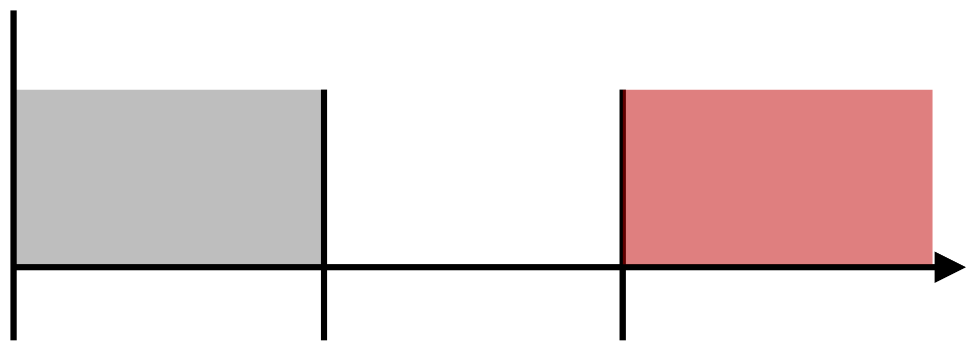
Lecture 17

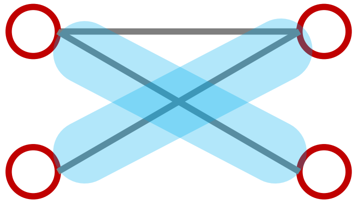
Duality



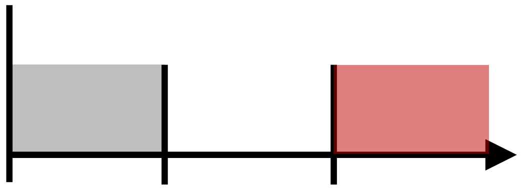
Outline

1. Maximum flow

2. Bipartite perfect matching



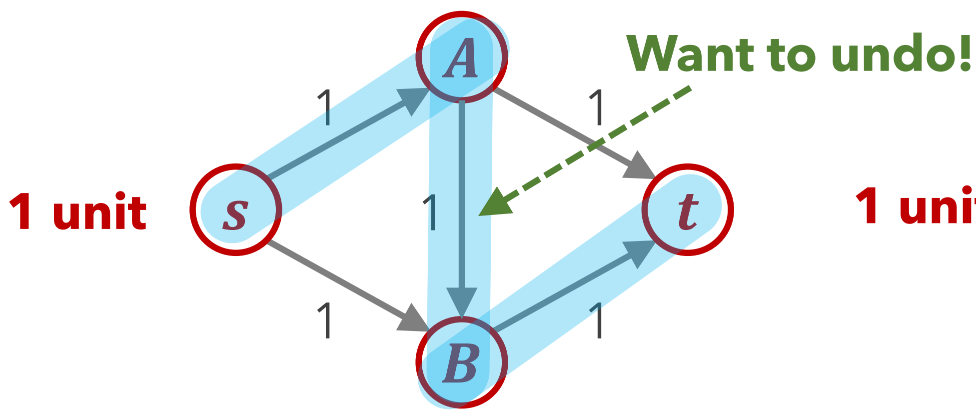
3. Linear programming duality

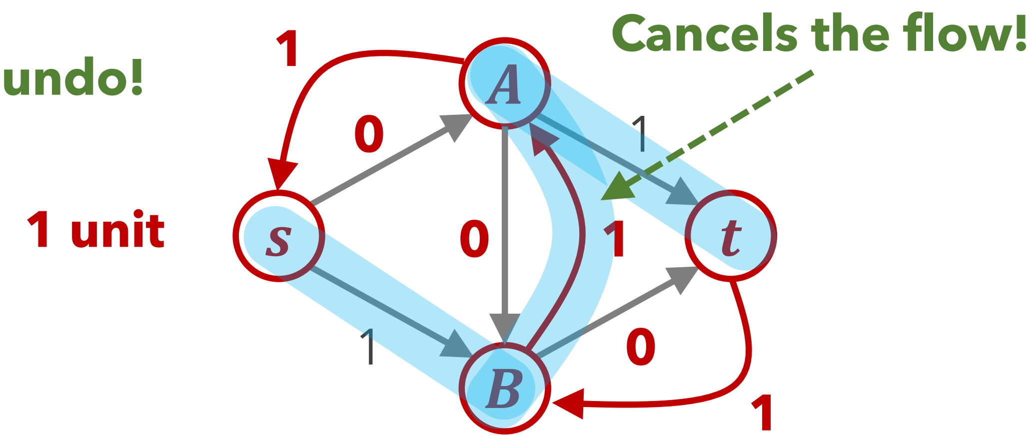


Maximum flow

Input: Directed graph $G = (V, E)$, source $s \in V$, sink $t \in V$, capacities $c_e \in \mathbb{Z}^+$

Goal: Route the maximum amount of water from s to t

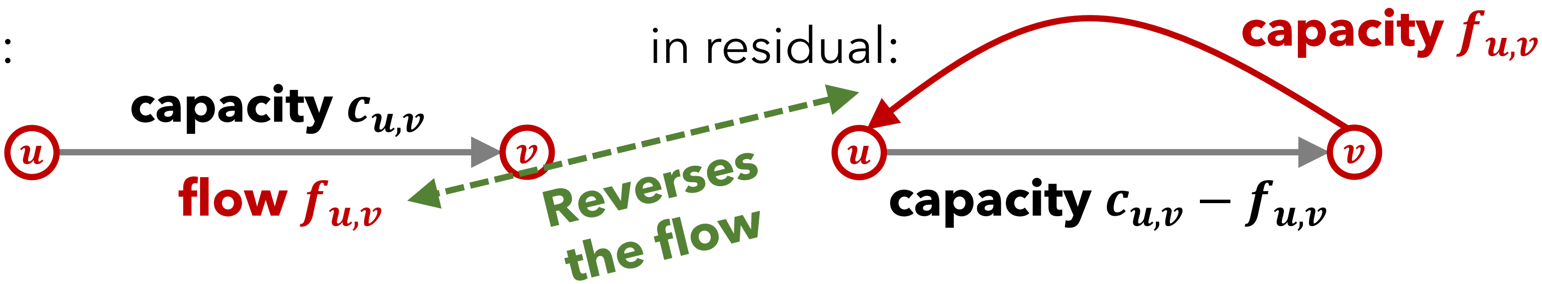




Def: Given a flow f on G, the residual graph G_f is as follows. For all edges (u, v):

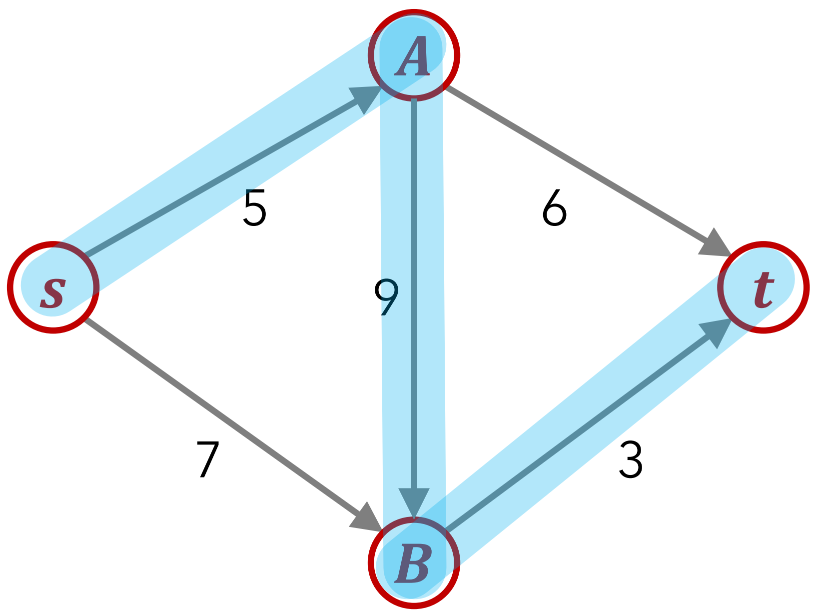
in original:

in residual:



Ford-Fulkerson algorithm

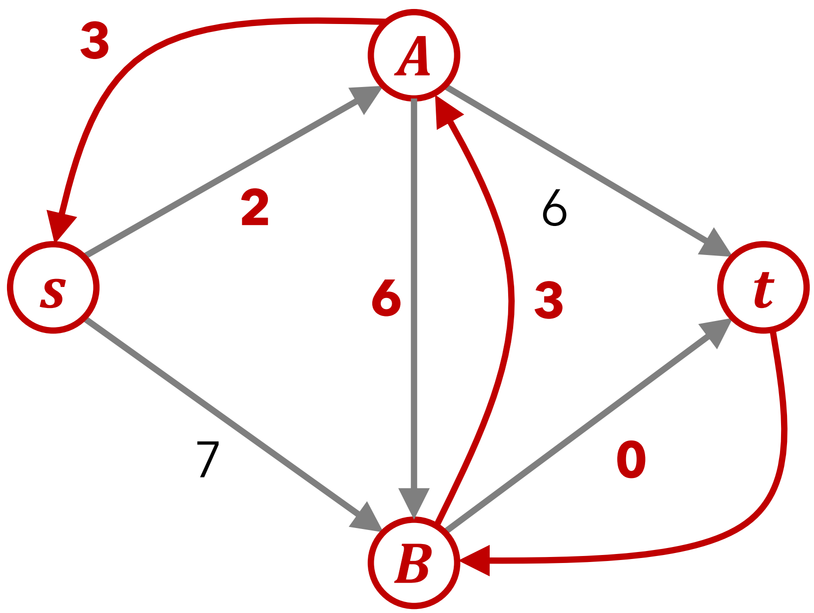
1. Find a path $\boldsymbol{P}$ from $\boldsymbol{s}$ to $\boldsymbol{t}$ **in the residual graph** which is not yet saturated
2. Send more flow along $\boldsymbol{P}$ = an **augmenting path**
3. Repeat



$s \to A \to B \to t$: **3 units**

Ford-Fulkerson algorithm

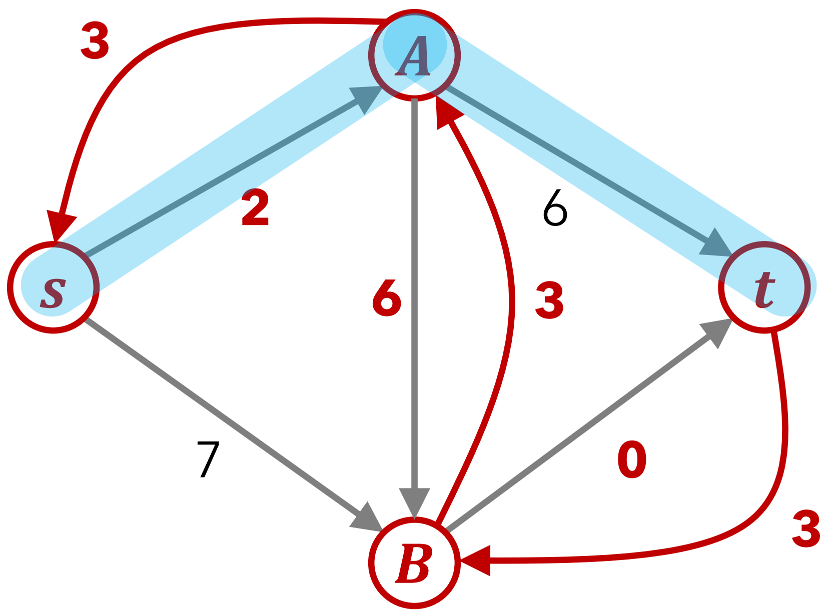
1. Find a path $\boldsymbol{P}$ from $\boldsymbol{s}$ to $\boldsymbol{t}$ **in the residual graph** which is not yet saturated
2. Send more flow along $\boldsymbol{P}$ = an **augmenting path**
3. Repeat



$s \to A \to B \to t$: **3 units**

Ford-Fulkerson algorithm

1. Find a path P from s to t **in the residual graph** which is not yet saturated
2. Send more flow along P = an **augmenting path**
3. Repeat

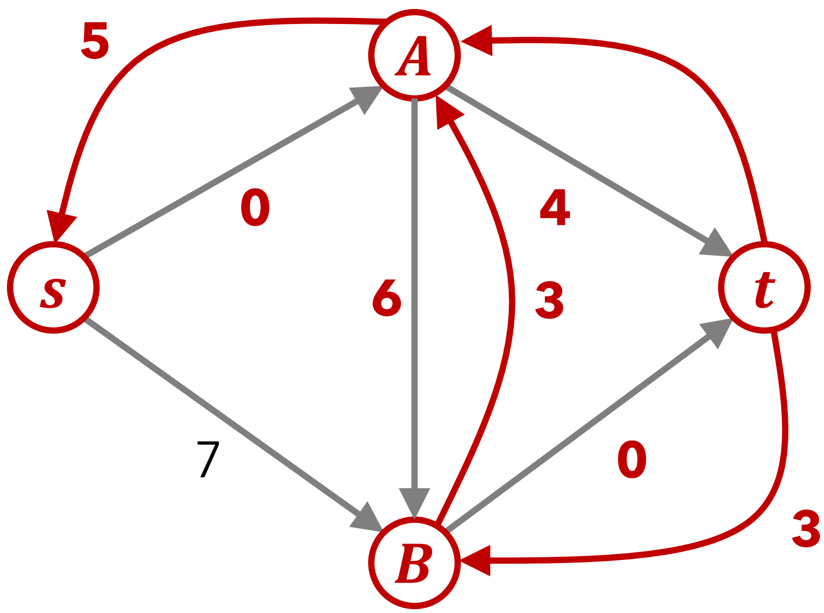


$s \rightarrow A \rightarrow B \rightarrow t$: **3 units**

$s \rightarrow A \rightarrow t$: **2 units**

Ford-Fulkerson algorithm

1. Find a path P from s to t **in the residual graph** which is not yet saturated
2. Send more flow along P = an **augmenting path**
3. Repeat

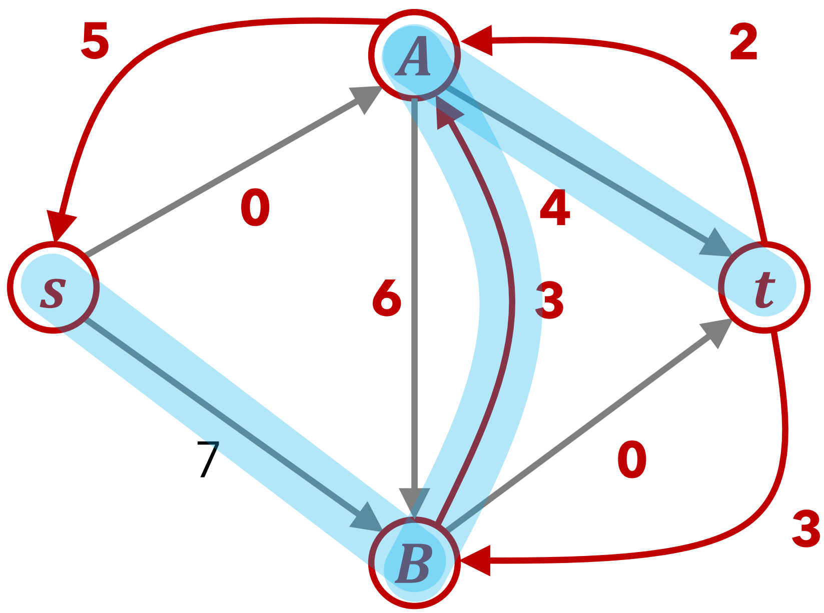


$s \to A \to B \to t$: **3 units**

$s \to A \to t$: **2 units**

Ford-Fulkerson algorithm

1. Find a path $\boldsymbol{P}$ from $\boldsymbol{s}$ to $\boldsymbol{t}$ **in the residual graph** which is not yet saturated
2. Send more flow along $\boldsymbol{P}$ = an **augmenting path**
3. Repeat



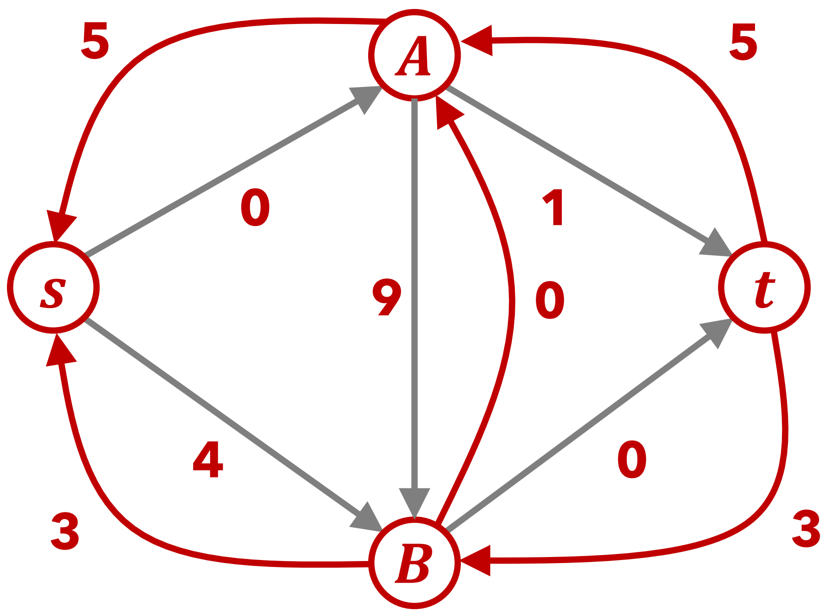
$s \to A \to B \to t$: **3 units**

$s \to A \to t$: **2 units**

$s \to B \to A \to t$: **3 units**

Ford-Fulkerson algorithm

1. Find a path ***P*** from ***s*** to ***t*** **in the residual graph** which is not yet saturated
2. Send more flow along ***P*** = an **augmenting path**
3. Repeat



$s \to A \to B \to t$: **3 units**

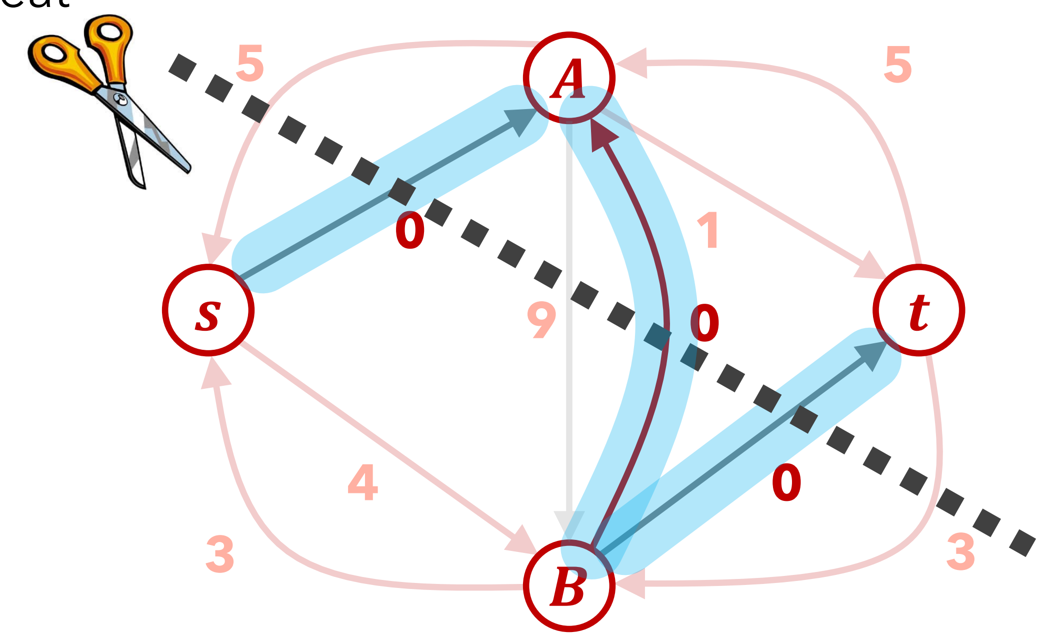
$s \to A \to t$: **2 units**

$s \to B \to A \to t$: **3 units**

total: **8 units**

Ford-Fulkerson algorithm

1. Find a path P from s to t **in the residual graph** which is not yet saturated
2. Send more flow along P = an **augmenting path**
3. Repeat



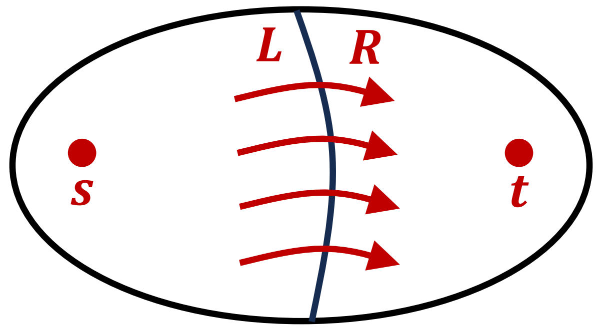
$s \to A \to B \to t$: **3 units**

$s \to A \to t$: **2 units**

$s \to B \to A \to t$: **3 units**

total: **8 units**

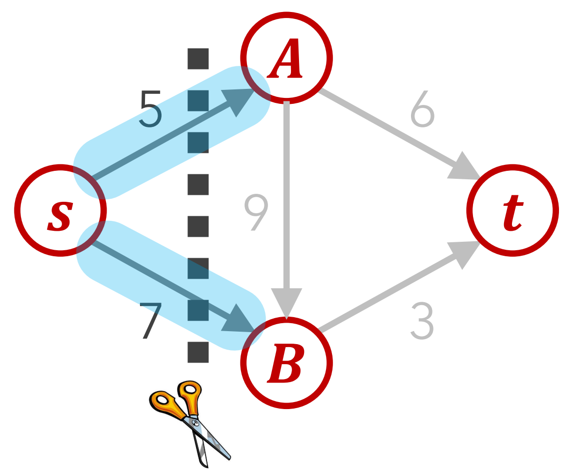
Def: An ***s*-*t* cut** is a partition $V = L \cup R$ of the vertices such that $s \in L$ and $t \in R$



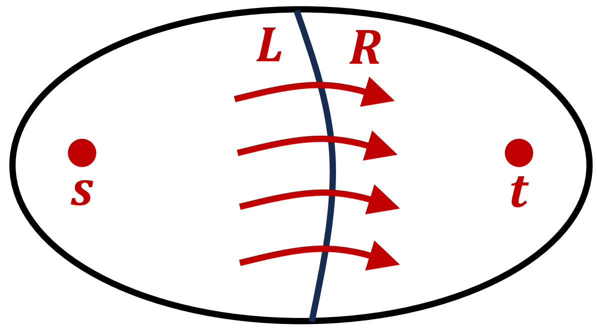
Def: The **capacity** of the **cut** is $\mathrm{capacity}(L,R) = \sum_{\substack{u \to v \\ u \in L,\, v \in R}} c_{u,v}$

Thm: For any flow f and any cut (L,R),

$$\mathrm{size}(f) \leq \mathrm{capacity}(L,R).$$



Def: An s-t **cut** is a partition $V = L \cup R$ of the vertices such that $s \in L$ and $t \in R$



Def: The **capacity** of the **cut** is $\mathbf{capacity}(L, R) = \sum_{\substack{u \to v \\ u \in L,\, v \in R}} c_{u,v}$

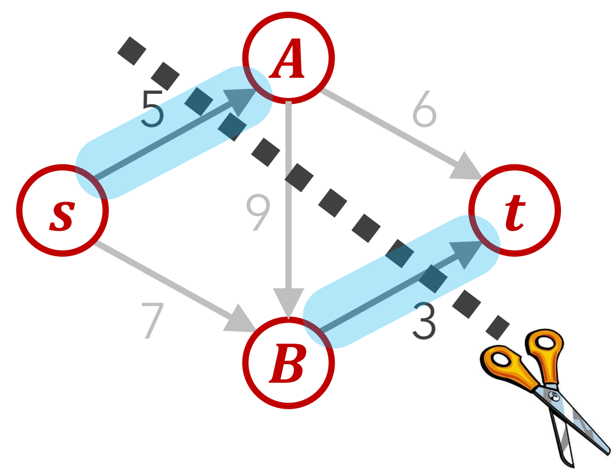
Thm: For any flow f and any cut (L, R),

$$\mathbf{size}(f) \leq \mathbf{capacity}(L, R).$$

Def: The **Min-cut** is the cut with **minimum** capacity.

Aka: Max-flow ≤ Min-cut

(Harris and Ross' "bottleneck")



Thm: Max-flow = Min-cut

Pf: Only need to show "≥"

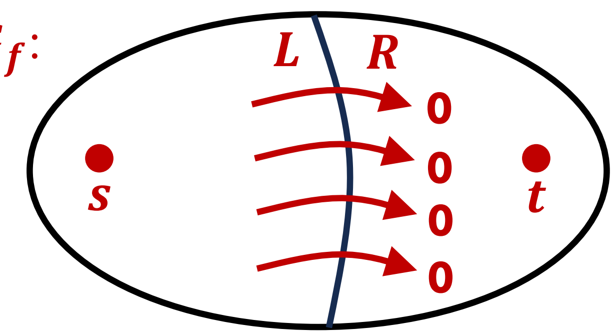
Run Ford-Fulkerson on G. Let f be the flow it outputs.

Then no $s \to t$ in residual graph G_f.

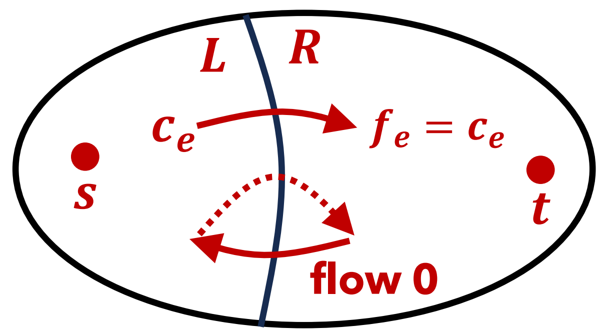
Set L = vertices reachable from s in G_f.

R = everything else.

in G_f:



in G:



Max-flow ≥ size(f) = capacity(L, R) ≥ Min-cut. □

Thm: Ford-Fulkerson outputs a **maximum flow**.

Runtime $\approx$ # of augmenting paths $\leq \boldsymbol{U}$, where $\boldsymbol{U}$ = Max-flow

(× the time to find the paths) (in graphs of integer weights)

$\leq \mathbf{O}(m+n) \cdot \boldsymbol{U}$

Is this a good runtime?

Suppose each capacity $\boldsymbol{c_e}$ was $\leq \boldsymbol{C}$.

Then $\boldsymbol{U} \leq \boldsymbol{m} \cdot \boldsymbol{C}$.

Each $\boldsymbol{c_e}$ is a $\mathbf{log_2}(\boldsymbol{C})$–bit integer.

So $\boldsymbol{U}$ can be **exponential** in the input length!

This is a **pseudo-polynomial** algorithm

(it is polynomial in the **numerical value** of the input)

Recall: Knapsack

Runtime $\approx$ # of augmenting paths $\leq \boldsymbol{U}$, where $\boldsymbol{U}$ = Max-flow

(× the time to find the paths) (in graphs of integer weights)

$\leq \mathbf{O}(m+n) \cdot \boldsymbol{U}$

Surprise: If all the capacities are integral, then the Max-flow is integral.

(all the capacities/flows are integers)

Other algorithms:

Dinitz 1970/Edmonds-Karp 1972: Always pick the shortest augmenting path

Runs in time $\mathbf{O}(n\, m^2)$!

⋮

(many, many more)

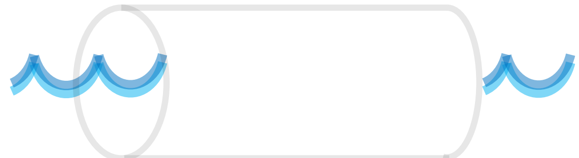
⋮

Chen-Kyng-Liu-Peng-Gutenberg-Sachdeva 2022: $\mathbf{O}(m^{1+o(1)} \cdot \log(\boldsymbol{U}))$

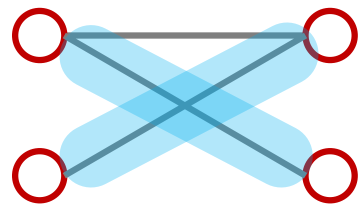
(only 112 pages!)

Outline

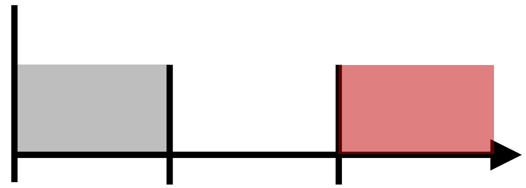
1. Maximum flow



2. Bipartite perfect matching



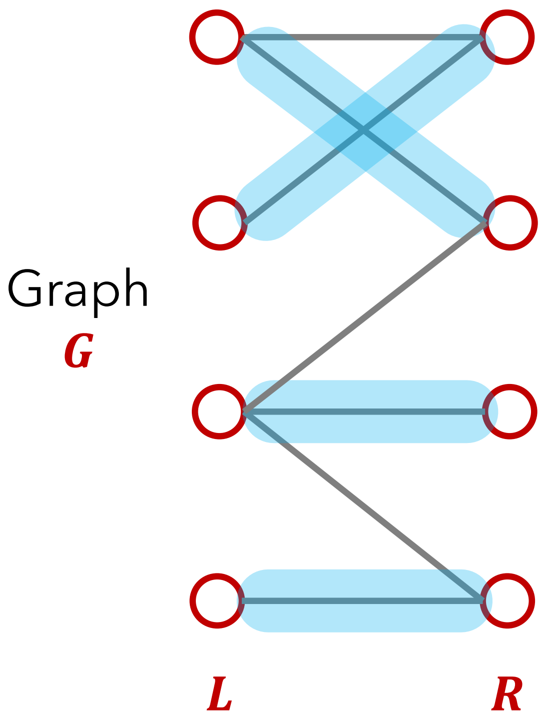
3. Linear programming duality



Bipartite Perfect Matching

Input: Bipartite (undirected) graph $G = (L, R, E)$ with $|L| = |R| = n$

Output: A perfect matching from L to R



Example:

L = UC Berkeley courses

R = UC Berkeley classrooms

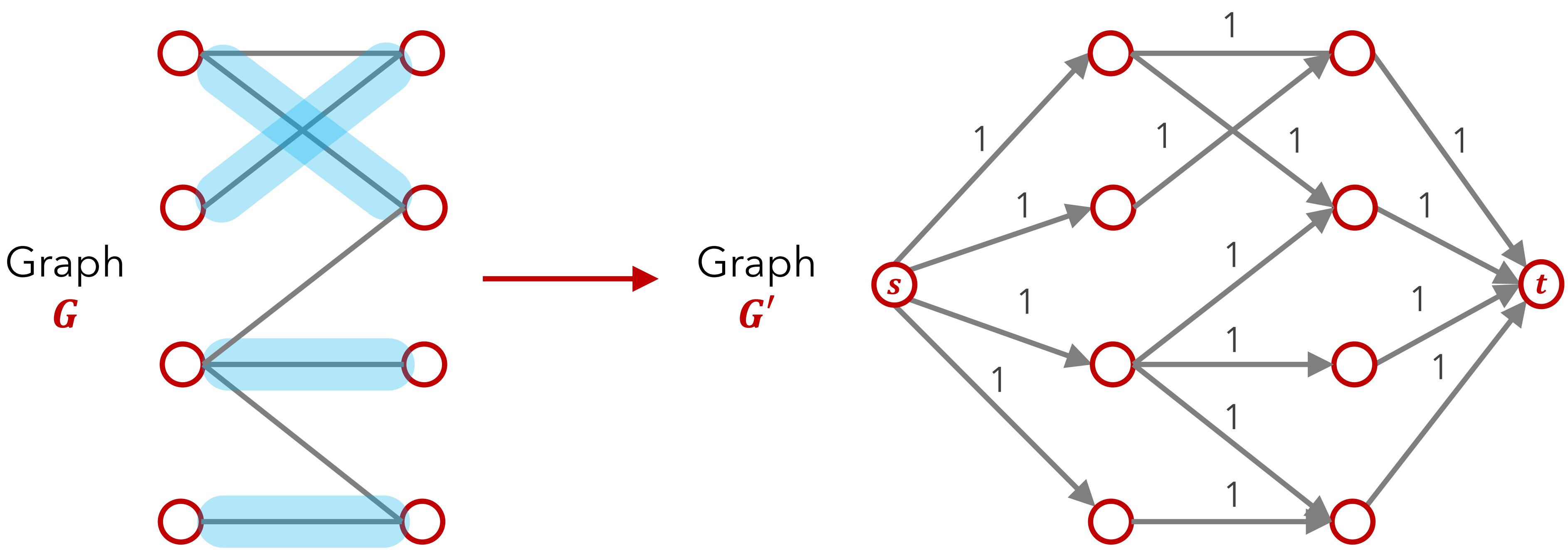
E = each course is connected to the classrooms it can be taught in

Q: Can we assign every course to a room?

Bipartite Perfect Matching

Input: Bipartite (undirected) graph $G = (L, R, E)$ with $|L| = |R| = n$

Output: A perfect matching from L to R



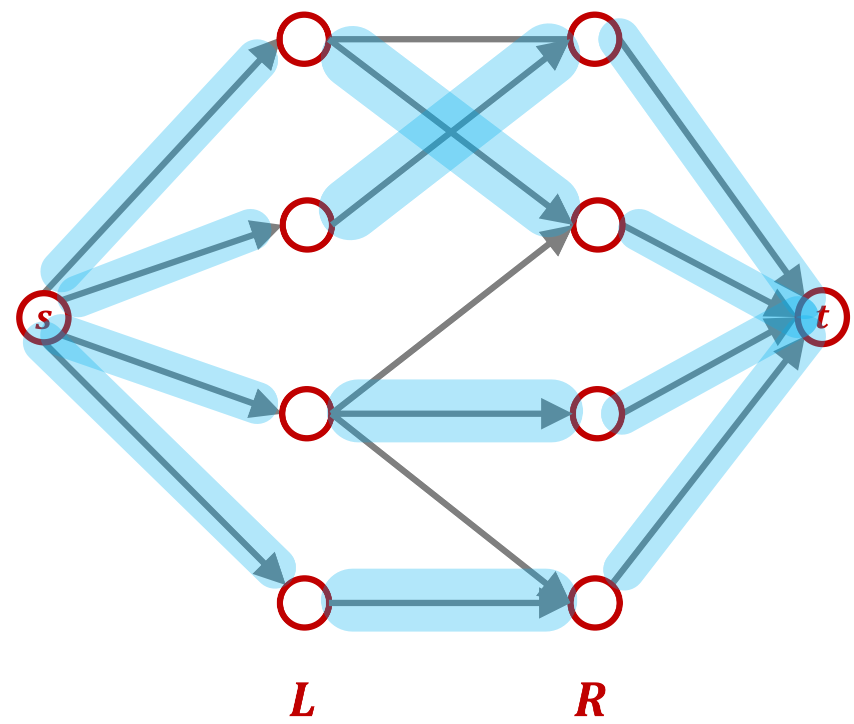
Thm: G has a perfect matching $\Leftrightarrow$ Max-flow$(G') = n$

Thm: $\boldsymbol{G}$ has a perfect matching
$\Leftrightarrow$ Max-flow($\boldsymbol{G'}$) = $\boldsymbol{n}$

Pf:

Case 1: ($\Rightarrow$) ☑

1. Let $\boldsymbol{M}$ be a perfect matching in $\boldsymbol{G}$.
2. Put **1** unit of flow on every edge in **M**
   and every $\boldsymbol{s} \to v$ edge
   and every $v \to \boldsymbol{t}$ edge.
3. Then this is a flow of size $\boldsymbol{n}$.



Graph $\boldsymbol{G'}$

Thm: G has a perfect matching
$\Leftrightarrow$ Max-flow$(G') = n$

a "**reduction** from perfect matching to maximum flow"

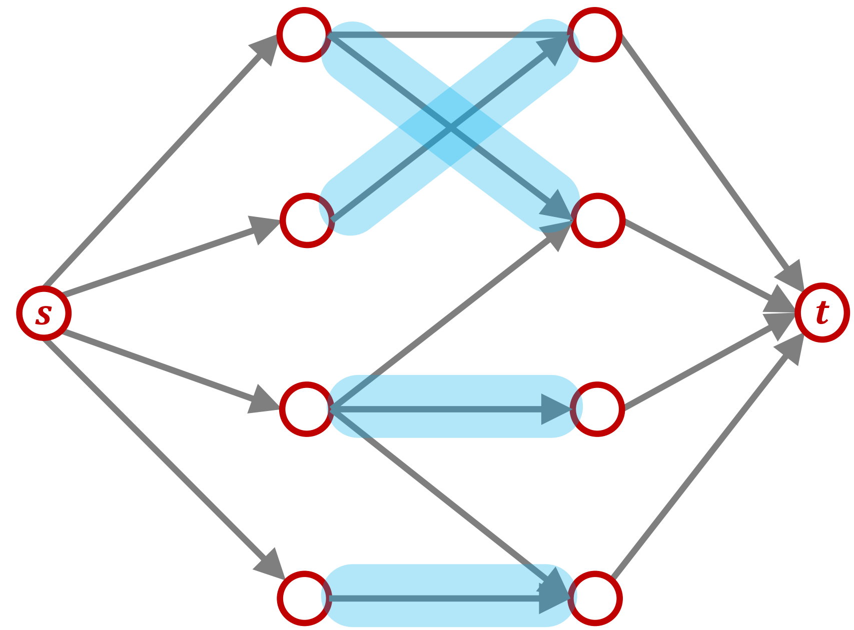
Pf:

Case 1: ($\Rightarrow$) ☑

Case 2: ($\Leftarrow$) ☑

Recall from earlier:

If the capacities are integral, then the Max-Flow is integral.

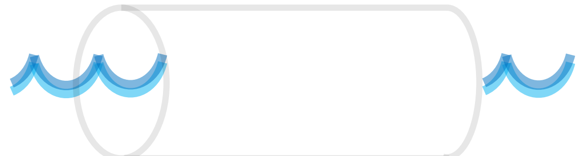


Graph G'

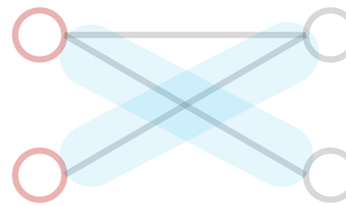
1. Let f be an **integral** flow of size n in G'.
   (all flow values 0 or 1)
2. Each $u \in L$ has **1** unit of flow on **1** outgoing edge
3. Each $v \in R$ has **1** unit of flow on **1** incoming edge
4. These edges form a matching of size n. □

Outline

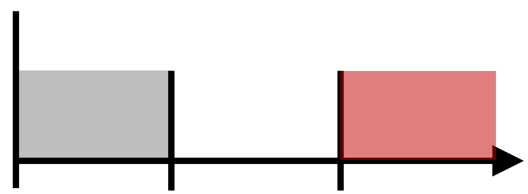
1. Maximum flow



2. Bipartite perfect matching



3. Linear programming duality



Earlier: Max-Flow = Min-Cut

Could always prove that a flow was **optimal** by showing a cut of the same value

This is a general property of LPs known as **duality**

The book calls duality a **magic trick**

$$
\begin{array}{llll}
\text{max} & 5x_1 + 4x_2 & & \\
\text{s.t.} & 2x_1 + \ \ x_2 \le 100 & \text{also} & x_1 \ge 0 \\
& \ \ x_1 \qquad\ \ \le 30 & & x_2 \ge 0 \\
& \qquad\quad\ \ x_2 \le 60 & &
\end{array}
$$

Solution: $x_1 = 20$, $x_2 = 60$, value $= 340$

Q: Can we **prove** this is optimal?

$$
\begin{array}{llll}
\text{max} & 5x_1 + 4x_2 & & \\
\text{s.t.} & 2x_1 + x_2 \le 100 & \text{also} & x_1 \ge 0 \\
 & (\ x_1 \le 30\ )\cdot 5 & & x_2 \ge 0 \\
+ & (\ \ x_2 \le 60\ )\cdot 4 & & \\
\hline
 & 5x_1 + 4x_2 \le \underbrace{5\cdot 30 + 4\cdot 60}_{390} & &
\end{array}
$$

Solution: $x_1 = 20$, $x_2 = 60$, value $= 340$

Q: Can we **prove** this is optimal?

$$
\begin{array}{llll}
\text{max} & 5x_1 + 4x_2 & & \\
\text{s.t.} & (2x_1 + x_2 \le 100) \cdot 3 & \text{also} & x_1 \ge 0 \\
 & (x_1 \le 30) \cdot 0 & & x_2 \ge 0 \\
+ & (x_2 \le 60) \cdot 1 & &
\end{array}
$$

$$5x_1 + 4x_2 \le 6x_1 + 4x_2 \le \underbrace{3 \cdot 100 + 60}_{360}$$

Solution: $x_1 = 20$, $x_2 = 60$, value $= 340$

Q: Can we **prove** this is optimal?

$$
\begin{array}{lllll}
\text{max} & 5x_1 + 4x_2 & & & \\
\text{s.t.} & (2x_1 + x_2 \leq 100) \cdot 5/2 & \text{also} & x_1 \geq 0 \\
 & (x_1 \leq 30) \cdot 0 & & x_2 \geq 0 \\
+ & (x_2 \leq 60) \cdot 3/2 & & \\
\hline
\end{array}
$$

$$5x_1 + 4x_2 \leq \underbrace{\frac{5}{2} \cdot 100 + \frac{3}{2} \cdot 60}_{340}$$

Solution: $x_1 = 20, x_2 = 60$, value $= 340$

Q: Can we **prove** this is optimal?

Primal LP:

$$
\begin{array}{llll}
\max & 5x_1 + 4x_2 & & \\
\text{s.t.} & (2x_1 + x_2 \le 100) \cdot y_1 & \text{also} & x_1 \ge 0 \\
 & (x_1 \le 30) \cdot y_2 & & x_2 \ge 0 \\
 & +\ (x_2 \le 60) \cdot y_3 & &
\end{array}
$$

$$(2y_1 + y_2) \cdot x_1 + (y_1 + y_3) \cdot x_2 \le 100 \cdot y_1 + 30 \cdot y_2 + 60 \cdot y_3$$

Dual LP:

$$
\begin{array}{ll}
\min & 100 \cdot y_1 + 30 \cdot y_2 + 60 \cdot y_3 \\
\text{s.t.} & y_1, y_2, y_3 \ge 0 \\
 & 5 \le 2y_1 + y_2 \\
 & 4 \le y_1 + y_3
\end{array}
$$

By construction: $5x_1 + 4x_2 \le 100 \cdot y_1 + 30 \cdot y_2 + 60 \cdot y_3$

Primal LP Opt $\le$ **Dual LP Opt**

Primal LP

max $\quad \begin{bmatrix} c^T \end{bmatrix} \cdot \begin{bmatrix} x \end{bmatrix}$

s.t. $\quad \begin{bmatrix} A \end{bmatrix} \cdot \begin{bmatrix} x \end{bmatrix} \leq \begin{bmatrix} b \end{bmatrix}$

and $\quad \begin{bmatrix} x \end{bmatrix} \geq 0$

Dual LP

max $\quad \begin{bmatrix} b^T \end{bmatrix} \cdot \begin{bmatrix} y \end{bmatrix}$

s.t. $\quad \begin{bmatrix} A^T \end{bmatrix} \cdot \begin{bmatrix} y \end{bmatrix} \geq \begin{bmatrix} c \end{bmatrix}$

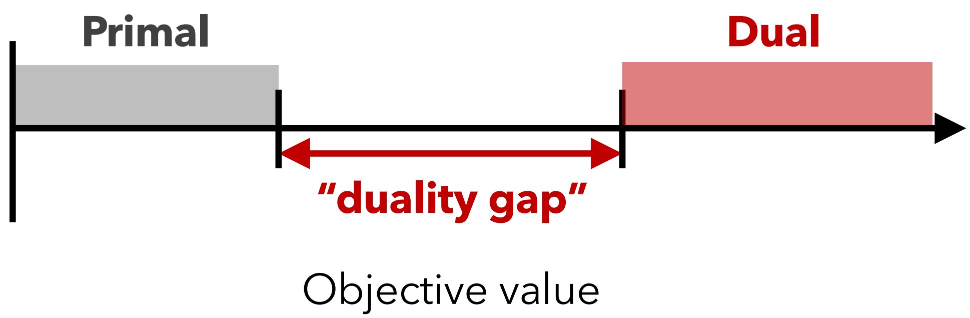
and $\quad \begin{bmatrix} y \end{bmatrix} \geq 0$

Thm: **(Weak duality)** all feasible solutions ***x*** to primal LP $\leq$ all feasible solutions ***y*** to dual LP

Pf:
$$\begin{bmatrix} c^T \end{bmatrix} \begin{bmatrix} x \end{bmatrix} \leq \begin{bmatrix} y^T \end{bmatrix} \begin{bmatrix} A \end{bmatrix} \begin{bmatrix} x \end{bmatrix} \leq \begin{bmatrix} y^T \end{bmatrix} \begin{bmatrix} b \end{bmatrix} = \begin{bmatrix} b^T \end{bmatrix} \begin{bmatrix} y \end{bmatrix} \qquad \square$$

Cor: **Primal LP OPT $\leq$ Dual LP OPT**

Weak duality:



Thm: **(Strong duality)**

If the **Primal LP Opt** is bounded,
then **Primal LP Opt** = **Dual LP Opt**

∴ **duality gap** = 0

Example: Max-Flow **=** Min-Cut (in recitation)

LP ⟷ **LP**
dual

LP duality history

George Dantzig

Co-inventor of LPs, inventor of simplex, Berkeley grad student and faculty

Was taking a statistics class

Professor wrote two of the most famous unsolved problems in statistics on the board

But Dantzig arrived late, mistook them for homework

Turned in solutions a few days later,

said they "seemed to be a little harder than usual"

LP duality history

George Dantzig

Co-inventor of LPs, inventor of simplex, Berkeley grad student and faculty

"Let me tell you about my newest invention: linear programming"

"Oh that!"

(Lectures Dantzig about linear programming for 1.5 hours, invents linear program duality)

"It's equivalent to zero-sum games, which I have also recently invented"

John von Neumann

All-time great mathematician