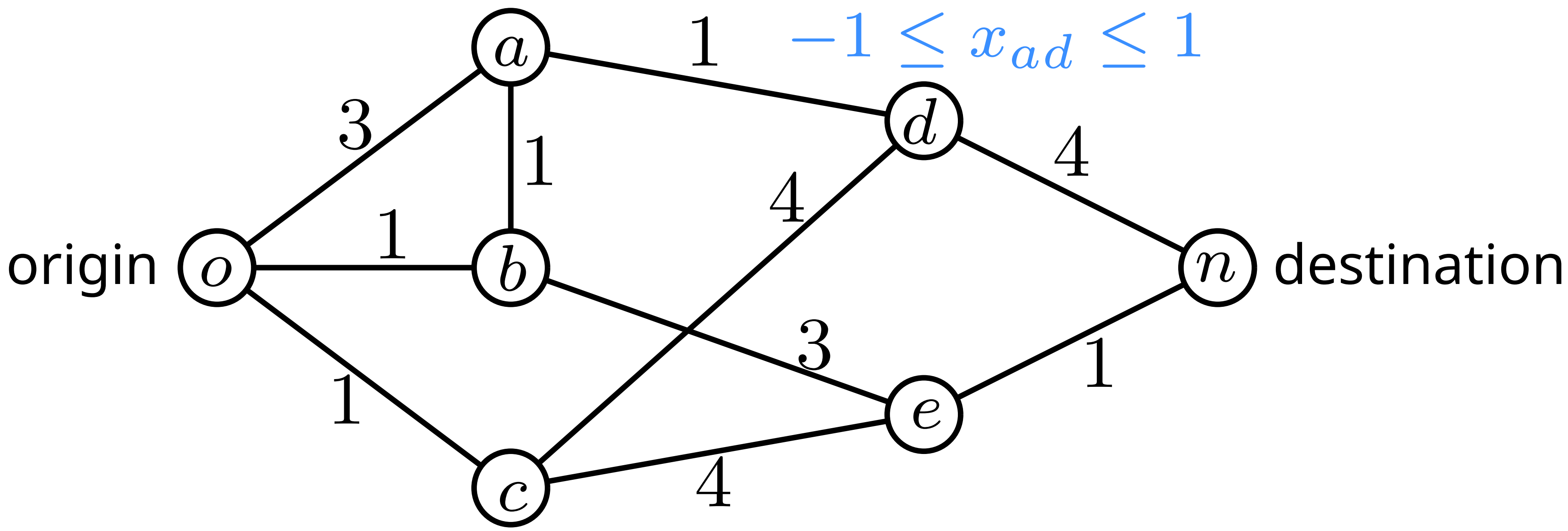
LP formulation?

$\to$ introduce variable x_{uv} for each edge (u, v) and require

1. flow $\leq$ capacities on edges
2. inflow = outflow on all nodes (except origin, destination)

Recall: Flow in a Network

How to send as much data as possible over a local network?



nodes cannot store data and links can transport in only one direction

$\rightarrow$ need to determine orientation and amount per edge (with direction)

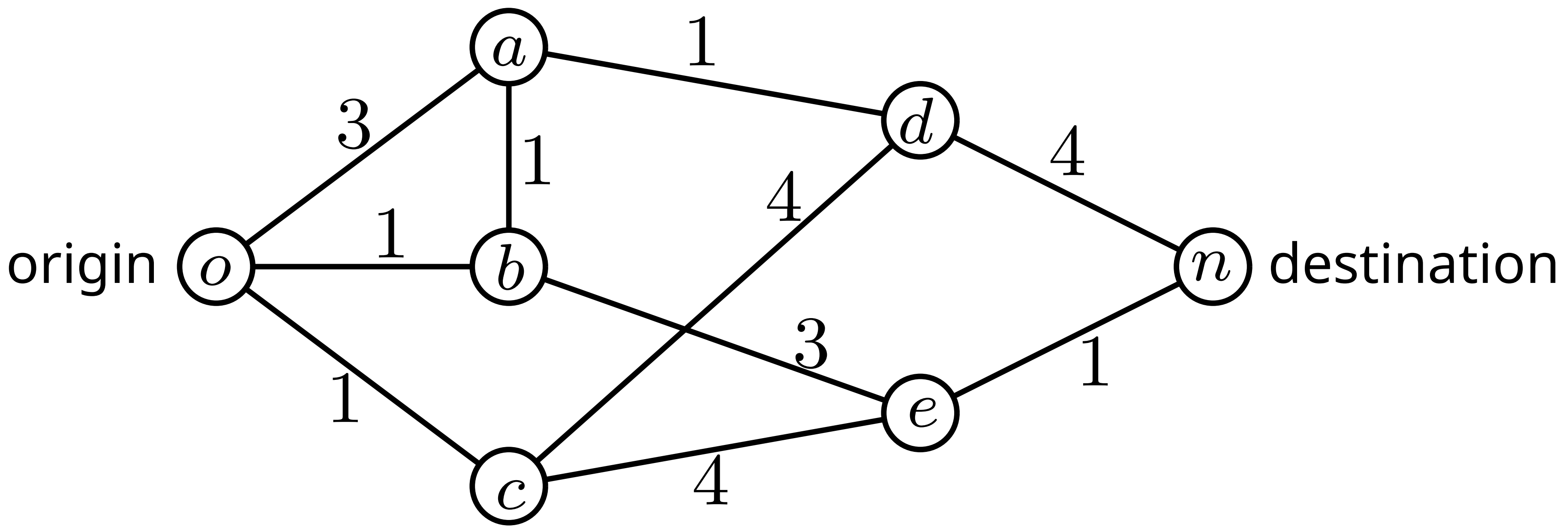
LP formulation?

$\rightarrow$ introduce variable x_{uv} for each edge (u, v) and require

1. flow $\leq$ capacities on edges
2. inflow = outflow on all nodes (except origin, destination)

Recall: Flow in a Network

How to send as much data as possible over a local network?



nodes cannot store data and links can transport in only one direction

$\rightarrow$ need to determine orientation and amount per edge (with direction)

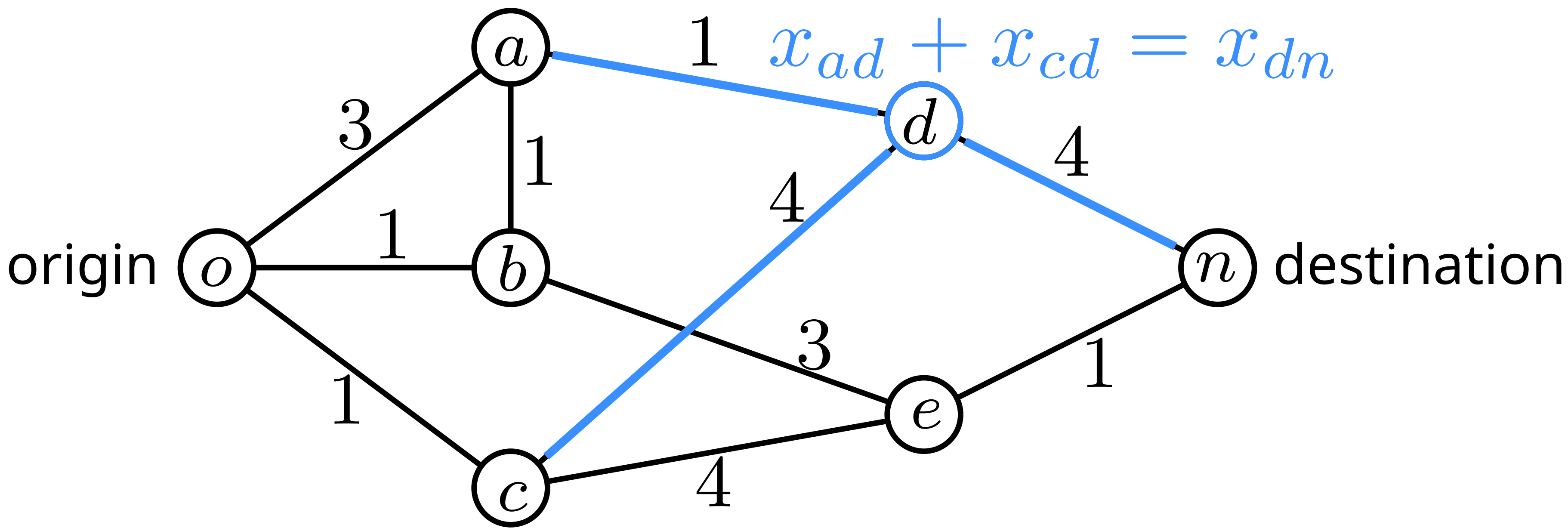
LP formulation?

$\rightarrow$ introduce variable x_{uv} for each edge (u, v) and require

1. flow $\leq$ capacities on edges how?
2. inflow = outflow on all nodes (except origin, destination)

Recall: Flow in a Network

How to send as much data as possible over a local network?



nodes cannot store data and links can transport in only one direction

$\rightarrow$ need to determine orientation and amount per edge (with direction)

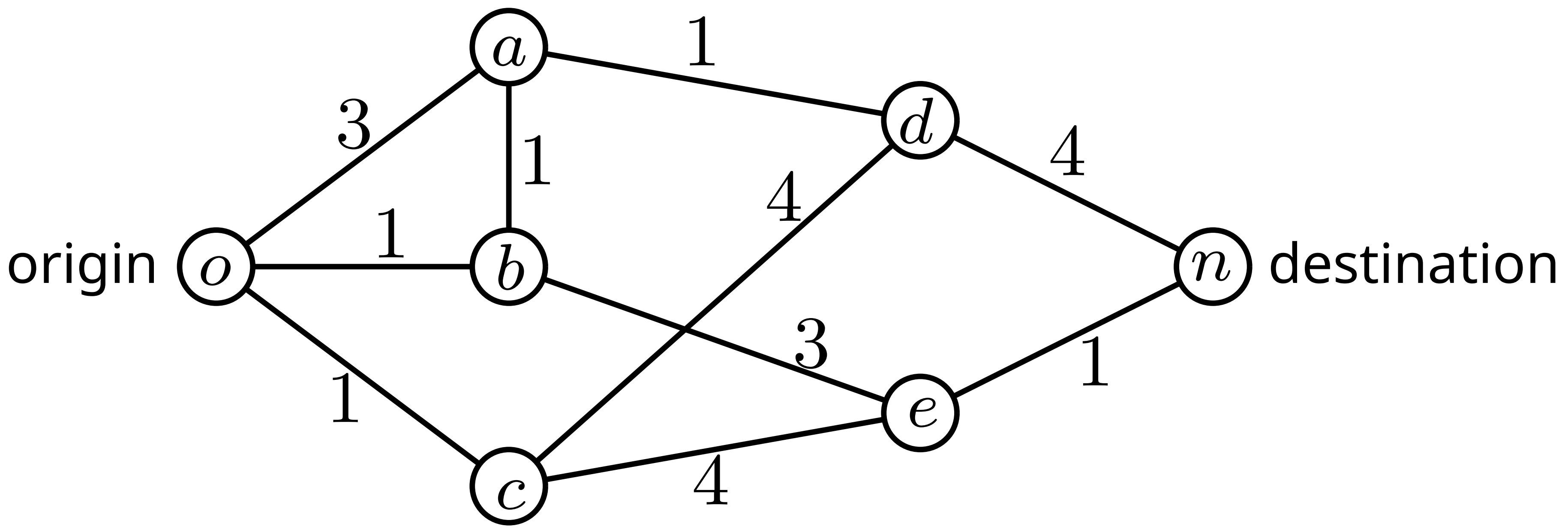
LP formulation?

$\rightarrow$ introduce variable x_{uv} for each edge (u, v) and require

1. flow $\leq$ capacities on edges how?
2. inflow = outflow on all nodes (except origin, destination)

Recall: Flow in a Network

How to send as much data as possible over a local network?



Maximize . . .?

nodes cannot store data and links can transport in only one direction

$\to$ need to determine orientation and amount per edge (with direction)

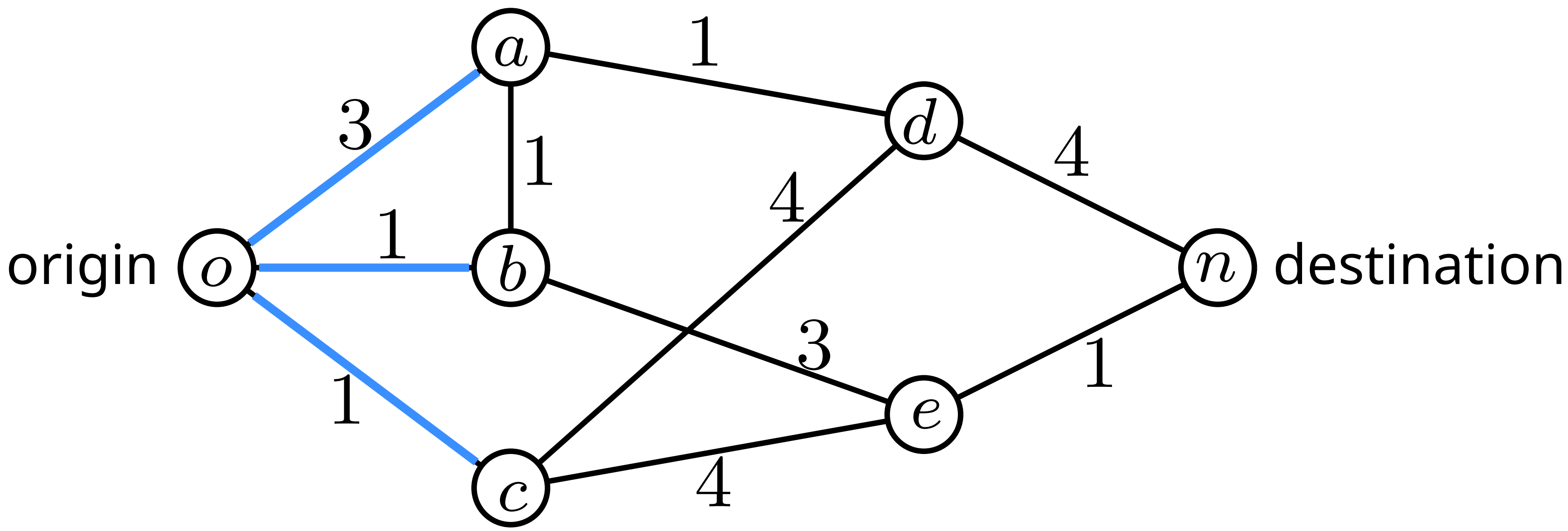
LP formulation?

$\to$ introduce variable x_{uv} for each edge (u, v) and require

1. flow $\leq$ capacities on edges how?
2. inflow = outflow on all nodes (except origin, destination)

Recall: Flow in a Network

How to send as much data as possible over a local network?



Maximize . . .?

$x_{oa} + x_{ob} + x_{oc}$

nodes cannot store data and links can transport in only one direction

$\to$ need to determine orientation and amount per edge (with direction)

LP formulation?

$\to$ introduce variable x_{uv} for each edge (u, v) and require

1. flow $\leq$ capacities on edges how?
2. inflow = outflow on all nodes (except origin, destination)

Recall: Flow in a Network

Linear Program Formulation

maximize $x_{oa} + x_{ob} + x_{oc}$

subject to

$$-3 \leq x_{oa} \leq 3,\ -1 \leq x_{ob} \leq 1,\ -1 \leq x_{oc} \leq 1$$

$$-1 \leq x_{ab} \leq 1,\ -1 \leq x_{ad} \leq 1,\ -3 \leq x_{be} \leq 3$$

$$-4 \leq x_{cd} \leq 4,\ -4 \leq x_{ce} \leq 4,\ -4 \leq x_{dn} \leq 4$$

$$-1 \leq x_{en} \leq 1$$

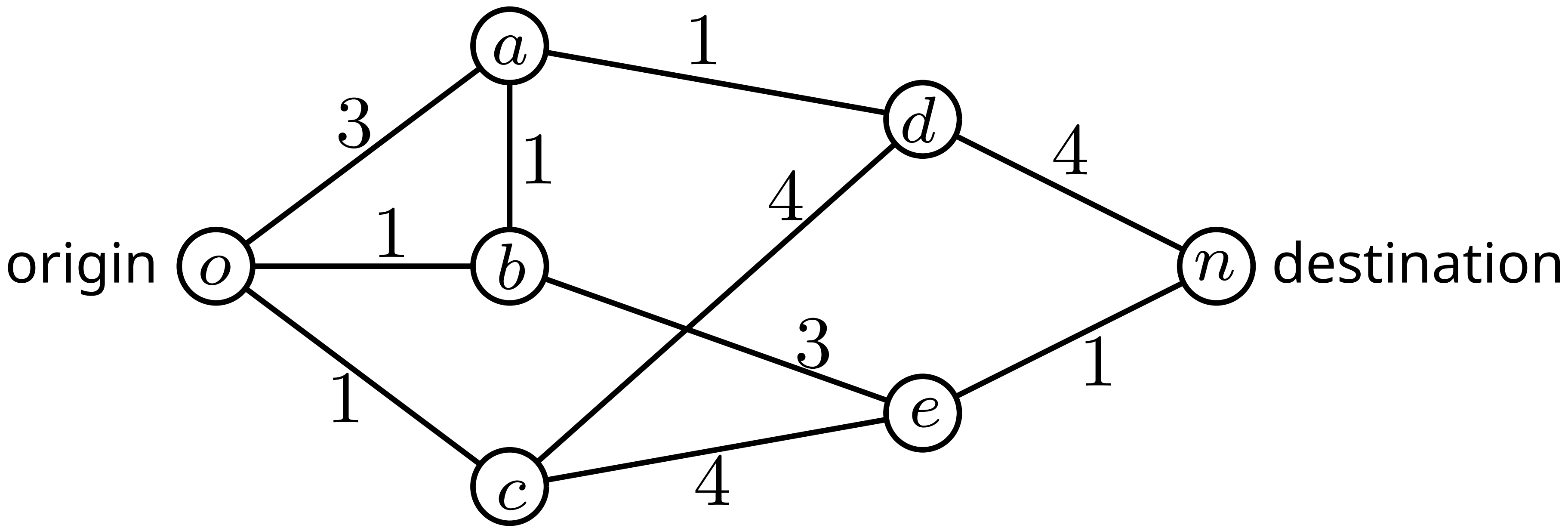
$$x_{oa} = x_{ab} + x_{ad}$$

$$x_{ob} + x_{ab} = x_{be}$$

$$x_{oc} = x_{cd} + x_{ce}$$

$$x_{ad} + x_{cd} = x_{dn}$$

$$x_{be} + x_{ce} = x_{en}$$



Recall: Flow in a Network

Linear Program Formulation

maximize $x_{oa} + x_{ob} + x_{oc}$

subject to

$$-3 \leq x_{oa} \leq 3,\ -1 \leq x_{ob} \leq 1,\ -1 \leq x_{oc} \leq 1$$

$$-1 \leq x_{ab} \leq 1,\ -1 \leq x_{ad} \leq 1,\ -3 \leq x_{be} \leq 3$$

$$-4 \leq x_{cd} \leq 4,\ -4 \leq x_{ce} \leq 4,\ -4 \leq x_{dn} \leq 4$$

$$-1 \leq x_{en} \leq 1$$

$$x_{oa} = x_{ab} + x_{ad}$$

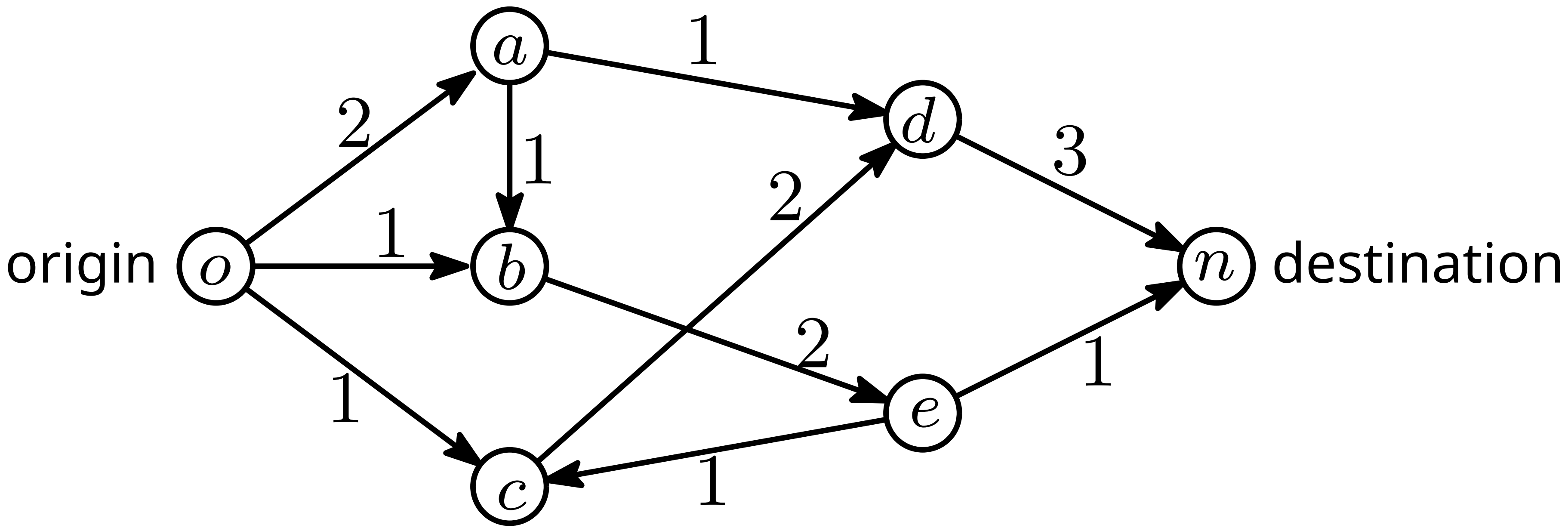
$$x_{ob} + x_{ab} = x_{be}$$

$$x_{oc} = x_{cd} + x_{ce}$$

$$x_{ad} + x_{cd} = x_{dn}$$

$$x_{be} + x_{ce} = x_{en}$$

Optimal solution: 4

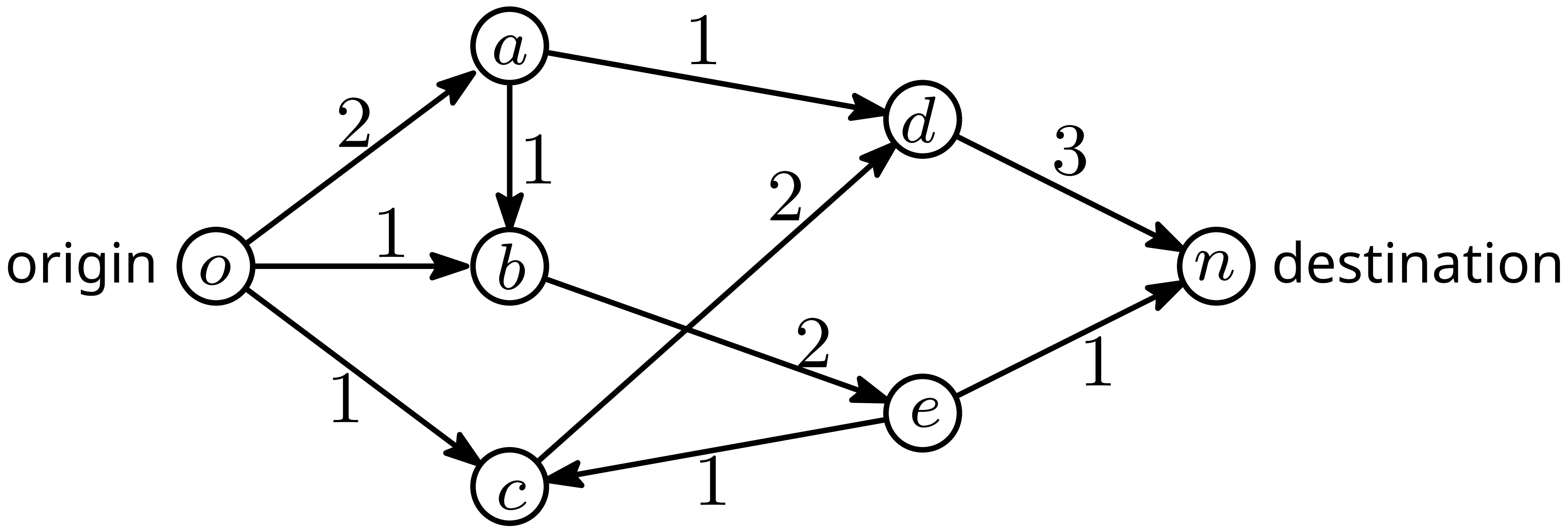


Recall: Flow in a Network

Linear Program Formulation

$$
\begin{aligned}
\text{maximize } & x_{oa} + x_{ob} + x_{oc} \\
\text{subject to } & -3 \le x_{oa} \le 3,\ -1 \le x_{ob} \le 1,\ -1 \le x_{oc} \le 1 \\
& -1 \le x_{ab} \le 1,\ -1 \le x_{ad} \le 1,\ -3 \le x_{be} \le 3 \\
& -4 \le x_{cd} \le 4,\ -4 \le x_{ce} \le 4,\ -4 \le x_{dn} \le 4 \\
& -1 \le x_{en} \le 1 \\
& x_{oa} = x_{ab} + x_{ad} \\
& x_{ob} + x_{ab} = x_{be} \\
& x_{oc} = x_{cd} + x_{ce} \\
& x_{ad} + x_{cd} = x_{dn} \\
& x_{be} + x_{ce} = x_{en}
\end{aligned}
$$

Optimal solution: 4



well-known “max flow = min cut” $\longrightarrow$ now via LP-duality!

Recall: Flow in a Network

Linear Program Formulation

maximize $x_{oa} + x_{ob} + x_{oc}$

subject to $-3 \leq x_{oa} \leq 3,\ -1 \leq x_{ob}$

$-1 \leq x_{ab} \leq 1,\ -1 \leq x_{ad} \leq 1,\ -3 \leq x_{be} \leq 3$

$-4 \leq x_{cd} \leq 4,\ -4 \leq x_{ce} \leq 4,\ -4 \leq x_{dn} \leq 4$

$-1 \leq x_{en} \leq 1$

$x_{oa} = x_{ab} + x_{ad}$

$x_{ob} + x_{ab} = x_{be}$

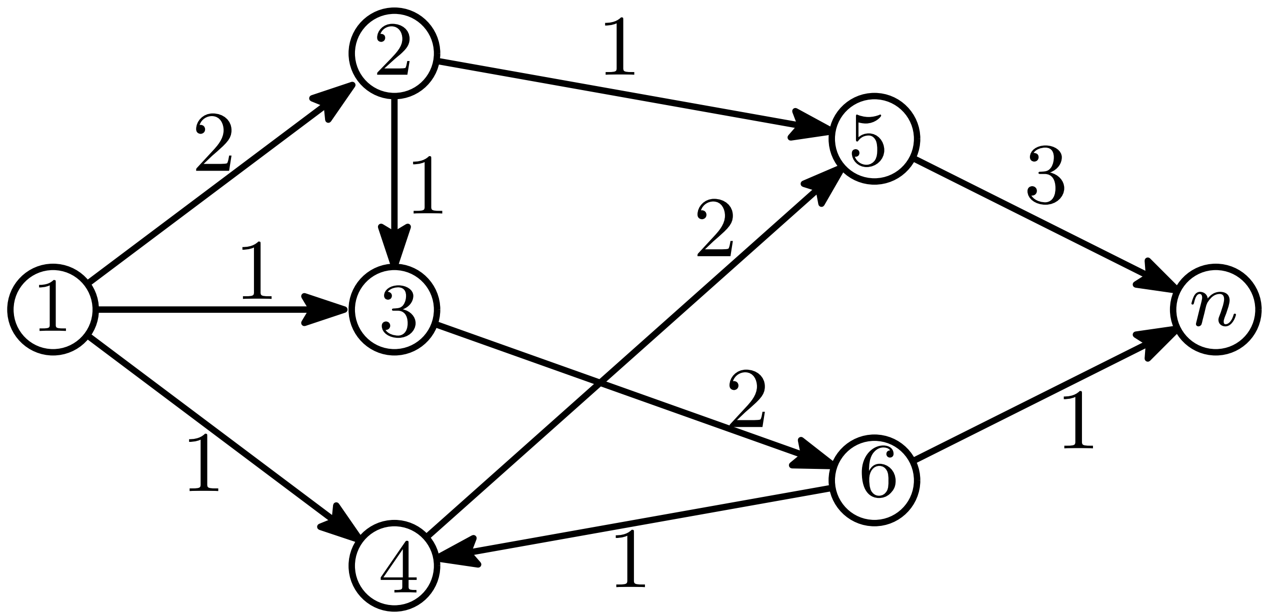
$x_{oc} = x_{cd} + x_{ce}$

$x_{ad} + x_{cd} = x_{dn}$

$x_{be} + x_{ce} = x_{en}$

first we formulate the LP more concisely:
let the vertices be numbered $1, \dots, n$, let c_{ij} the capacity and x_{ij} the flow on directed edge (i, j), and let f be the max flow.

Optimal solution: 4



well-known "max flow = min cut" $\longrightarrow$ now via LP-duality!

Recall: Flow in a Network

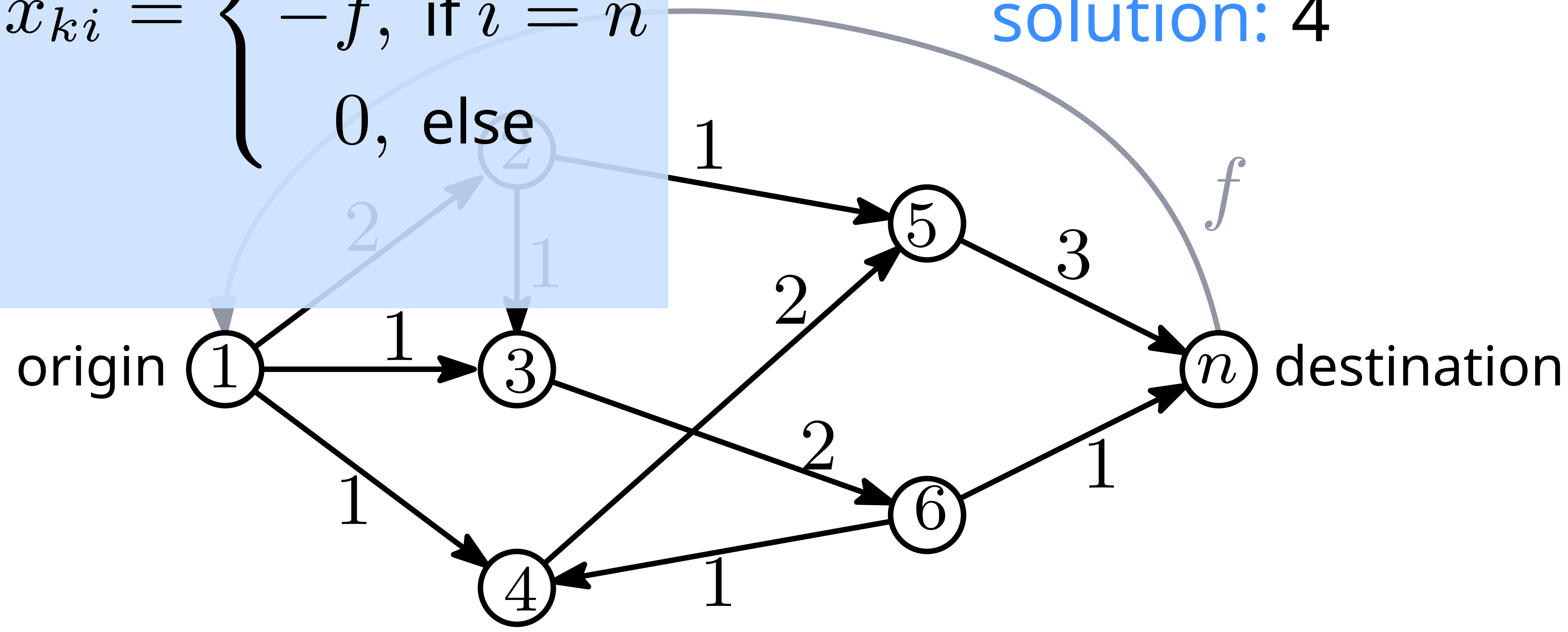
Linear Program Formulation

first we formulate the LP more concisely:
let the vertices be numbered $1, \dots, n$, let c_{ij} the capacity and x_{ij} the flow on directed edge (i, j), and let f be the max flow.

maximize f

subject to $$\sum_{(i,j)\in E} x_{ij} - \sum_{(k,i)\in E} x_{ki} = \begin{cases} f, & \text{if } i = 1 \\ -f, & \text{if } i = n \\ 0, & \text{else} \end{cases}$$

$$0 \le x_{ij} \le c_{ij}$$



Optimal solution: 4

Recall: Flow in a Network

Linear Program Formulation

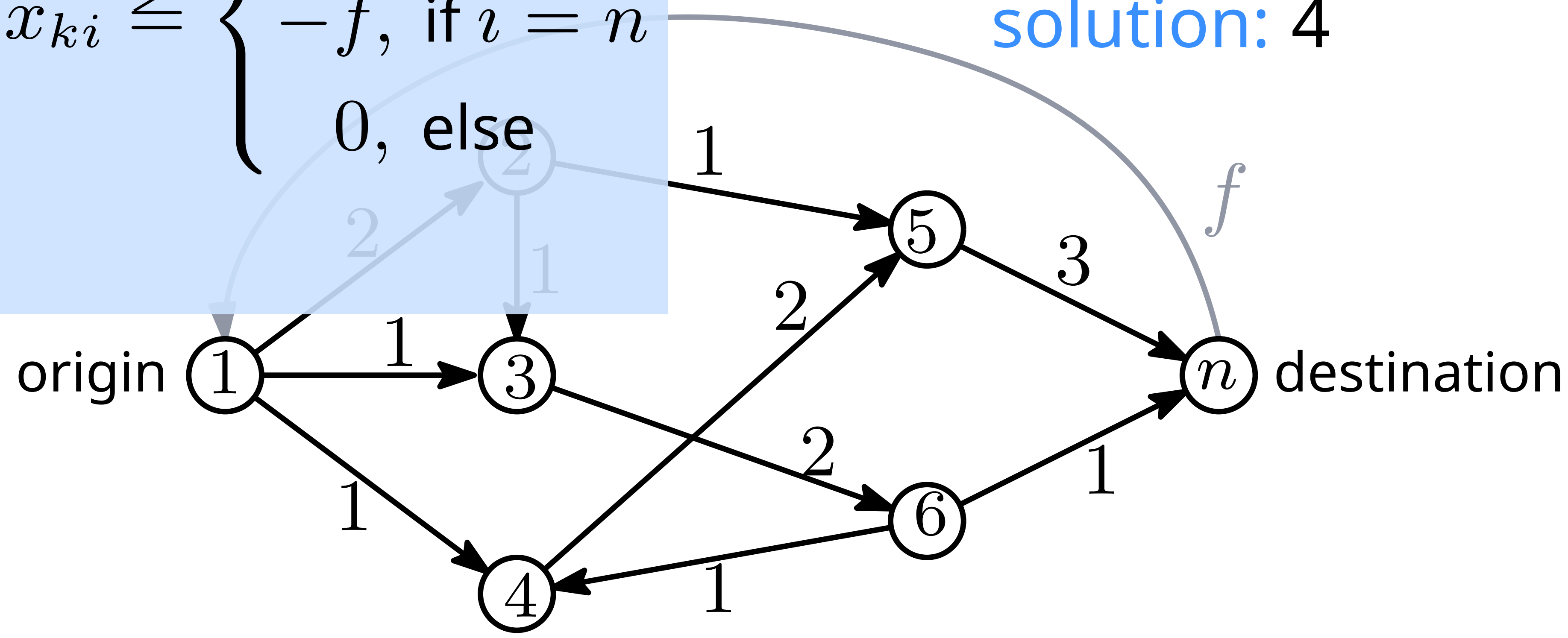
first we formulate the LP more concisely:
let the vertices be numbered $1, \ldots, n$, let c_{ij} the capacity and x_{ij} the flow on directed edge (i, j), and let f be the max flow.

maximize f

subject to $$\sum_{(i,j)\in E} x_{ij} - \sum_{(k,i)\in E} x_{ki} \leq \begin{cases} f, & \text{if } i = 1 \\ -f, & \text{if } i = n \\ 0, & \text{else} \end{cases}$$

$$0 \leq x_{ij} \leq c_{ij}$$

Optimal solution: 4



actually we can relax the constraint without changing the optimum

Recall: Flow in a Network

Linear Program Formulation

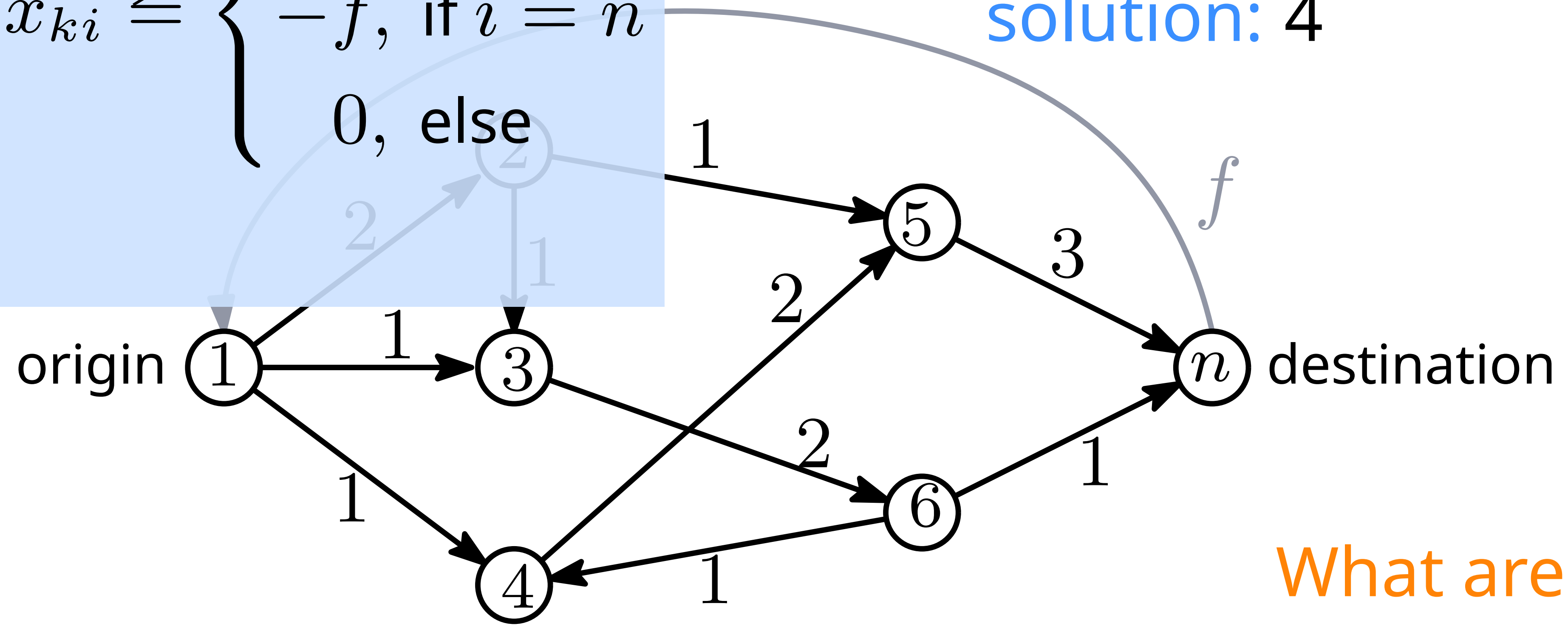
first we formulate the LP more concisely:
let the vertices be numbered $1, \dots, n$, let c_{ij} the capacity and x_{ij} the flow on directed edge (i, j), and let f be the max flow.

maximize f

subject to $$\sum_{(i,j)\in E} x_{ij} - \sum_{(k,i)\in E} x_{ki} \leq \begin{cases} f, & \text{if } i = 1 \\ -f, & \text{if } i = n \\ 0, & \text{else} \end{cases}$$

$$0 \leq x_{ij} \leq c_{ij}$$

Optimal solution: 4



What are A, x, b, c?

Let's write this in Matrix form $\max f$ subject to $Ax \leq b, x \geq 0$.

Recall: Flow in a Network

Linear Program Formulation

in Matrix form $\max f$ subject to $Ax \le b, x \ge 0$ where

$$x = \begin{bmatrix} f \\ x_{ij} \\ .. \\ .. \end{bmatrix} \quad c = \begin{bmatrix} 1 \\ 0 \\ .. \\ .. \end{bmatrix} \quad A = \left[\begin{array}{c|ccccc} -1 & .. & .. & .. & .. \\ 0 & .. & .. & .. & .. \\ .. & .. & .. & .. & .. \\ 1 & .. & .. & .. & .. \\ \hline 0 & 1 & 0 & .. & .. \\ .. & 0 & .. & .. & .. \\ .. & .. & .. & .. & 0 \\ 0 & .. & .. & 0 & 1 \end{array}\right] \quad b = \left[\begin{array}{c} 0 \\ .. \\ .. \\ 0 \\ \hline c_{ij} \\ .. \\ .. \\ .. \end{array}\right]$$

Recall: Flow in a Network

Linear Program Formulation

in Matrix form $\max f$ subject to $Ax \leq b, x \geq 0$ where

$$x = \begin{bmatrix} f \\ x_{ij} \\ .. \\ .. \end{bmatrix} \quad c = \begin{bmatrix} 1 \\ 0 \\ .. \\ .. \end{bmatrix} \quad A = \left[\begin{array}{c|ccccc} -1 & .. & .. & .. & .. \\ 0 & .. & .. & .. & .. \\ .. & .. & .. & .. & .. \\ 1 & .. & .. & .. & .. \\ \hline 0 & 1 & 0 & .. & .. \\ .. & 0 & .. & .. & .. \\ .. & .. & .. & .. & 0 \\ 0 & .. & .. & 0 & 1 \end{array}\right] \quad b = \left[\begin{array}{c} 0 \\ .. \\ .. \\ 0 \\ \hline c_{ij} \\ .. \\ .. \\ .. \end{array}\right]$$

every column contains exactly one -1 and one 1

Recall: Flow in a Network

Linear Program Formulation

in Matrix form $\max f$ subject to $Ax \le b, x \ge 0$ where

$$x = \begin{bmatrix} f \\ x_{ij} \\ .. \\ .. \end{bmatrix} \quad c = \begin{bmatrix} 1 \\ 0 \\ .. \\ .. \end{bmatrix} \quad A = \left[\begin{array}{c|cccc} -1 & .. & .. & .. & .. \\ 0 & .. & .. & .. & .. \\ .. & .. & .. & .. & .. \\ 1 & .. & .. & .. & .. \\ \hline 0 & 1 & 0 & .. & .. \\ .. & 0 & .. & .. & .. \\ .. & .. & .. & .. & 0 \\ 0 & .. & .. & 0 & 1 \end{array}\right] \quad b = \left[\begin{array}{c} 0 \\ .. \\ .. \\ 0 \\ \hline c_{ij} \\ .. \\ .. \\ .. \end{array}\right]$$

every column contains exactly one -1 and one 1

What is the dual?

Dual of the Max Flow LP

Linear Program Formulation

in Matrix form $\min \sum c_{ij} y_{ij}$ subject to $A^T y \geq c, x \geq 0$ where

every row contains exactly one -1 and one 1

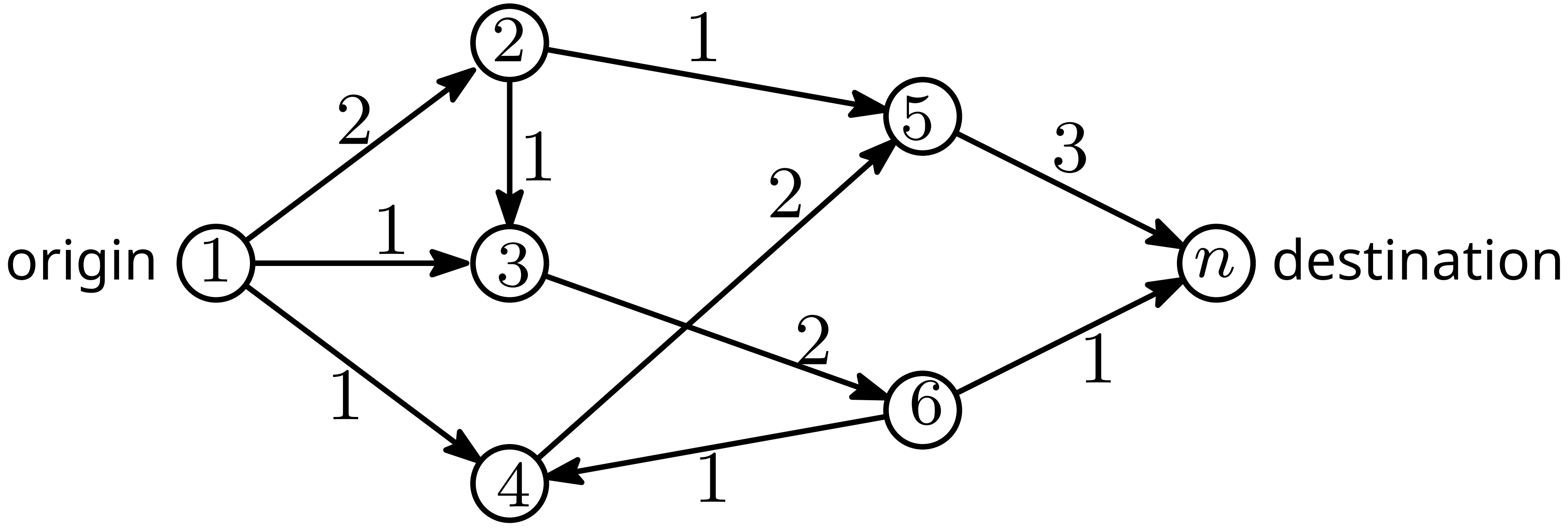
$$y = \begin{bmatrix} u_1 \\ .. \\ .. \\ u_n \\ y_{ij} \\ .. \\ .. \\ .. \end{bmatrix} \quad b = \begin{bmatrix} 0 \\ .. \\ .. \\ 0 \\ c_{ij} \\ .. \\ .. \\ .. \end{bmatrix} \quad A^T = \left[\begin{array}{cccc|cccc} -1 & 0 & .. & 1 & 0 & .. & .. & 0 \\ \hline .. & .. & .. & .. & 1 & 0 & .. & .. \\ .. & .. & .. & .. & 0 & .. & .. & .. \\ .. & .. & .. & .. & .. & .. & .. & .. \\ .. & .. & .. & .. & .. & .. & .. & .. \\ .. & .. & .. & .. & .. & .. & .. & .. \\ .. & .. & .. & .. & .. & .. & .. & 0 \\ .. & .. & .. & .. & .. & .. & 0 & 1 \end{array}\right] \quad c = \begin{bmatrix} 1 \\ 0 \\ .. \\ .. \end{bmatrix}$$

Dual of the Max Flow LP

Linear Program Formulation

in constraint form

$$\begin{aligned} \text{minimize} \quad & \sum c_{ij} y_{ij} \\ \text{subject to} \quad & -u_1 + u_n \geq 1 \\ & u_i - u_j + y_{ij} \geq 0 \\ & u_i \geq 0 \\ & y_{ij} \geq 0 \end{aligned}$$

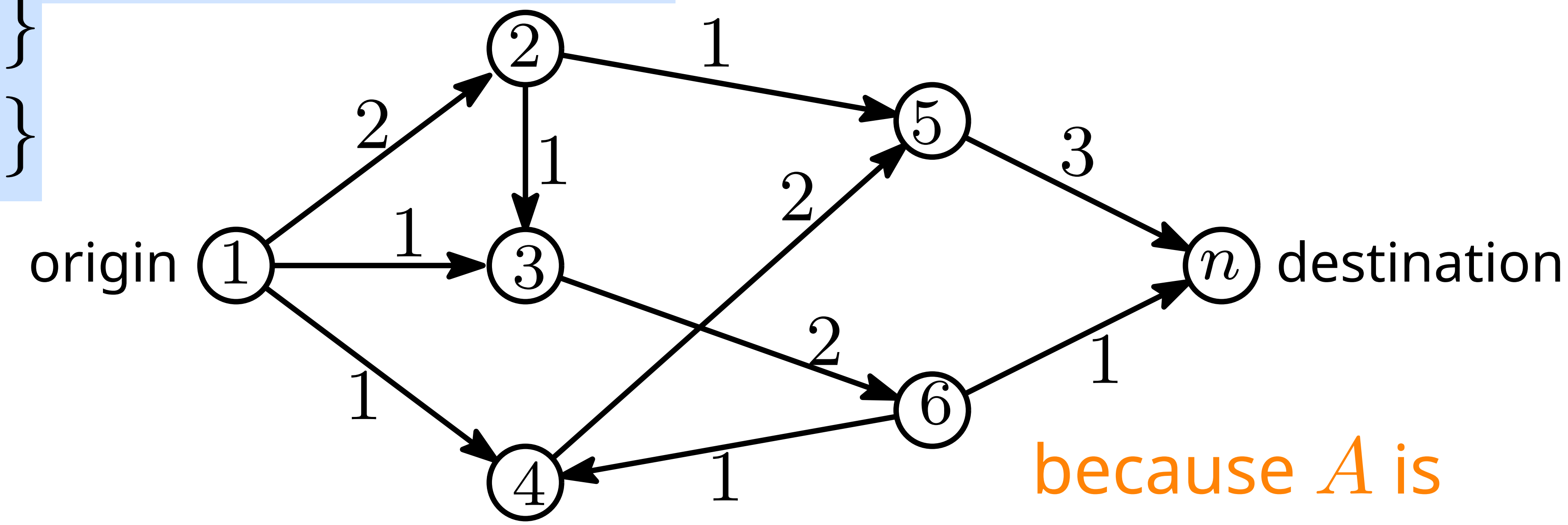


Dual of the Max Flow LP

Linear Program Formulation

in constraint form

$$
\begin{aligned}
\text{minimize} \quad & \sum c_{ij} y_{ij} \\
\text{subject to} \quad & -u_1 + u_n \geq 1 \\
& u_i - u_j + y_{ij} \geq 0 \\
& u_i \in \{0, 1\} \\
& y_{ij} \in \{0, 1\}
\end{aligned}
$$



actually, we can restrict all variables to be integer, even 0-1

because A is total unimodular!

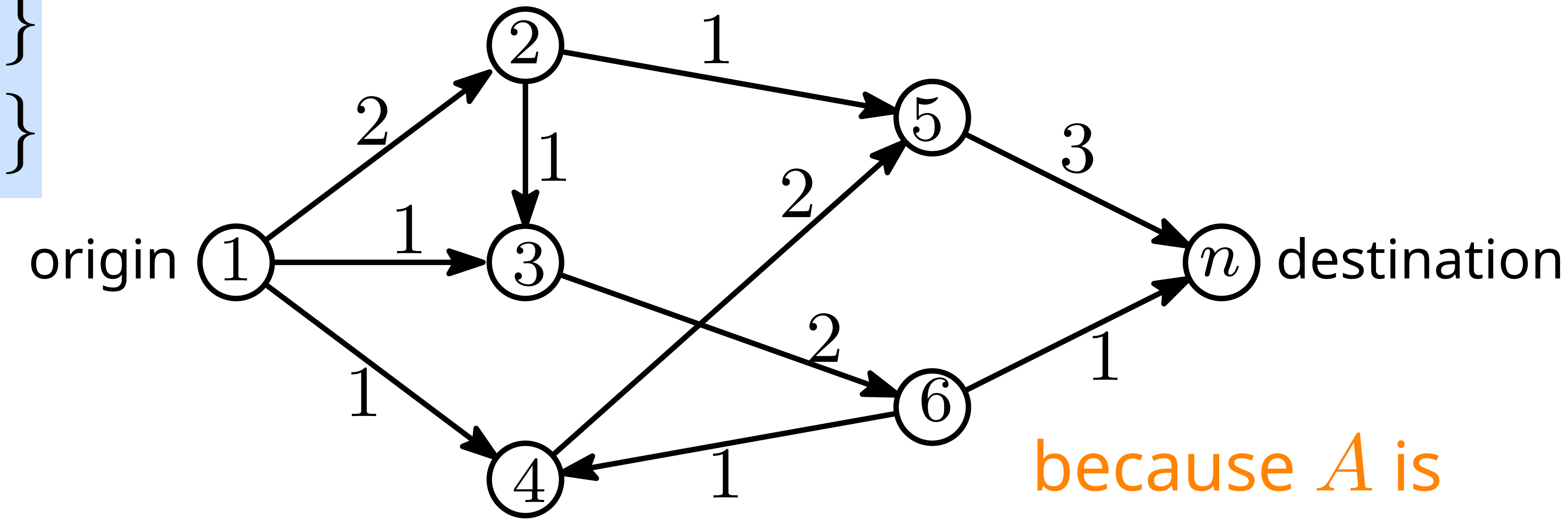
Dual of the Max Flow LP

Linear Program Formulation

in constraint form

$$
\begin{aligned}
\text{minimize} \quad & \sum c_{ij} y_{ij} \\
\text{subject to} \quad & -u_1 + u_n \geq 1 \\
& u_i - u_j + y_{ij} \geq 0 \\
& u_i \in \{0,1\} \\
& y_{ij} \in \{0,1\}
\end{aligned}
$$

$\rightarrow$ Min Cut



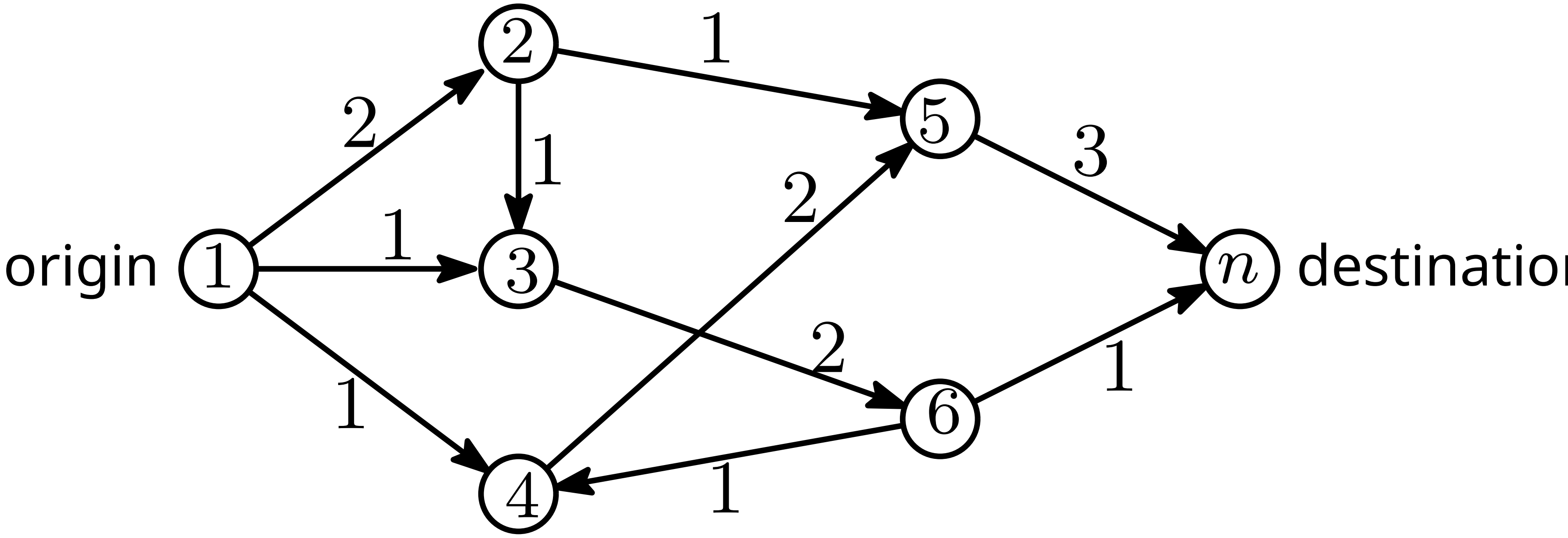
actually, we can restrict all variables to be integer, even 0-1

because A is total unimodular!

Flows and Cuts in a Network

Alternative LP

Let P be the set of all paths $q \rightsquigarrow s$
use variables x_p, for all $p \in P$



$$\max \sum_{p \in P} x_p$$

subject to

$$\sum_{p \ni e} x_p \le c(e) \quad \forall e \in E$$

$$0 \le x_p \quad \forall p \in P$$

primal

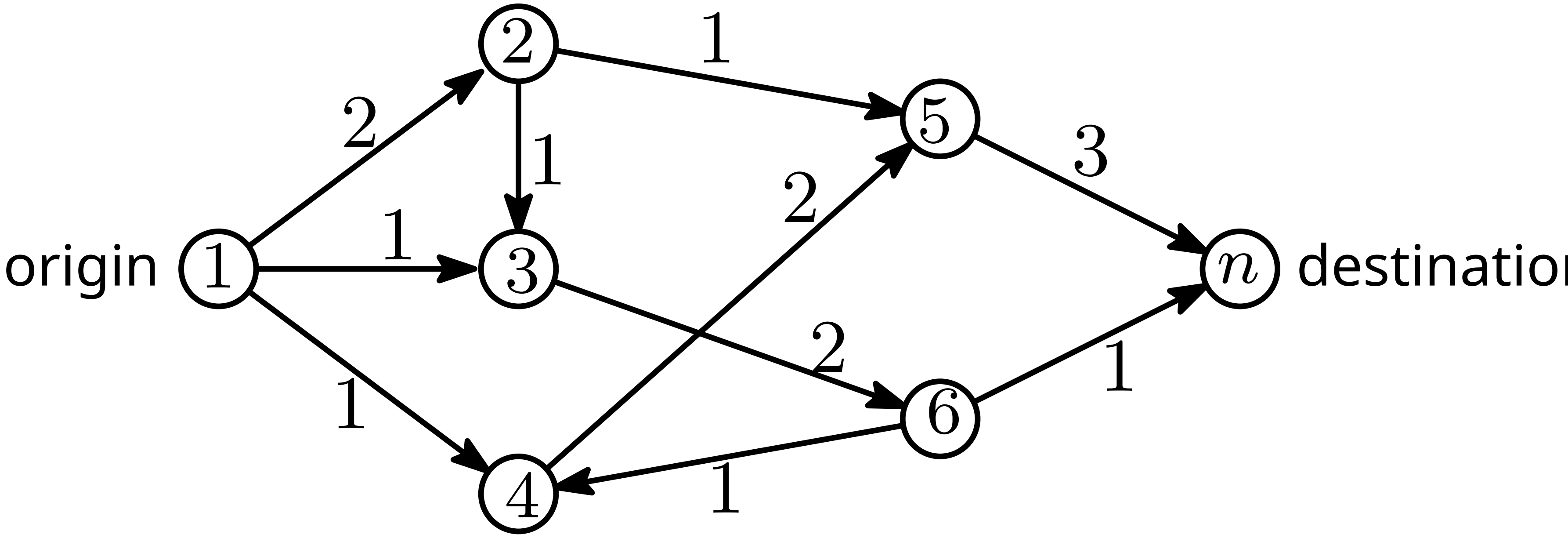
Max Flow

Flows and Cuts in a Network

Alternative LP

Let P be the set of all paths $q \rightsquigarrow s$

use variables x_p, for all $p \in P$



$\max \sum_{p \in P} x_p$

subject to

$\sum_{p \ni e} x_p \leq c(e) \quad \forall e \in E$

$0 \leq x_p \quad \forall p \in P$

primal

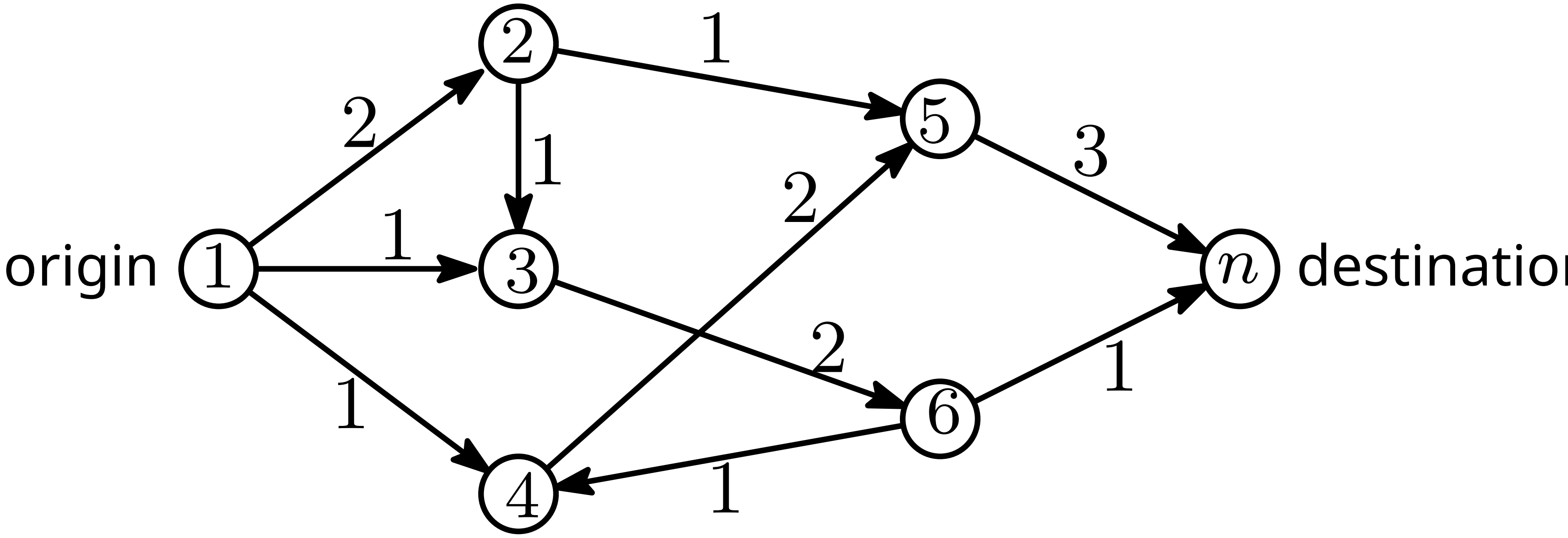
Max Flow

What is the dual?

Flows and Cuts in a Network

Alternative LP

Let P be the set of all paths $q \rightsquigarrow s$
use variables x_p, for all $p \in P$



$$\max \sum_{p \in P} x_p$$

subject to

$$\sum_{p \ni e} x_p \leq c(e) \quad \forall e \in E$$

$$0 \leq x_p \quad \forall p \in P$$

primal

Max Flow

$$\min \sum_{e \in E} c(e) y_e$$

subject to

$$\sum_{e \in p} y_e \geq 1 \quad \forall p \in P$$

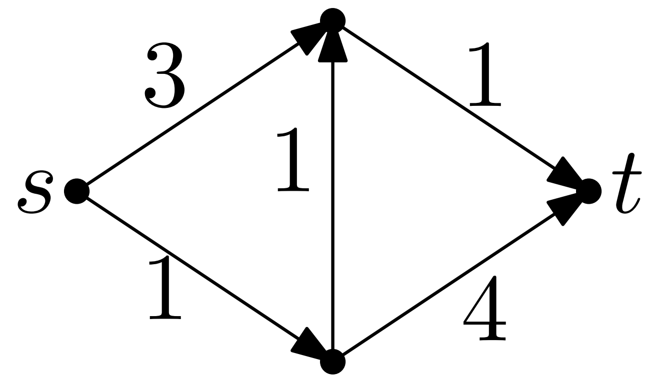
$$y_e \geq 0 \quad \forall e \in E$$

dual

Min Cut

Shortest Path Problem

Given a directed graph $G = (V, E)$ with edge weights w, we are looking for a shortest path from s to t.



Can we also model this as ILP? Yes! How?

Idea 1: use variable x_{uv} for whether edge (u, v) is used.

minimize $\displaystyle\sum_{(u,v)\in E} w(u,v)x_{uv}$

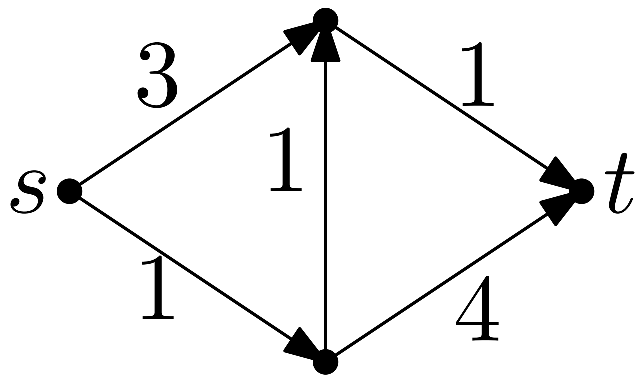
subject to $\displaystyle\sum_{(u,v)\in E} x_{uv} = \sum_{(v,w)\in E} x_{vw}$ for each vertex $v \in V \setminus \{s, t\}$, and

$$\sum_{(u,t)\in E} x_{ut} = 1.$$

$x_{uv} \in \{0, 1\}$ for each edge $(u, v) \in E$.

Shortest Path Problem

Given a directed graph $G = (V, E)$ with edge weights w, we are looking for a shortest path from s to t.



Can we also model this as ILP? Yes! How?

Idea 1: use variable x_{uv} for whether edge (u, v) is used.

minimize $\displaystyle\sum_{(u,v)\in E} w(u,v)x_{uv}$

subject to $\displaystyle\sum_{(u,v)\in E} x_{uv} = \sum_{(v,w)\in E} x_{vw}$ for each vertex $v \in V \setminus \{s,t\}$, and

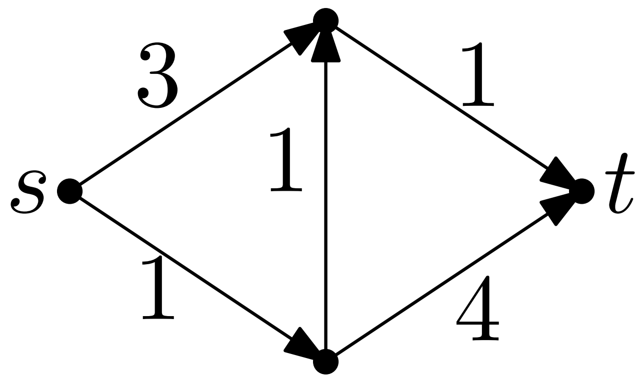
$$\sum_{(u,t)\in E} x_{ut} = 1.$$

$x_{uv} \in \{0,1\}$ for each edge $(u,v) \in E$.

LP-relaxation has $\{0,1\}$-solution, like MinWeight Perfect Matching

Shortest Path Problem

Given a directed graph $G = (V, E)$ with edge weights w, we are looking for a shortest path from s to t.



Can we also model this as ILP? Yes! How?

Idea 1: use variable x_{uv} for whether edge (u, v) is used.

minimize $\displaystyle\sum_{(u,v)\in E} w(u,v)x_{uv}$

subject to $\displaystyle\sum_{(u,v)\in E} x_{uv} = \sum_{(v,w)\in E} x_{vw}$ for each vertex $v \in V \setminus \{s,t\}$, and

$\displaystyle\sum_{(u,t)\in E} x_{ut} = 1.$

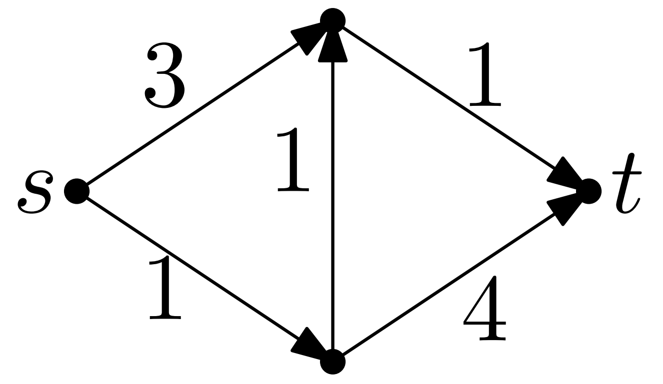
$x_{uv} \in \{0,1\}$ for each edge $(u,v) \in E$.

LP-relaxation has $\{0,1\}$-solution, like MinWeight Perfect Matching

Why?

Shortest Path Problem

Given a directed graph $G = (V, E)$ with edge weights w, we are looking for a shortest path from s to t.



Can we also model this as ILP? Yes! How?

Idea 1: use variable x_{uv} for whether edge (u, v) is used.

minimize $\displaystyle\sum_{(u,v)\in E} w(u,v)x_{uv}$

subject to $\displaystyle\sum_{(u,v)\in E} x_{uv} = \sum_{(v,w)\in E} x_{vw}$ for each vertex $v \in V \setminus \{s,t\}$, and

$\displaystyle\sum_{(u,t)\in E} x_{ut} = 1.$

$x_{uv} \in \{0,1\}$ for each edge $(u,v) \in E$.

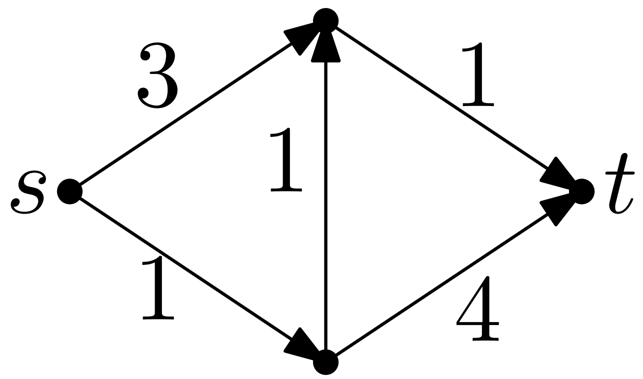
LP-relaxation has $\{0,1\}$-solution, like MinWeight Perfect Matching

Why?

A is total unimodular

Shortest Path Problem

Given a directed graph $G = (V, E)$ with edge weights w, we are looking for a shortest path from s to t.

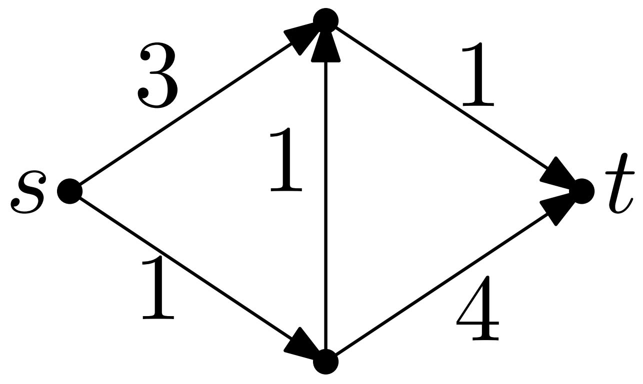


Can we also model this as ILP? Yes! How?

Idea 2: use variable d_v for distance from s to v.
Then we want that $d_v \leq d_u + w(u, v)$ for all edges (u, v).

Shortest Path Problem

Given a directed graph $G = (V, E)$ with edge weights w, we are looking for a shortest path from s to t.



Can we also model this as ILP? Yes! How?

Idea 2: use variable d_v for distance from s to v.
Then we want that $d_v \leq d_u + w(u, v)$ for all edges (u, v).

maximize d_t

subject to

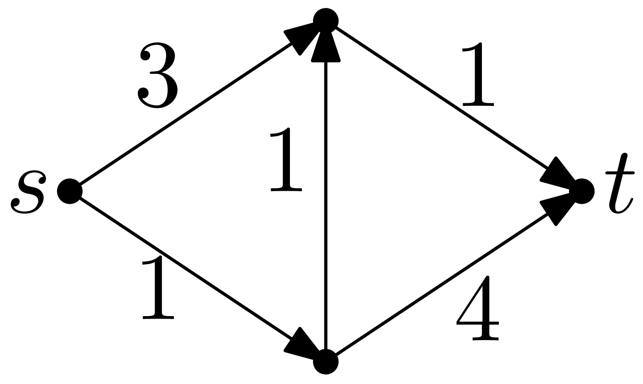
$d_v - d_u \leq w(u, v) \ \ \forall (u, v) \in E$

$d_s = 0$

Actually an LP!

Shortest Path Problem

Given a directed graph $G = (V, E)$ with edge weights w, we are looking for a shortest path from s to t.



Can we also model this as ILP? Yes! How?

Idea 2: use variable d_v for distance from s to v.
Then we want that $d_v \leq d_u + w(u, v)$ for all edges (u, v).

maximize d_t

subject to

$d_v - d_u \leq w(u, v) \;\; \forall (u, v) \in E$

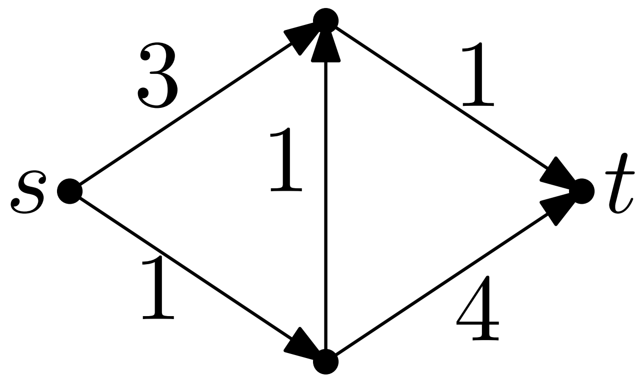
$d_s = 0$

Why do we maximize?

Actually an LP!

Shortest Path Problem

Given a directed graph $G = (V, E)$ with edge weights w, we are looking for a shortest path from s to t.



Can we also model this as ILP? Yes! How?

Idea 2: use variable d_v for distance from s to v.
Then we want that $d_v \leq d_u + w(u, v)$ for all edges (u, v).

maximize d_t

subject to

$d_v - d_u \leq w(u, v) \;\; \forall (u, v) \in E$

$d_s = 0$

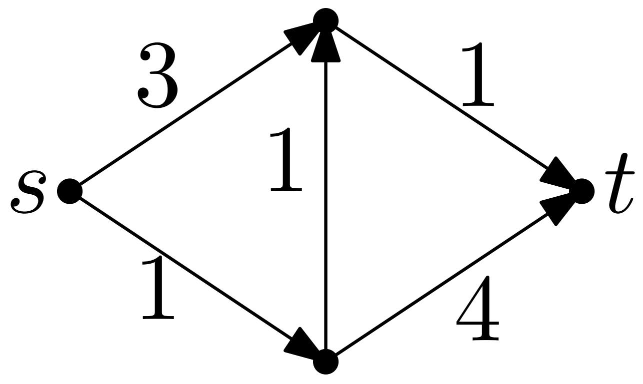
Why do we maximize?

otherwise we could set all d_v to zero

Actually an LP!

Shortest Path Problem

Given a directed graph $G = (V, E)$ with edge weights w, we are looking for a shortest path from s to t.



Can we also model this as ILP? Yes! How?

Idea 2: use variable d_v for distance from s to v.
Then we want that $d_v \leq d_u + w(u, v)$ for all edges (u, v).

maximize d_t

subject to

$d_v - d_u \leq w(u, v) \;\; \forall (u, v) \in E$

$d_s = 0$

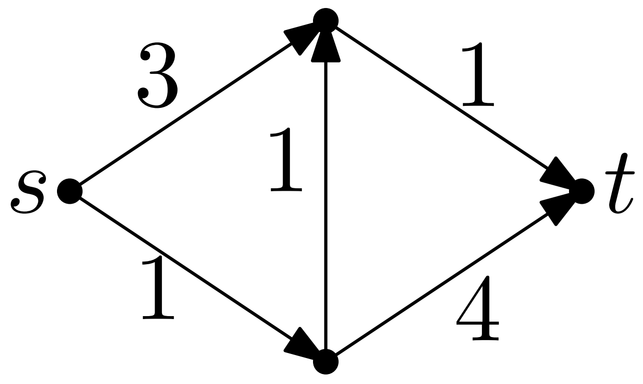
Why do we maximize?

otherwise we could set all d_v to zero

Is there a connection between the two LPs?

Shortest Path Problem

Given a directed graph $G = (V, E)$ with edge weights w, we are looking for a shortest path from s to t.



Can we also model this as ILP? Yes! How?

Idea 2: use variable d_v for distance from s to v.
Then we want that $d_v \leq d_u + w(u, v)$ for all edges (u, v).

maximize d_t

subject to

$d_v - d_u \leq w(u, v) \ \forall (u, v) \in E$

$d_s = 0$

Why do we maximize?

otherwise we could set all d_v to zero

Is there a connection between the two LPs? They are dual to each other!