CSE 421
Introduction to Algorithms

Lecture 19: More Flow Applications

Last time: Circulation with Demands

- Single commodity, directed graph $\boldsymbol{G = (V, E)}$
- Each node $\boldsymbol{v}$ has an associated demand $\boldsymbol{d(v)}$
  - Needs to receive an amount of the commodity: demand $\boldsymbol{d(v) > 0}$
  - Supplies some amount of the commodity: "demand" $\boldsymbol{d(v) < 0}$ (amount = $|\boldsymbol{d(v)}|$)
- Each edge $\boldsymbol{e}$ has a capacity $\boldsymbol{c(e) \geq 0}$.
- Nothing lost: $\sum_{v} \boldsymbol{d(v) = 0}$.

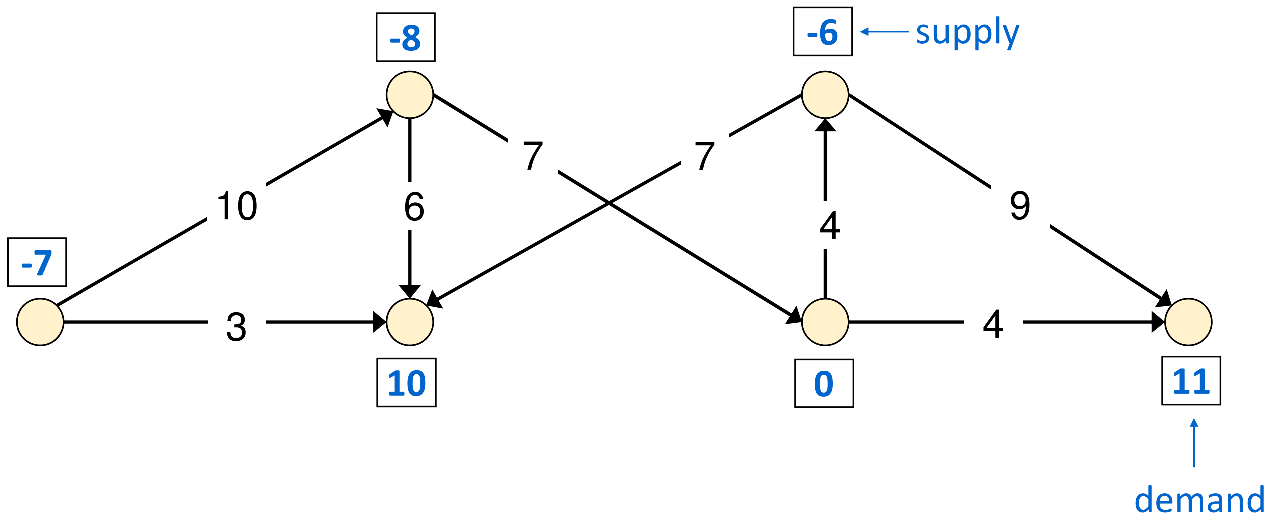
Defn: A circulation for $\boldsymbol{(G, c, d)}$ is a flow function $\boldsymbol{f}: \boldsymbol{E} \to \mathbb{R}$ meeting all the capacities, $\boldsymbol{0 \leq f(e) \leq c(e)}$, and demands:
$\sum_{e \text{ into } v} \boldsymbol{f(e)} - \sum_{e \text{ out of } v} \boldsymbol{f(e)} = \boldsymbol{d(v)}$.

Circulation with Demands: Given $\boldsymbol{(G, c, d)}$, does it have a circulation? If so, find it.

Last time: Circulation with Demands

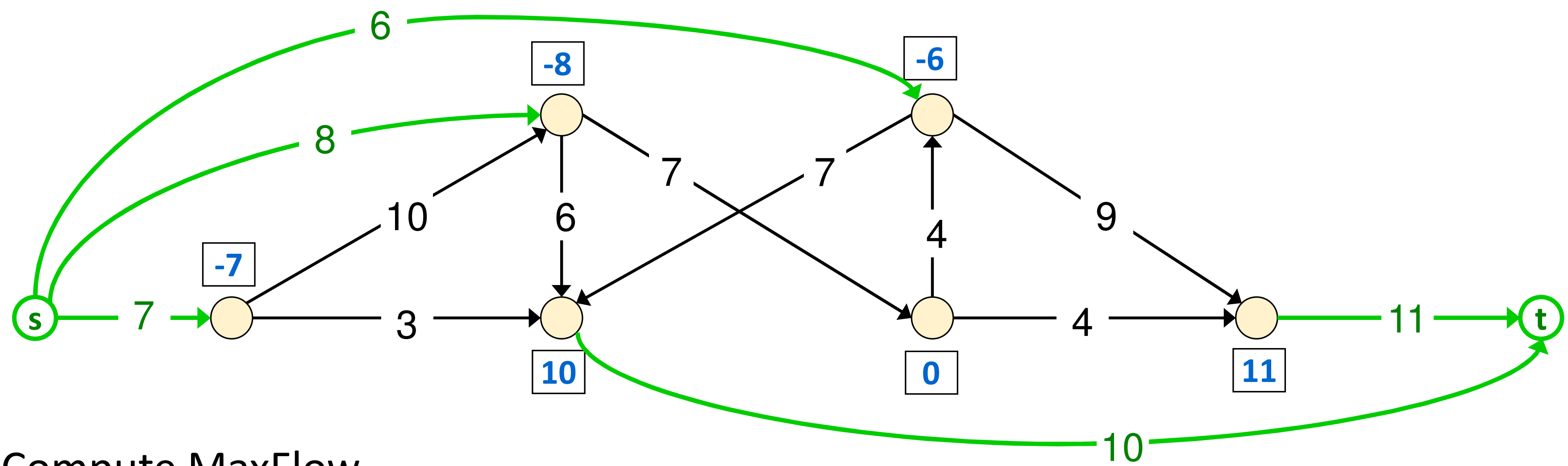
Defn: Total supply $D = \sum_{v:\, d(v)<0} |d(v)| = -\sum_{v:\, d(v)<0} d(v)$.

Necessary condition: $\sum_{v:\, d(v)>0} d(v) = D$ (no supply is lost)



Circulation with Demands using Network Flow

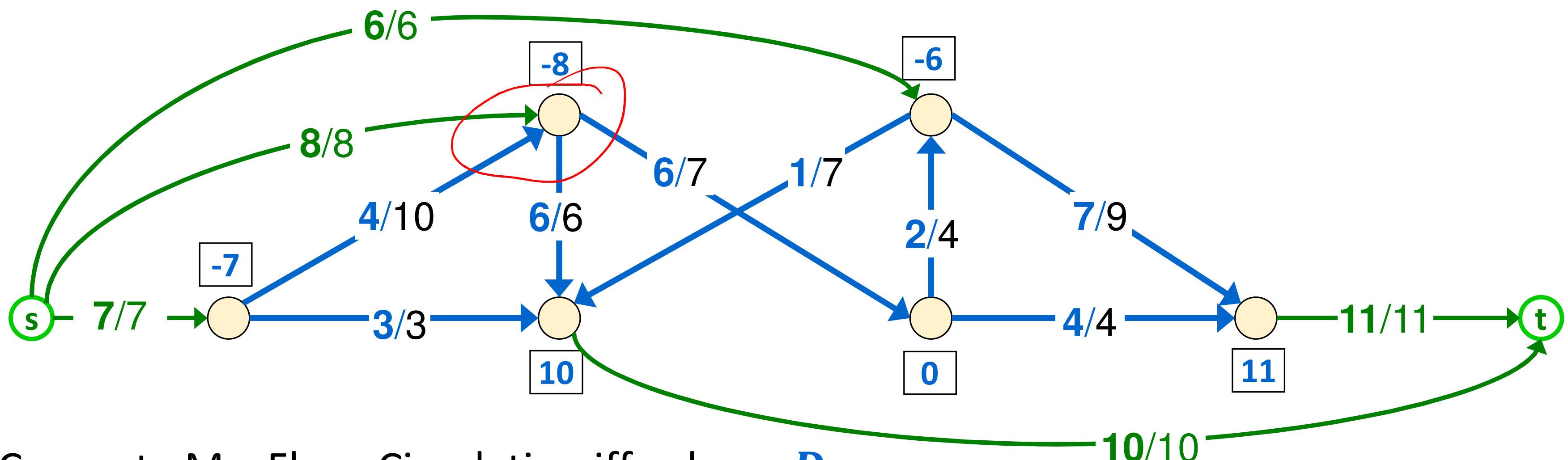
- Add new source s and sink t.
- Add edge (s, v) for all supply nodes v with capacity $|d(v)|$.
- Add edge (v, t) for all demand nodes v with capacity $d(v)$.



- Compute MaxFlow.

Circulation with Demands using Network Flow

- MaxFlow $\leq D$ based on cuts out of s or into t.
- If MaxFlow $= D$ then all supply/demands satisfied.

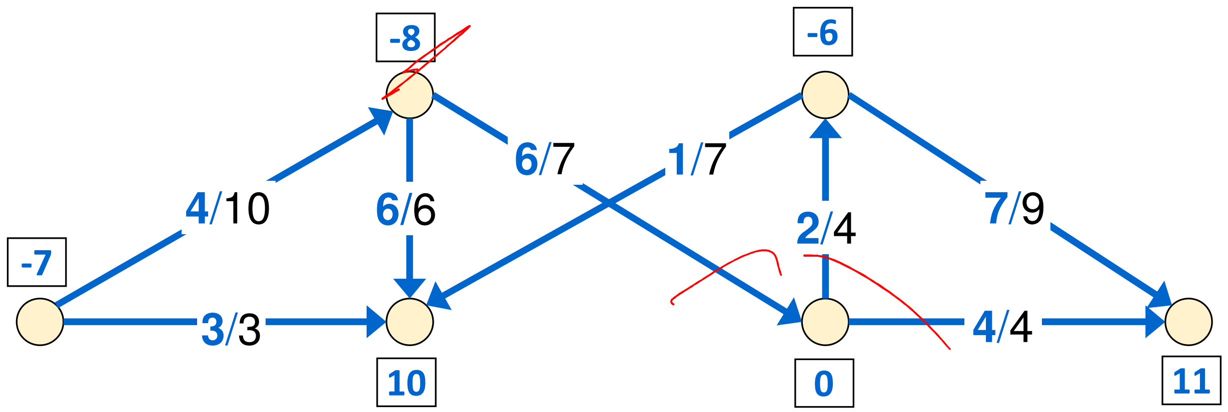


- Compute MaxFlow. Circulation iff value = D

Circulation with Demands using Network Flow

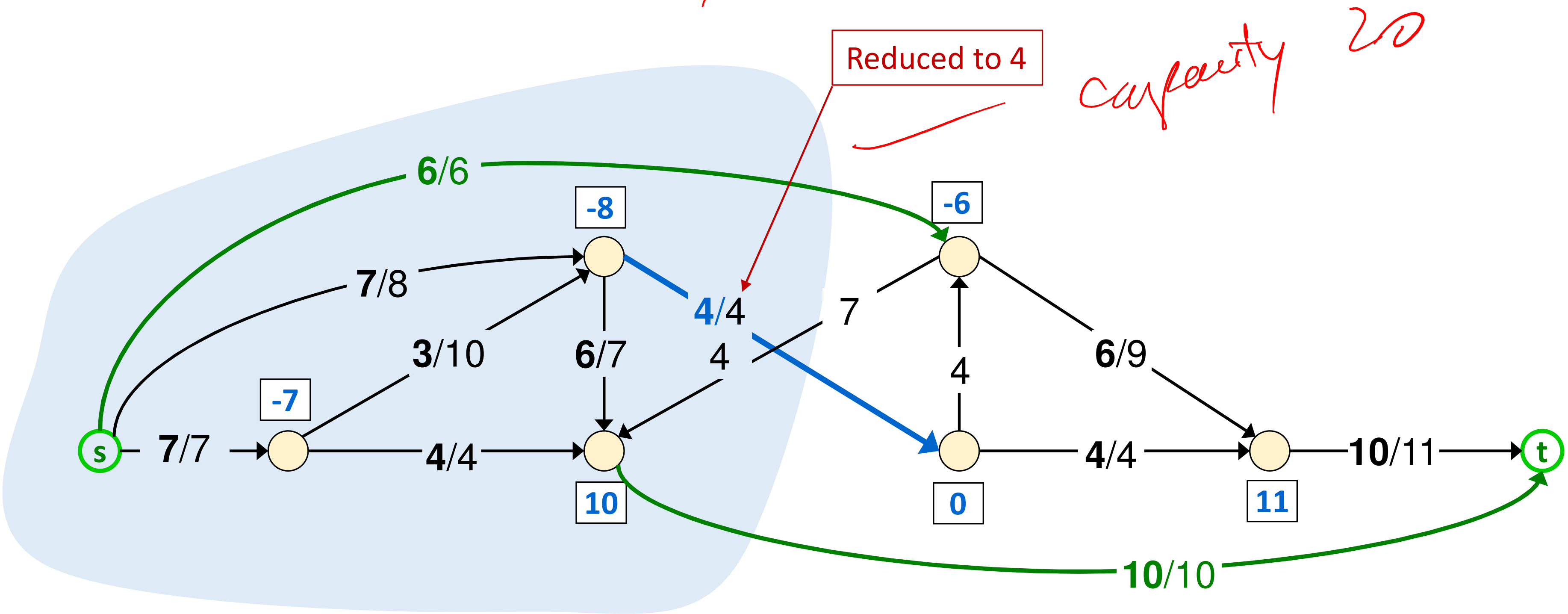
Circulation = flow on original edges

Circulations only need integer flows



Circulation with Demands using Network Flow

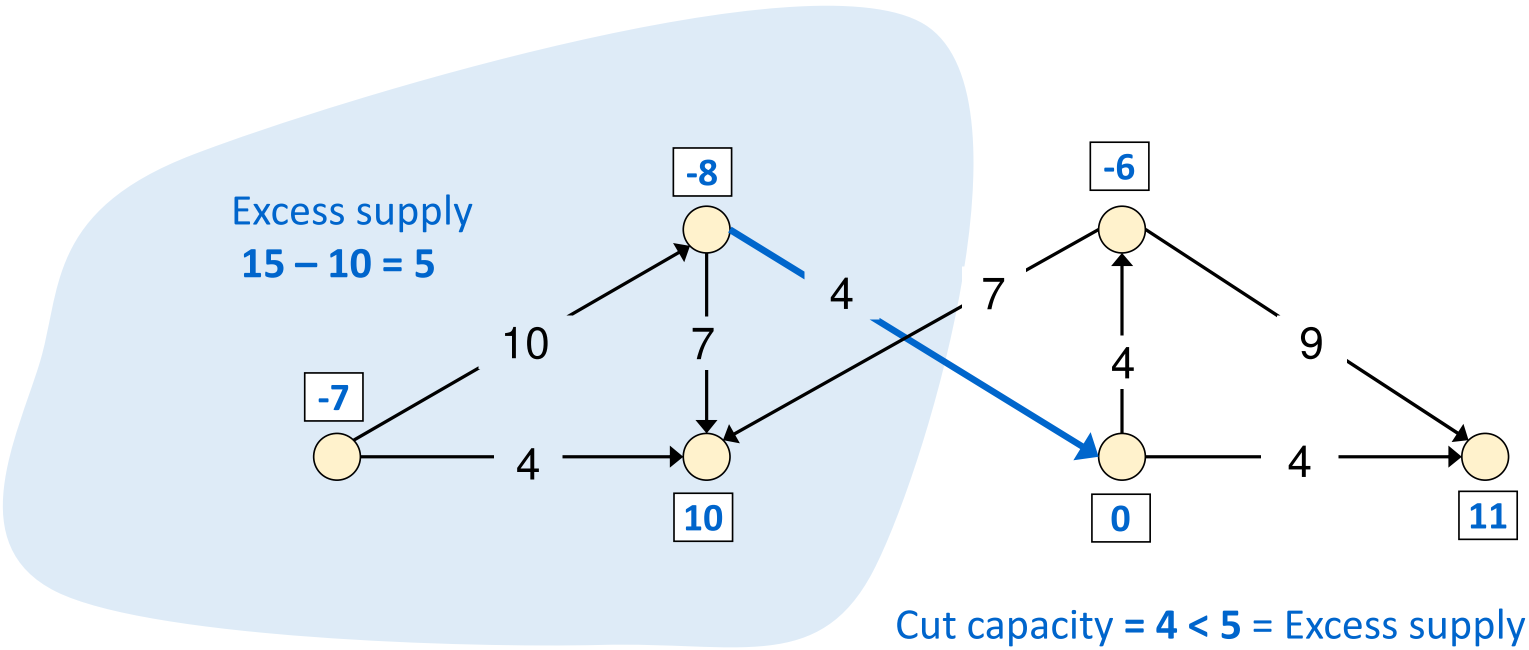
When does a circulation not exist? MaxFlow $< \boldsymbol{D}$ iff MinCut $< \boldsymbol{D}$.



Circulation with Demands using Network Flow

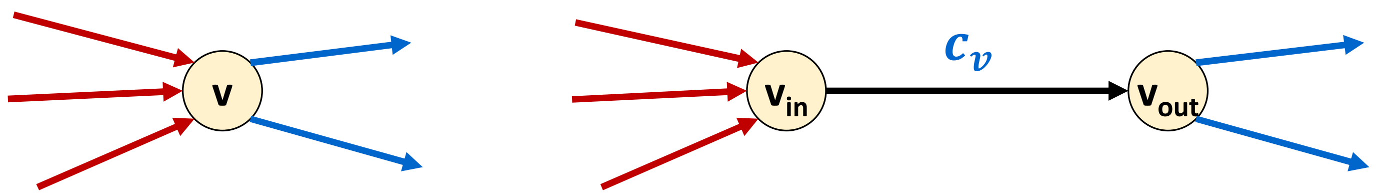
When does a circulation not exist? MaxFlow $< D$ iff MinCut $< D$.

Equivalent to excess supply on "source" side of cut smaller than cut capacity.



Some general ideas for using MaxFlow/MinCut

- If no source/sink, add them with appropriate capacity depending on application
- Sometimes can have edges with no capacity limits
  - Infinite capacity (or, equivalently, very large integer capacity)
- Convert undirected graphs to directed ones
- Can remove unnecessary flow cycles in answers
- Another idea:
  - To use them for vertex capacities c_v
    - Make two copies of each vertex v named v_{in}, v_{out}



Kinds of applications

- So far we mostly have focused on flow-like problems

- Applications that involve cut problems are also important...

Image Segmentation

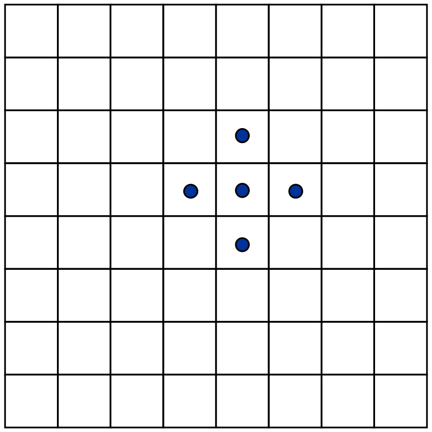
Image Segmentation

Image segmentation:

Given: an Image

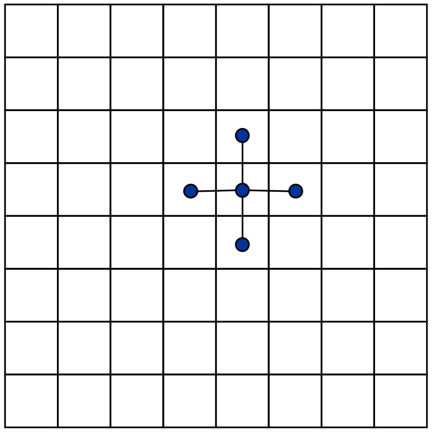
- a grid of pixels with RGB values

Divide image into coherent regions.



Example: Three people standing in front of complex background scene.
Identify each person as a coherent object.

Image Segmentation



Foreground / background segmentation:

Given: A grid V of pixels, E set of pairs of neighboring pixels.

- $a_i \geq 0$ is likelihood pixel i is foreground.
- $b_i \geq 0$ is likelihood pixel i is background.
- For $(i, j) \in E$, $p_{ij} \geq 0$ is separation penalty for labeling one of i and j as foreground, and the other as background.

Label each pixel in image as belonging to foreground (in A) or background (in B)

Goals: Maximize

Accuracy: if $a_i > b_i$ in isolation, prefer to label i in foreground.

Smoothness: if many neighbors of i are labeled foreground, we should be inclined not to label i as background.

so… **Find:** partition (A, B) that maximizes $\sum_{i \in A} a_i + \sum_{i \in B} b_i - \sum_{(i,j) \in E,\ |A \cap \{i,j\}|=1} p_{ij}$

Image Segmentation

Issues with formulating as min cut problem:

- Maximization.
- No source or sink.
- Undirected graph.

But maximizing
$$\sum_{i\in A} a_i + \sum_{i\in B} b_i - \sum_{\substack{(i,j)\in E \\ |A\cap\{i,j\}|=1}} p_{ij}$$

The sum in red is a constant

is equivalent to maximizing
$$\sum_{i\in V} a_i - \sum_{i\in B} a_i + \sum_{i\in V} b_i - \sum_{i\in A} b_i - \sum_{\substack{(i,j)\in E \\ |A\cap\{i,j\}|=1}} p_{ij}$$

or, alternatively, minimizing
$$\sum_{i\in A} b_i + \sum_{i\in B} a_i + \sum_{\substack{(i,j)\in E \\ |A\cap\{i,j\}|=1}} p_{ij}$$

Image Segmentation

Minimize $\sum_{i \in A} b_i + \sum_{i \in B} a_i + \sum_{\substack{(i,j) \in E \\ |A \cap \{i,j\}| = 1}} p_{ij}$

Formulate as min cut problem.

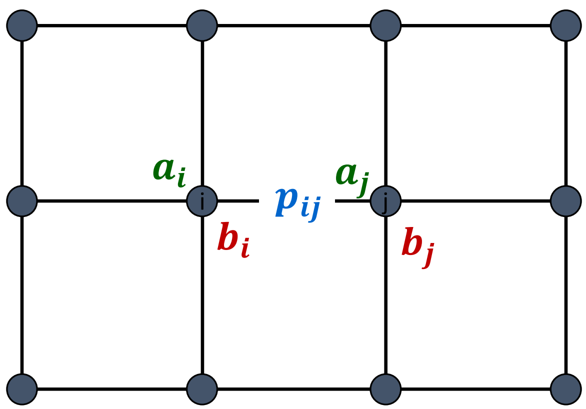


Image Segmentation

Minimize $\sum_{i \in A} b_i + \sum_{i \in B} a_i + \sum_{\substack{(i,j) \in E \\ |A \cap \{i,j\}| = 1}} p_{ij}$

Formulate as min cut problem.

- Add source s to correspond to foreground, edges (s, i) with capacity a_i; add sink t to correspond to background, edges (j, t) with capacity b_j.

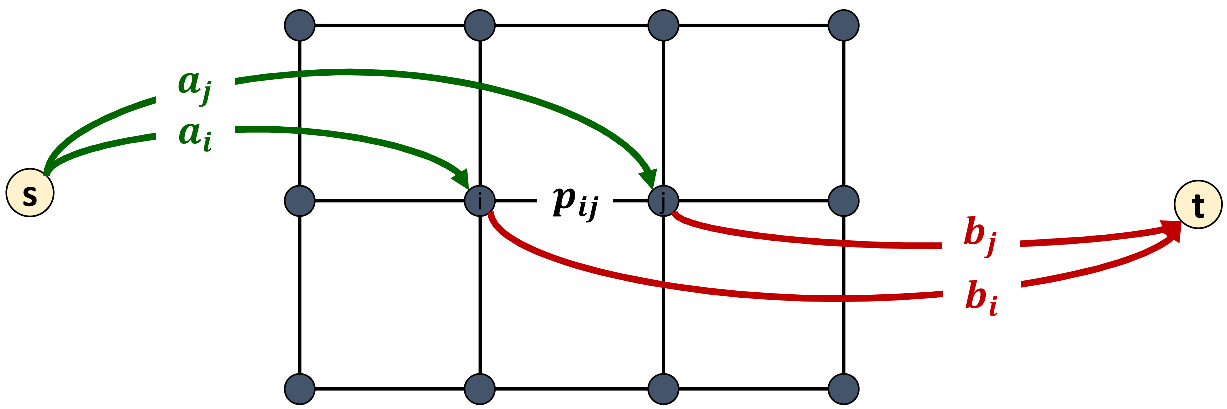


Image Segmentation

Minimize $\sum_{i \in A} b_i + \sum_{i \in B} a_i + \sum_{\substack{(i,j) \in E \\ |A \cap \{i,j\}|=1}} p_{ij}$

Formulate as min cut problem.

- Add source s to correspond to foreground, edges (s, i) with capacity a_i; add sink t to correspond to background, edges (j, t) with capacity b_j.
- Use two anti-parallel edges instead of undirected edge, capacity p_{ij}.
- $G' = (V', E')$.

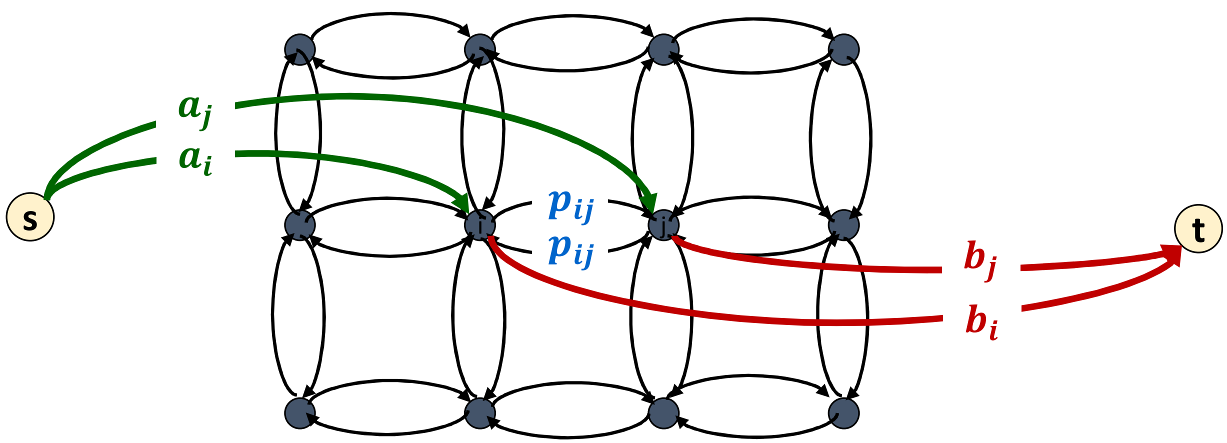
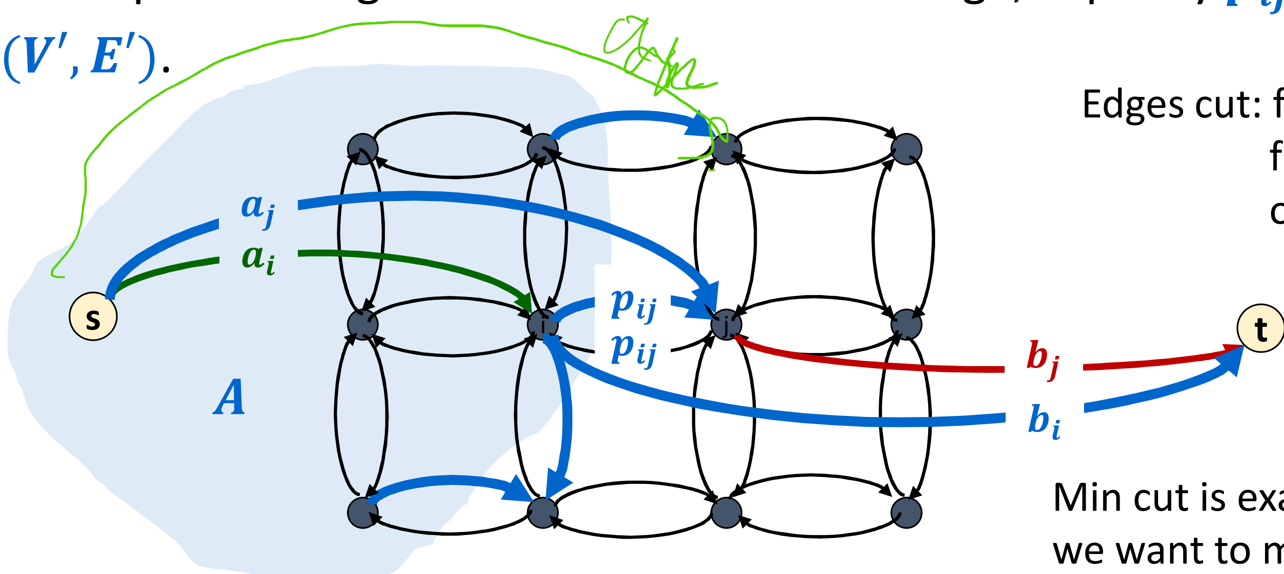


Image Segmentation

Minimize $\sum_{i\in A} b_i + \sum_{i\in B} a_i + \sum_{\substack{(i,j)\in E \\ |A\cap\{i,j\}|=1}} p_{ij}$

Formulate as min cut problem.

- Add source s to correspond to foreground, edges (s, i) with capacity a_i; add sink t to correspond to background, edges (j, t) with capacity b_j.
- Use two anti-parallel edges instead of undirected edge, capacity p_{ij}.
- $G' = (V', E')$.



Edges cut: from s leaving A
from A to t
one direction for p_{ij}

Min cut is exactly the quantity we want to minimize!

Project Selection

Project Selection

Project Selection:

Input: Set P of possible projects; each project $v \in P$ has an associated revenue p_v which can be negative.

- Some projects generate money so $p_v > 0$; e.g., create interactive e-commerce interface, redesign web page
- others cost money so $p_v < 0$; e.g., upgrade computers, get site license

Set E of "prerequisites": If $(v, w) \in E$, can't do project v and unless also do project w.

Defn: A subset of projects is feasible iff the prerequisite of every project in A also belongs to A.

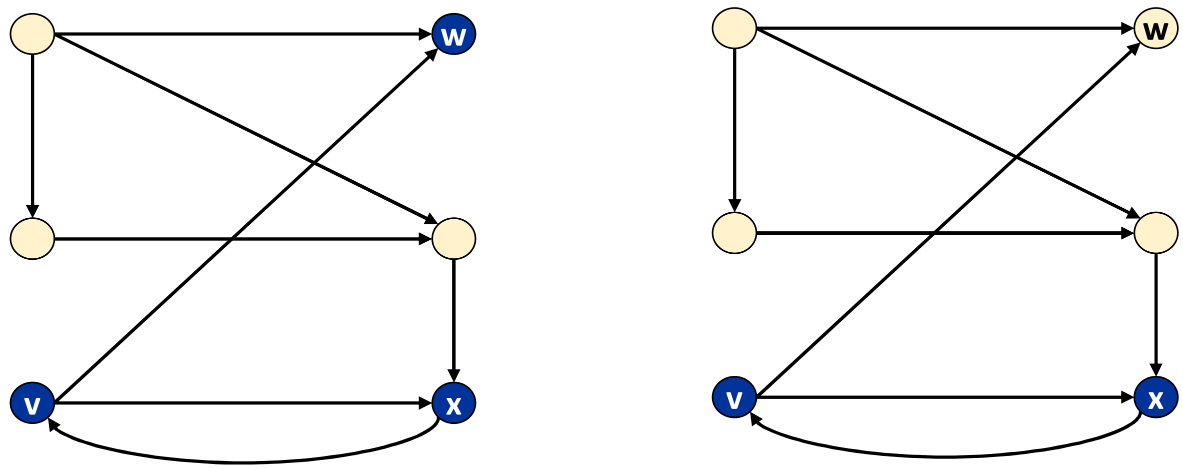
Find: A feasible subset of projects $A \subseteq P$ that maximizes total revenue.

Note: "prerequisites" may have nothing to do with time. E may include cycles.

Project Selection: Prerequisite Graph

Prerequisite graph:

- Include an edge from v to w if can't do v without also doing w.
- $\{v, w, x\}$ is feasible subset of projects.
- $\{v, x\}$ is infeasible subset of projects.

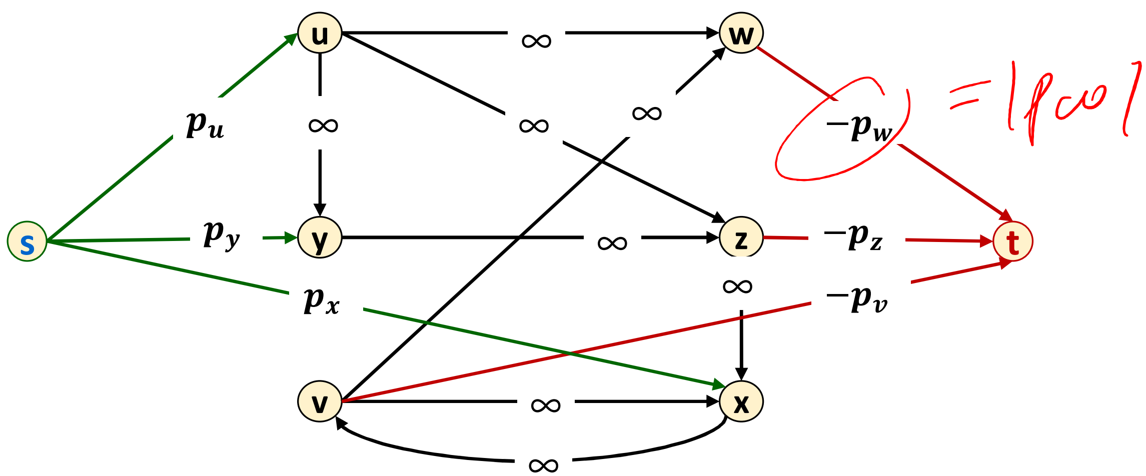


feasible   infeasible

Project Selection: Min Cut Formulation

Min cut formulation:

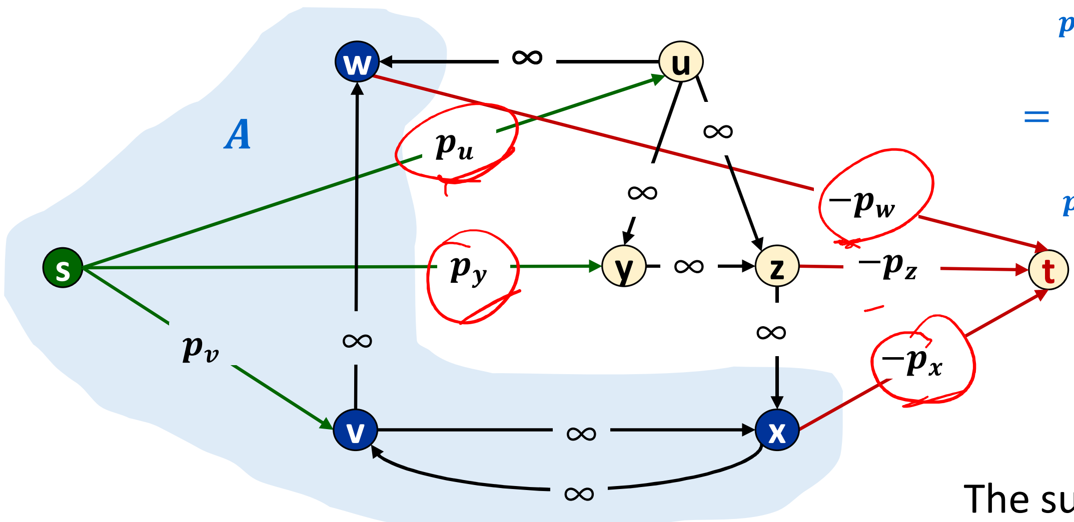
- Assign capacity ∞ to all prerequisite edges.
- Add edge (s, v) with capacity p_v if $p_v > 0$.
- Add edge (v, t) with capacity $|p_v| = -p_v$ if $p_v < 0$.
- For notational convenience, define $p_s = p_t = 0$.



Project Selection: Min Cut Formulation

Claim: (A, B) is min cut iff $A - \{s\}$ is optimal set of projects.

- Infinite capacity edges ensure $A - \{s\}$ is feasible. (No original edges leave A.)
- Max revenue because:



$$c(A,B) = \sum_{\substack{v \in B \\ p(v)>0}} p_v + \sum_{\substack{v \in A \\ p(v)<0}} (-p(v))$$

$$= \sum_{\substack{v \in B \\ p(v)>0}} p_v + \sum_{\substack{v \in A \\ p(v)>0}} p_v - \sum_{\substack{v \in A \\ p(v)>0}} p(v) - \sum_{\substack{v \in A \\ p(v)<0}} p(v)$$

$$= \sum_{\substack{v \in V \\ p(v)>0}} p_v - \sum_{v \in A} p(v)$$

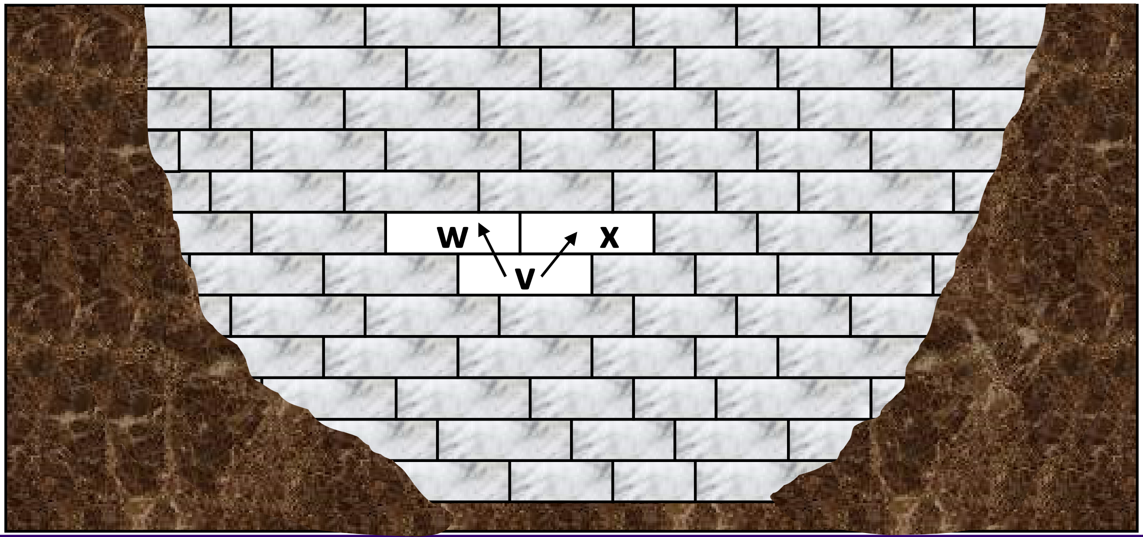
what we want to maximize

The sum in red is constant so minimizing $c(A, B)$ is the same as maximizing $\sum_{v \in A} p(v)$.

Also known as Strip Mining problem

Open-pit mining. (studied since early 1960s)

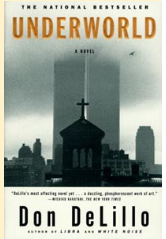
- Blocks of earth are extracted from surface to retrieve ore.
- Each block v has net value p_v = value of ore − processing cost.
- Can't remove block v without removing w and x.



Baseball Elimination?

"See that thing in the paper last week about Einstein? . . .
Some reporter asked him to figure out the mathematics of the pennant race. You know, one team wins so many of their remaining games, the other teams win this number or that number. What are the myriad possibilities? Who's got the edge?"
"The hell does he know?"
"Apparently not much. He picked the Dodgers to eliminate the Giants last Friday."

- Don DeLillo, Underworld



Baseball Elimination

- Though you probably don't care at all about baseball or sports in general, the way that the solution works is interesting.
  - This particular problem is a bit old style since baseball scheduling doesn't work this way any more…
- Near the end of a season
  - Sportswriters use simple notions to tell which teams can be eliminated from getting a top place finish:
    - "magic number", "elimination number", etc.
- These are not accurate
  - We can do better with network flow

Baseball Elimination: Scenario

Team i	Wins w_i	Losses l_i	To play r_i	Against = r_{ij}			
				Tex	Hou	Sea	Oak
Texas	**83**	**71**	**8**	**-**	**1**	**6**	**1**
Houston	**80**	**79**	**3**	**1**	**-**	**0**	**2**
Seattle	**78**	**78**	**6**	**6**	**0**	**-**	**0**
Oakland	**77**	**82**	**3**	**1**	**2**	**0**	**-**

- Which teams have a chance of finishing the season with most wins?
  - Oakland eliminated since it can finish with at most 80 wins, but Texas already has 83.
  - If $w_i + r_i < w_j \Rightarrow$ team i eliminated.
  - Only reason sports broadcasters appear to be aware of.
  - Sufficient, but not necessary!

Baseball Elimination: Scenario

Team i	Wins w_i	Losses l_i	To play r_i	Against = r_{ij}			
				Tex	Hou	Sea	Oak
Texas	**83**	**71**	**8**	**-**	**1**	**6**	**1**
Houston	**80**	**79**	**3**	**1**	**-**	**0**	**2**
Seattle	**78**	**78**	**6**	**6**	**0**	**-**	**0**
Oakland	**77**	**82**	**3**	**1**	**2**	**0**	**-**

- Which teams have a chance of finishing the season with most wins?
  - Houston can win **83** games, but is still eliminated . . .
  - If Texas doesn't get to **84** wins then Seattle will get 6 more wins and finish with **84** wins.

- The answer depends on more than current wins and # of remaining games
  - It also depends on all the games that are being played.

Baseball Elimination

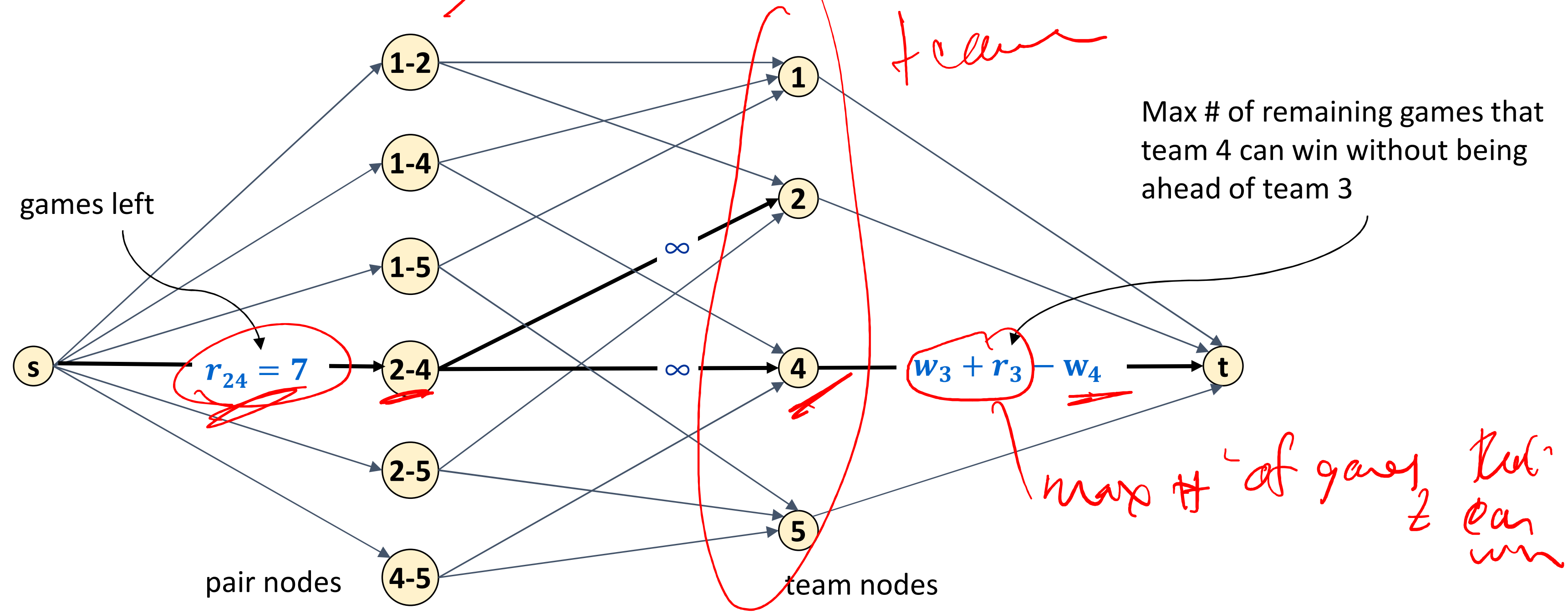
Baseball elimination problem:

- Set of teams $\boldsymbol{S}$.
- Distinguished team $\boldsymbol{z} \in \boldsymbol{S}$.
- Team $\boldsymbol{x}$ has won $\boldsymbol{w_x}$ games already.
- Teams $\boldsymbol{x}$ and $\boldsymbol{y}$ play each other $\boldsymbol{r_{xy}}$ additional times.
- Is there any outcome of the remaining games in which team $\boldsymbol{z}$ finishes with the most (or tied for the most) wins?

Baseball Elimination: Max Flow Formulation

Can team 3 finish with most wins?

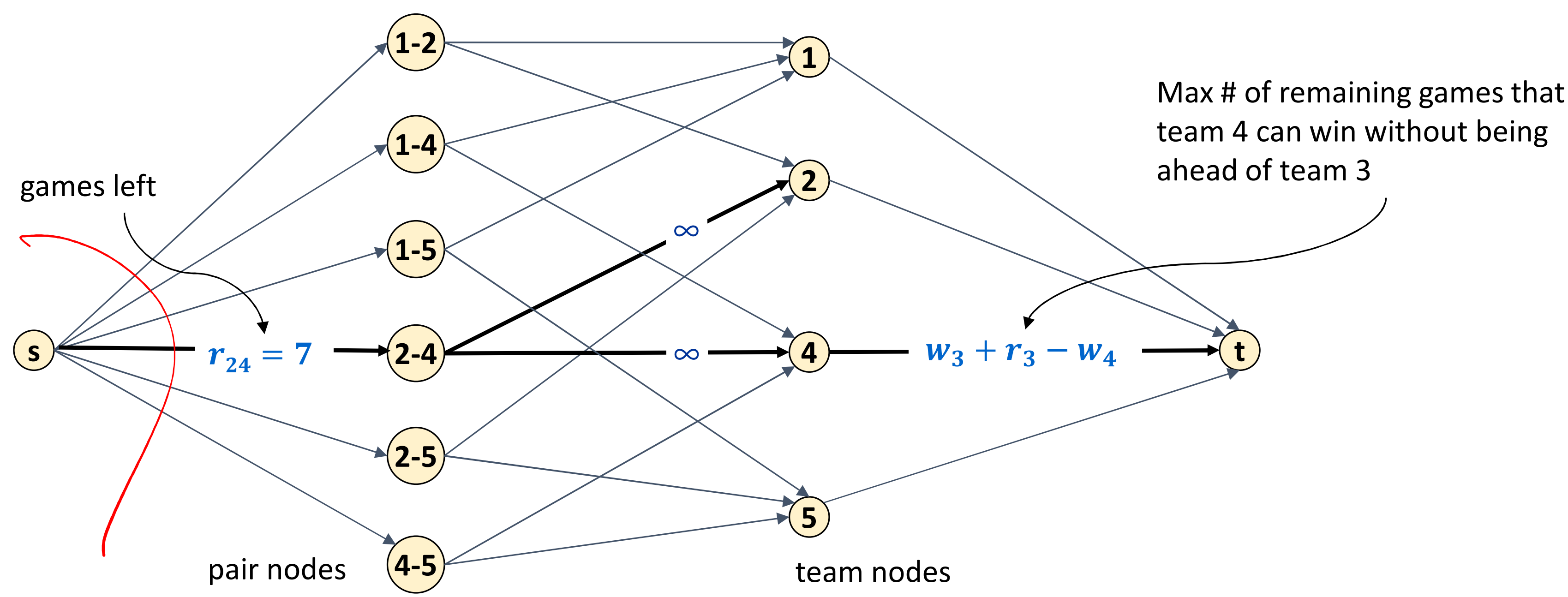
- Assume team 3 wins all remaining games $\Rightarrow w_3 + r_3$ wins.
- Divide up remaining games so that all teams have $\leq w_3 + r_3$ wins.



Baseball Elimination: Max Flow Formulation

Theorem: Team 3 is not eliminated iff max flow equals capacity leaving source.

- Integrality theorem implies that each remaining x-y game counts as a win for x or y.
- Capacity on (x, t) edge ensures no team wins too many games.



Baseball Elimination: Explanation for Sports Writers

Team i	Wins w_i	Losses l_i	To play r_i	Against = r_{ij}				
				NY	Bal	Bos	Tor	Det
NY	75	59	28	-	3	8	7	3
Baltimore	71	63	28	3	-	2	7	4
Boston	69	66	27	8	2	-	0	0
Toronto	63	72	27	7	7	0	-	-
Detroit	49	86	27	3	4	0	0	-

AL East: August 30, 1996

Which teams have a chance of finishing the season with most wins?

- Detroit could finish season with **49 + 27 = 76** wins.

Certificate of elimination. T = {NY, Bal, Bos, Tor}

- Have already won $w(T)$ =**75+71+69+63=278** games.
- Must win at least $r(T)$ = **3+8+7+2+7=27** more among themselves.
- Average team in T wins at least **305/4** > **76** games.

Baseball Elimination: Explanation for Sports Writers

Defn: Given a set T of teams define

- $w(T) = \sum_{x \in T} w_x$ total number of wins for teams in T
- $r(T) = \sum_{\{x,y\} \subseteq T} r_{xy}$ total remaining games between teams in T

We say that T eliminates team z iff $\frac{w(T)+r(T)}{|T|} > w_z + r_z$ since an average team in T will win more than $w_z + r_z$ games.

Theorem [Hoffman-Rivlin 1967]: Team z is eliminated
$\Leftrightarrow$ there is some set T of teams that eliminates z (as defined above).

Proof: $\Leftarrow$ Shown above

$\Rightarrow$ Choose T to be the set of teams on the source side of the min cut...

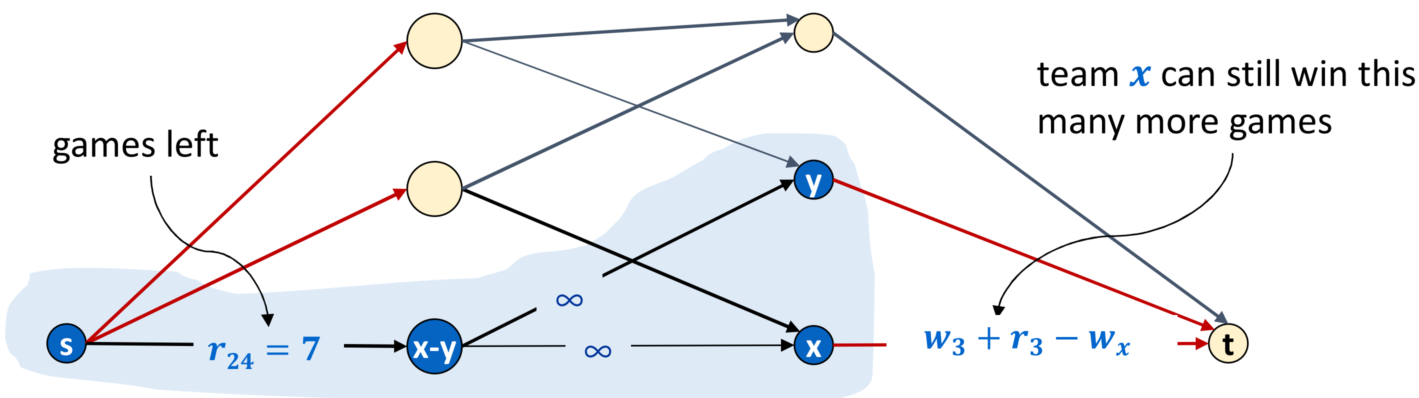
Baseball Elimination: Explanation for Sports Writers

Proof of ⇒: Assume that z is eliminated

Let T = team nodes in A for minimum cut (A, B) with capacity $< \sum_{xy} r_{xy}$.

Claim: x–$y \in A \Leftrightarrow$ both $x \in A$ and $y \in A$ (equivalently $x \in T$ and $y \in T$).

- infinite capacity edges ensure that if x–$y \in A$ then $x \in A$ and $y \in A$
- if $x \in A$ and $y \in A$ but x–$y \notin A$, then adding x–y to A decreases cut capacity by r_{xy}.



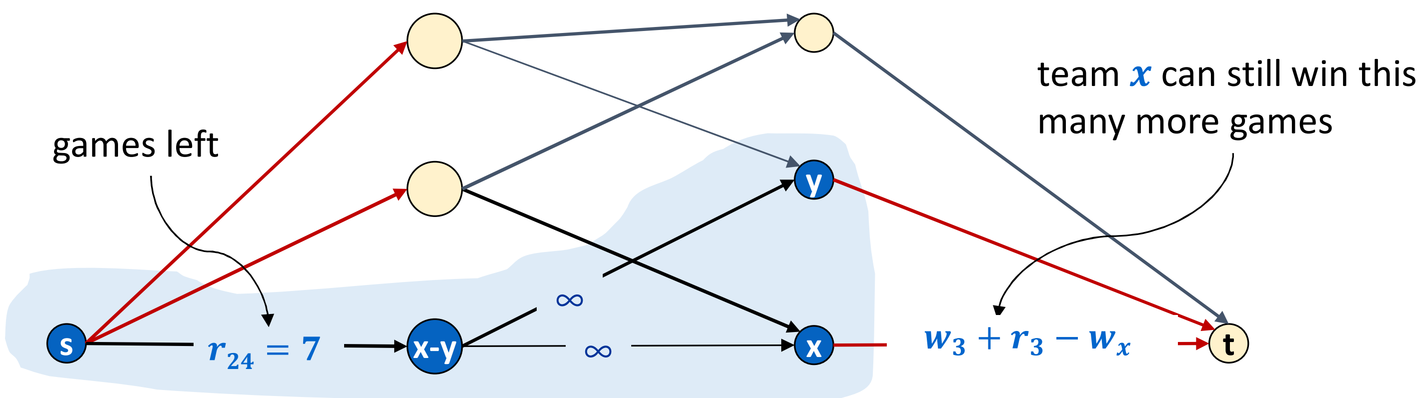
Baseball Elimination: Explanation for Sports Writers

Proof of ⇒: Assume that z is eliminated.

Let T = team nodes in A for minimum cut (A, B) with capacity $< \sum_{xy} r_{xy}$.

Claim: x–$y \in A \iff$ both $x \in A$ and $y \in A$ (equivalently $x \in T$ and $y \in T$).

Then $c(A, B) = \sum_{xy} r_{xy} - r(T) + |T|(w_z + r_z) - w(T)$



Baseball Elimination: Explanation for Sports Writers

Proof of ⇒: Assume that z is eliminated.

Let T = team nodes in A for minimum cut (A, B) with capacity $< \sum_{xy} r_{xy}$.

Claim: x–$y \in A \Leftrightarrow$ both $x \in A$ and $y \in A$ (equivalently $x \in T$ and $y \in T$).

Then $c(A, B) = \sum_{xy} r_{xy} - r(T) + |T|(w_z + r_z) - w(T)$

Now $c(A, B) < \sum_{xy} r_{xy}$ implies that $r(T) - |T|(w_z + r_z) + w(T) > 0$.

Rearranging, we have $r(T) + w(T) > |T|(w_z + r_z)$ so $\frac{w(T)+r(T)}{|T|} > w_z + r_z$ which means that T eliminates z.