Algorithm II

7. Network Flow II

WU Xiaokun 吴晓堃

xkun.wu [at] gmail

Max-flow and min-cut applications

Max-flow and min-cut problems are widely applicable model.

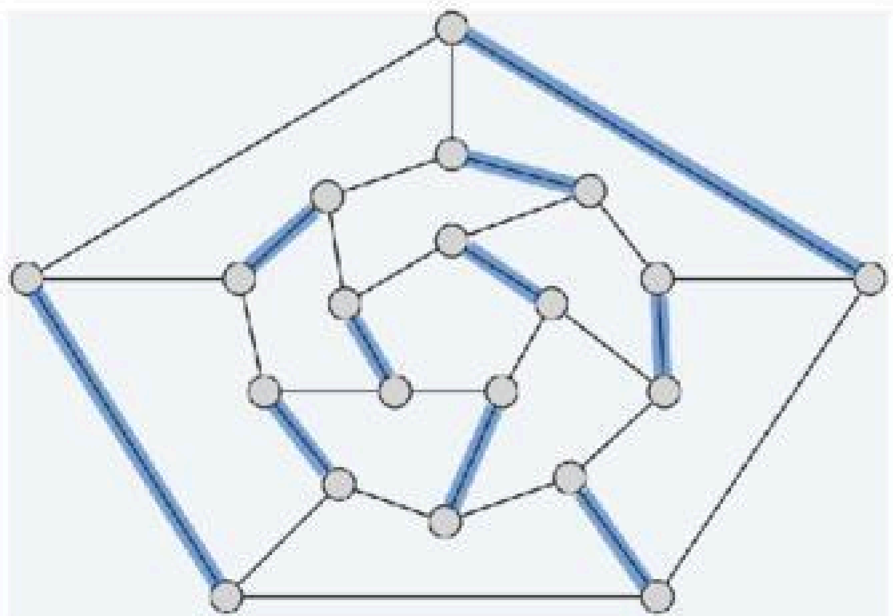
- Data mining.
- Open-pit mining.
- Bipartite matching.
- Network reliability.
- Baseball elimination.
- Image segmentation.
- Network connectivity.
- Markov random fields.
- Distributed computing.
- Security of statistical data.
- Egalitarian stable matching.
- Network intrusion detection.
- Multi-camera scene reconstruction.
- Sensor placement for homeland security.
- Many, many, more.

Bipartite matching

Max matching

Def. Given an undirected graph $G = (V, E)$, subset of edges $M \subseteq E$ is a **matching** if each node appears in at most one edge in M.

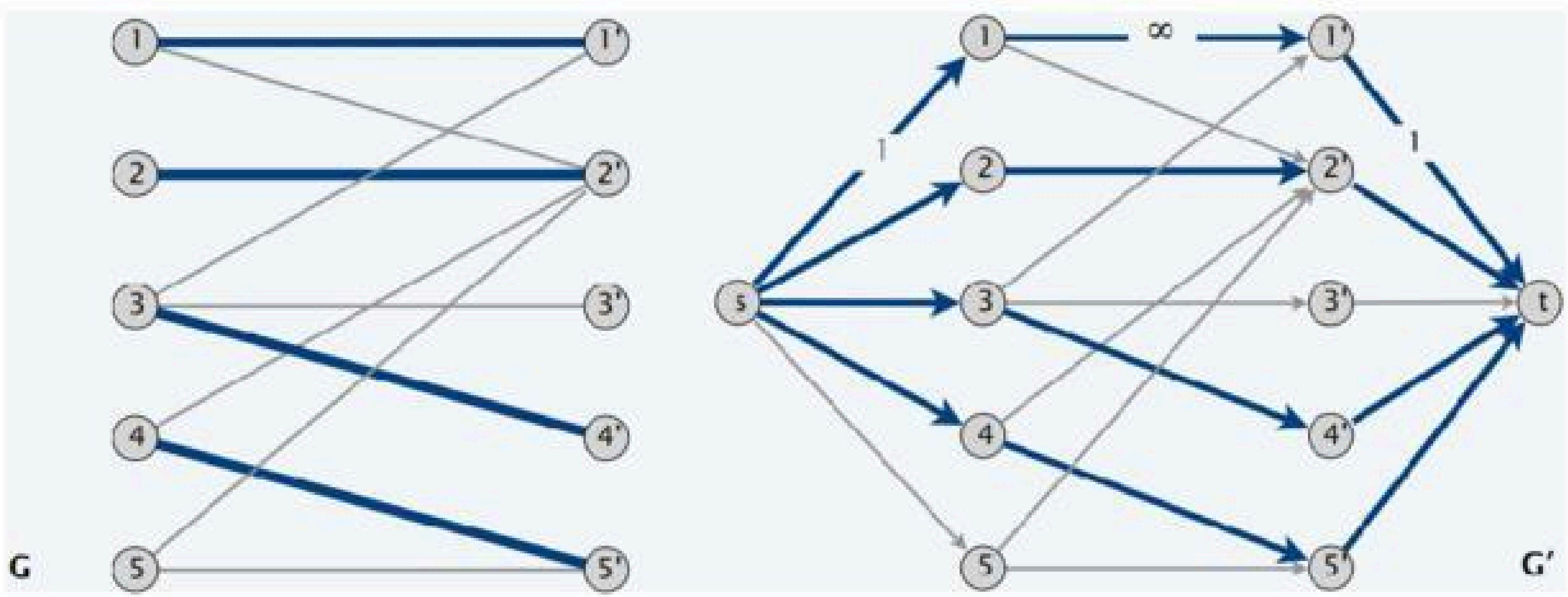
Max matching. Given a graph G, find a max-cardinality matching.



Bipartite matching

Def. A graph G is **bipartite** if the nodes can be partitioned into two subsets L and R such that every edge connects a node in L with a node in R.

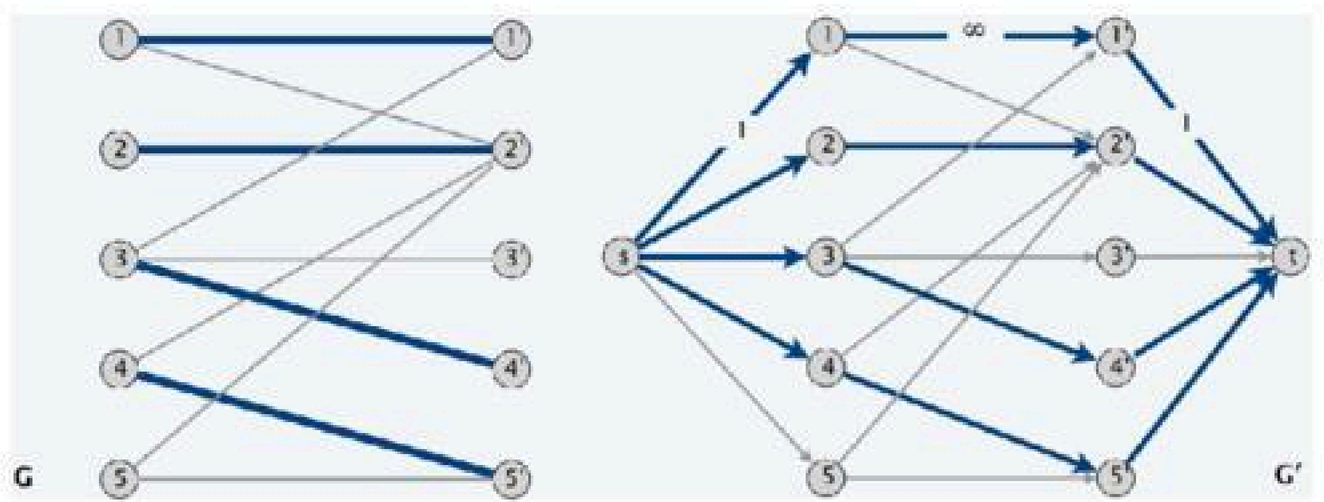
Bipartite matching. Given a bipartite graph $G = (L \cup R, E)$, find a max-cardinality matching.



Bipartite matching: max-flow formulation

Formulation.

- Create digraph $G' = (L \cup R \cup \{s, t\}, E')$.
- Direct all edges from L to R, and assign infinite (or unit) capacity.
- Add unit-capacity edges from s to each node in L.
- Add unit-capacity edges from each node in R to t.

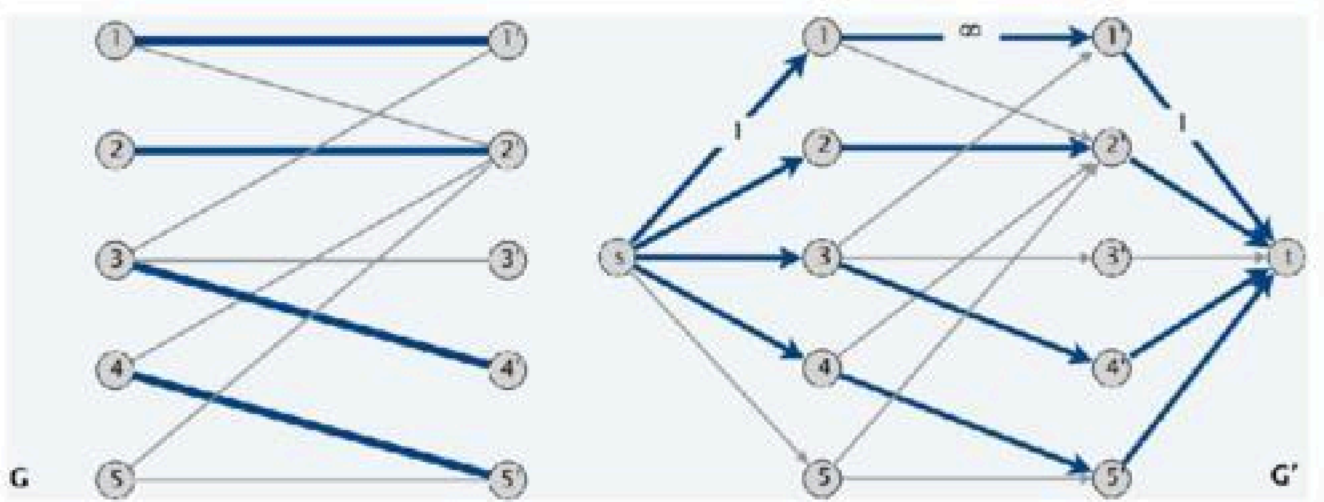


Max-flow formulation: correctness

Theorem. 1-1 correspondence between matchings of cardinality k in G and integral flows of value k in G'.

Pf. $\Rightarrow$

- Let M be a matching in G of cardinality k.
- Consider flow f that sends 1 unit on each of the k corresponding paths.
- f is a flow of value k.

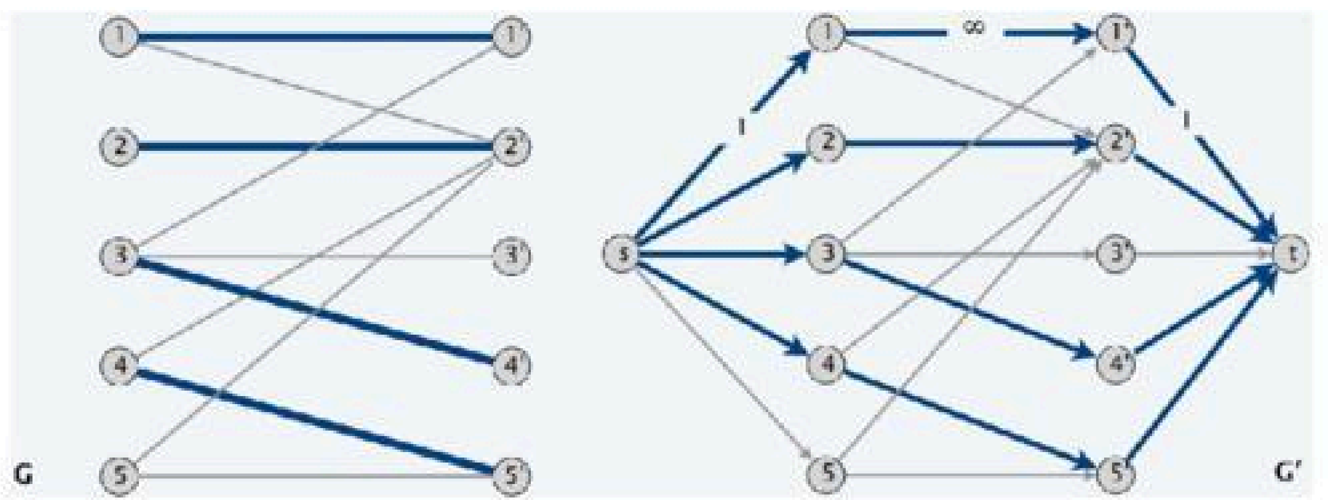


Max-flow formulation: correctness (cont.)

Theorem. 1-1 correspondence between matchings of cardinality k in G and integral flows of value k in G'.

Pf. $\Leftarrow$

- Let f be an integral flow in G' of value k.
- Consider M = set of edges from L to R with $f(e) = 1$.
  - each node in L and R participates in at most one edge in M
  - $|M| = k$: cut $(L \cup \{s\}, R \cup \{t\})$ has k leaving and 0 entering

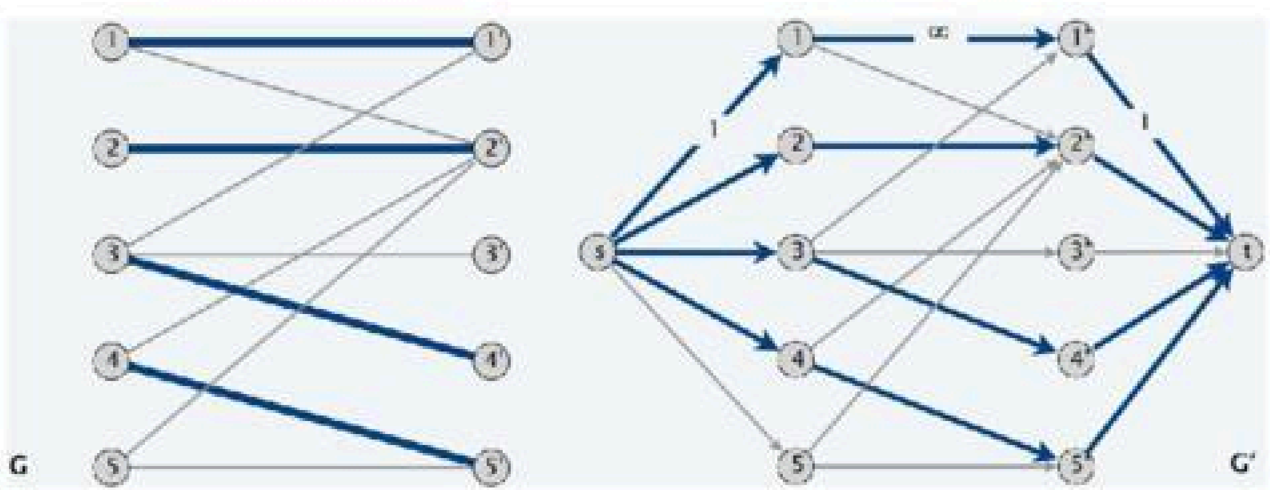


Max-flow for bipartite matching

Theorem. 1-1 correspondence between matchings of cardinality k in G and integral flows of value k in G'.

Corollary. Can solve bipartite matching problem via max-flow formulation.
Pf.

- Integrality theorem $\Rightarrow$ there exists a max flow f^* in G' that is integral.
- 1-1 correspondence $\Rightarrow$ f^* corresponds to max-cardinality matching.



Quiz: bipartite graph via Ford-Fulkerson

What is running time of Ford-Fulkerson algorithms to find a max-cardinality matching in a bipartite graph with $|L| = |R| = n$?

A. $O(m+n)$
B. $O(mn)$
C. $O(mn^2)$
D. $O(m^2n)$

Quiz: bipartite graph via Ford-Fulkerson

What is running time of Ford-Fulkerson algorithms to find a max-cardinality matching in a bipartite graph with $|L| = |R| = n$?

A. $O(m+n)$
B. $O(mn)$
C. $O(mn^2)$
D. $O(m^2n)$

B. $O(mnC)$ and C is a constant now.

Perfect matchings

Def. Given a graph $G = (V, E)$, a subset of edges $M \subseteq E$ is a **perfect matching** if each node appears in exactly one edge in M.

≡

Perfect matchings

Def. Given a graph $G = (V, E)$, a subset of edges $M \subseteq E$ is a **perfect matching** if each node appears in exactly one edge in M.

Q. When does a bipartite graph have a perfect matching?

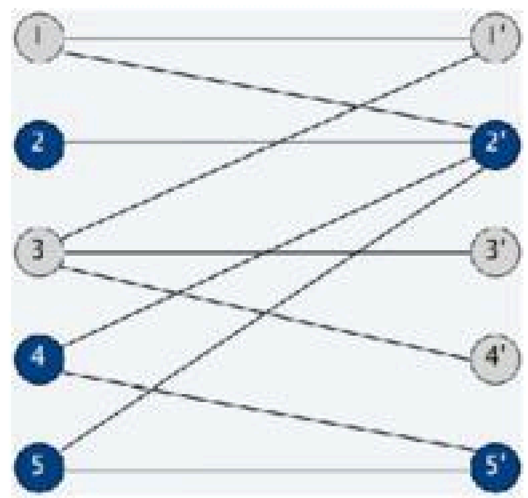
Structure of bipartite graphs with perfect matchings.

- Clearly, we must have $|L| = |R| = n$.
- Which other conditions are necessary?
- Which other conditions are sufficient?

Perfect matchings (cont.)

Notation. Let S be a subset of nodes, and let $N(S)$ be the set of nodes adjacent to nodes in S.

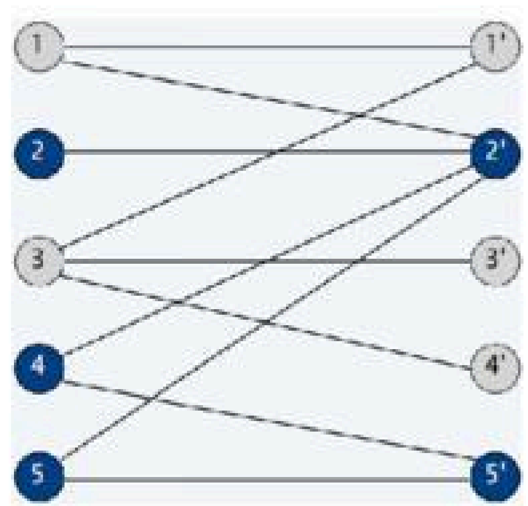
Observation. If a bipartite graph $G = (L \cup R, E)$ has a perfect matching, then $|N(S)| \geq |S|$ for all subsets $S \subseteq L$.
Pf. Each node in S has to be matched to a different node in $N(S)$.



Hall's marriage theorem

Theorem. [Frobenius 1917, Hall 1935] Let $G = (L \cup R, E)$ be a bipartite graph with $|L| = |R|$. Then, graph G has a perfect matching iff $|N(S)| \geq |S|$ for all subsets $S \subseteq L$.

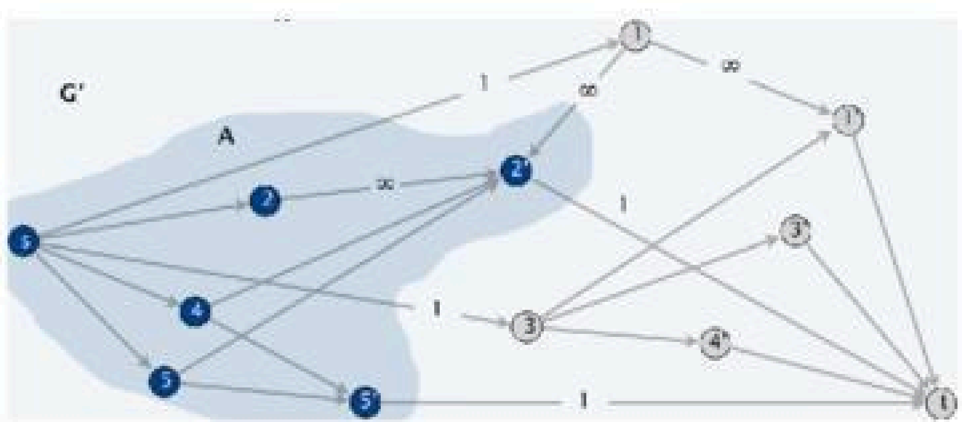
Pf. $\Rightarrow$ This is the previous observation.



Hall's marriage theorem (cont.)

Pf. $\Leftarrow$ Suppose G does not have a perfect matching.

- Formulate as a max-flow problem and let (A, B) be a min cut in G'.
  - By max-flow min-cut theorem, $cap(A, B) < |L|$.
- Define $L_A = L \cap A, L_B = L \cap B, R_A = R \cap A$.
  - $cap(A, B) = |L_B| + |R_A| \Rightarrow |R_A| < |L| - |L_B| = |L_A|$.
  - Min-cut can't use ∞ edges $\Rightarrow N(L_A) \subseteq R_A$.
    - $|N(L_A)| \leq |R_A| < |L_A|$.
- Choose $S = L_A$, contrapositive.



$$L_A = \{2, 4, 5\}$$
$$L_B = \{1, 3\}$$
$$R_A = \{2', 5'\}$$
$$N(L_A) = \{2', 5'\}$$

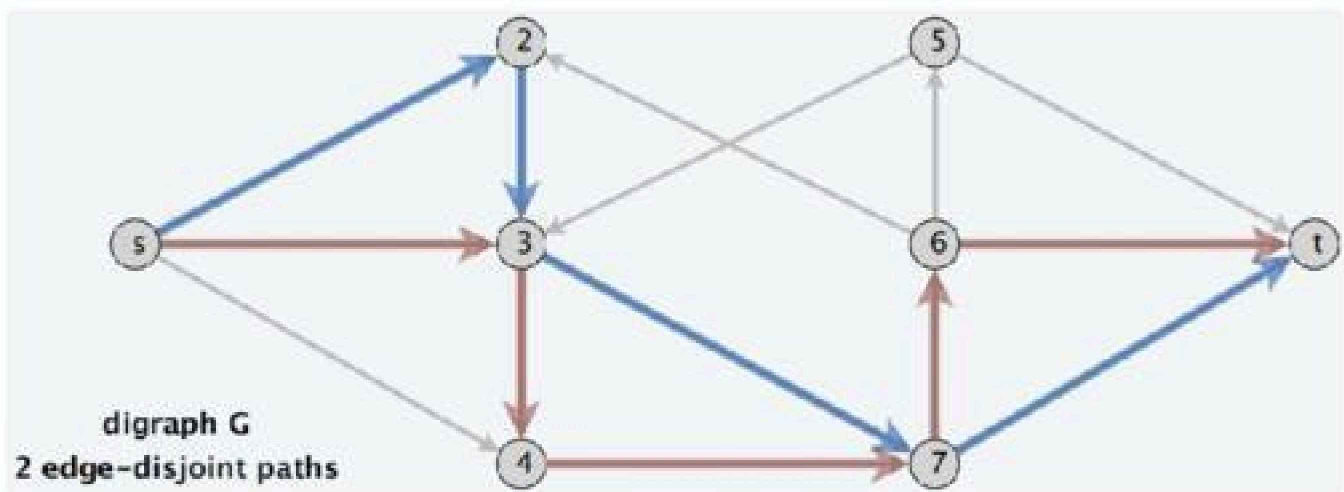
Disjoint paths

Edge-disjoint paths

Def. Two paths are **edge-disjoint** if they have no edge in common.

Edge-disjoint paths problem. Given a digraph $G = (V, E)$ and two nodes s and t, find the max number of edge-disjoint $s \rightsquigarrow t$ paths.

Ex. Communication networks.



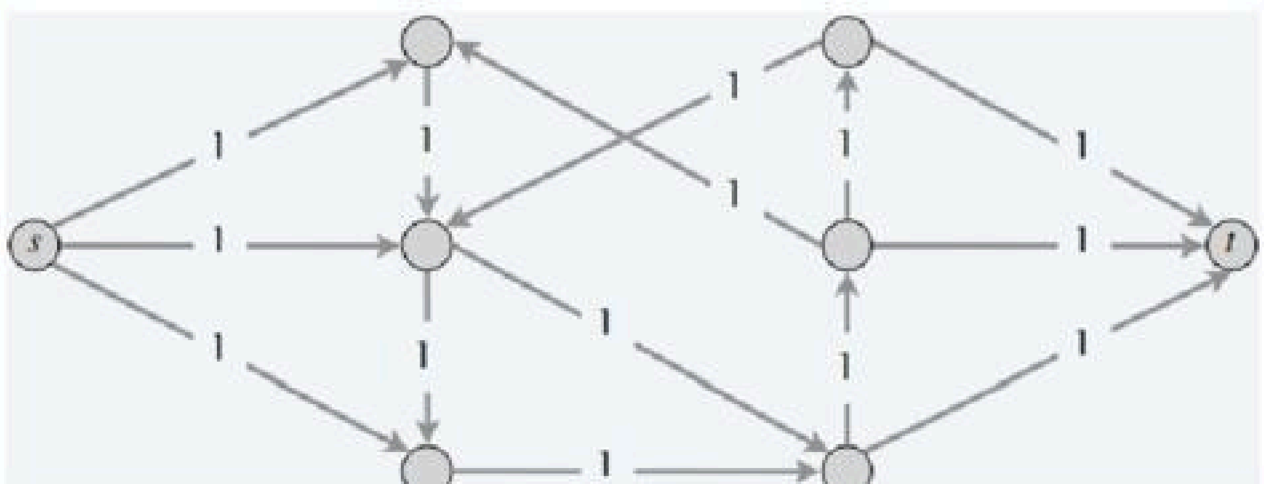
Edge-disjoint: Max-flow

Max-flow formulation. Assign unit capacity to every edge.

Theorem. 1-1 correspondence between k edge-disjoint $s \rightsquigarrow t$ paths in G and integral flows of value k in G'.

Pf. $\Rightarrow$ Let $P_1, \ldots, P_k$ be k edge-disjoint $s \rightsquigarrow t$ paths in G.

- Set $f(e) = 1 : \text{ edge } e \text{ participates in some path}; 0 : \text{ otherwise}.$
- Since paths are edge-disjoint, f is a flow of value k.



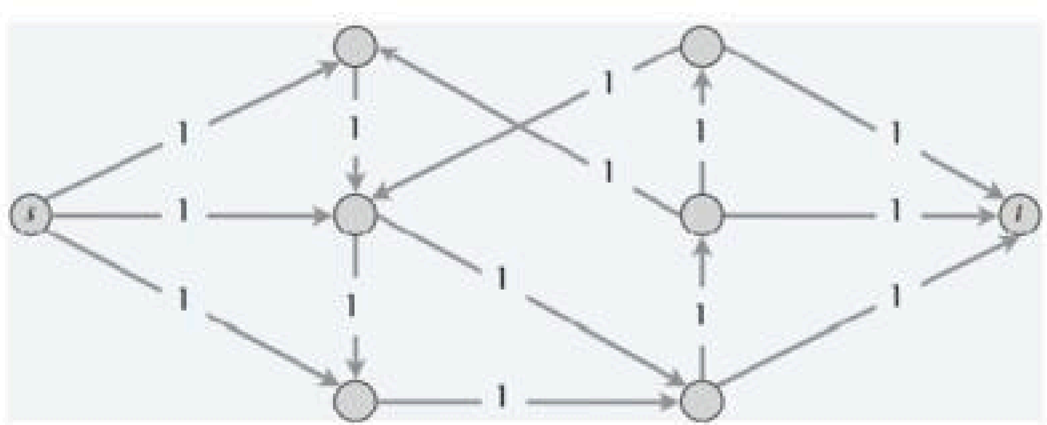
Edge-disjoint: Max-flow (cont.)

Max-flow formulation. Assign unit capacity to every edge.

Theorem. 1-1 correspondence between k edge-disjoint $s \rightsquigarrow t$ paths in G and integral flows of value k in G'.

Pf. $\Leftarrow$ Let f be an integral flow in G' of value k.

- Consider edge (s, u) with $f(s, u) = 1$.
  - by flow conservation, there exists an edge (u, v) with $f(u, v) = 1$
  - continue until reach t, always choosing a new edge
- Produces k (not necessarily simple) edge-disjoint paths.



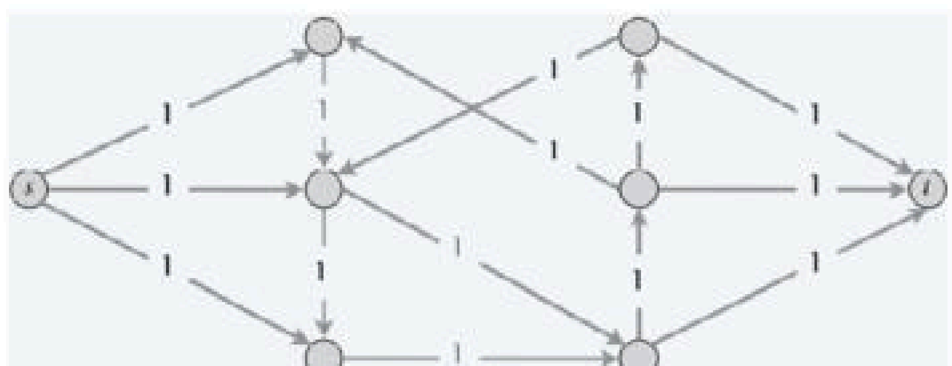
Edge-disjoint: Max-flow solution

Max-flow formulation. Assign unit capacity to every edge.

Theorem. 1-1 correspondence between k edge-disjoint $s \leadsto t$ paths in G and integral flows of value k in G'.

Corollary. Can solve edge-disjoint paths problem via max-flow formulation.
Pf.

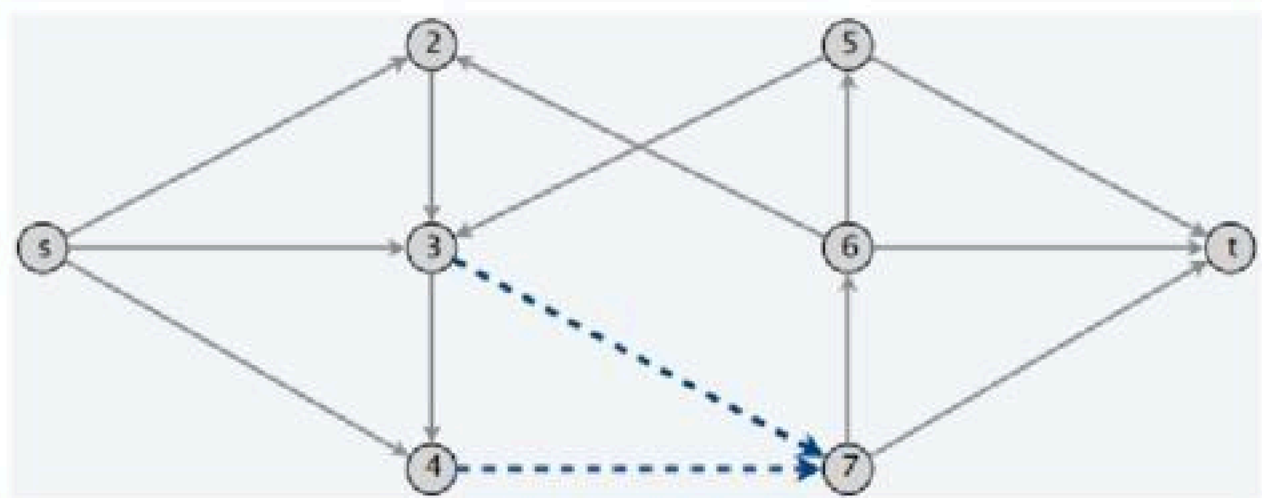
- Integrality theorem $\Rightarrow$ there exists a max flow f^* in G' that is integral.
- 1-1 correspondence $\Rightarrow$ f^* corresponds to max number of edge-disjoint $s \leadsto t$ paths in G.



Network connectivity

Def. A set of edges $F \subseteq E$ **disconnects** t from s if every $s \rightsquigarrow t$ path uses at least one edge in F.

Network connectivity. Given a digraph $G = (V, E)$ and two nodes s and t, find min number of edges whose removal disconnects t from s.

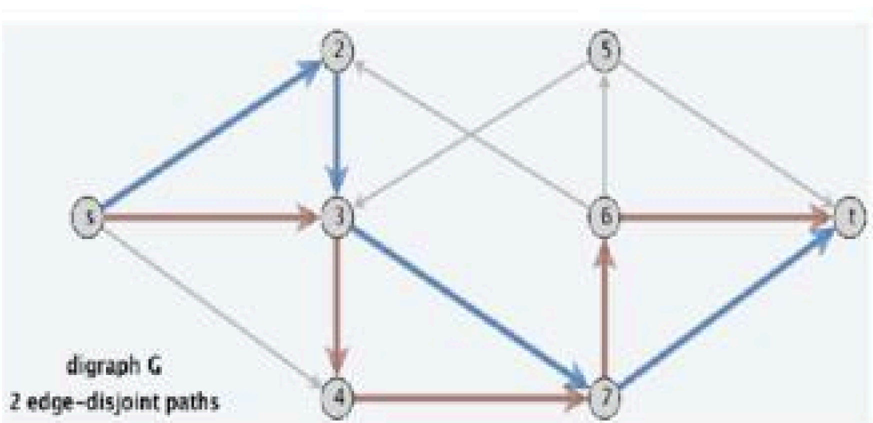


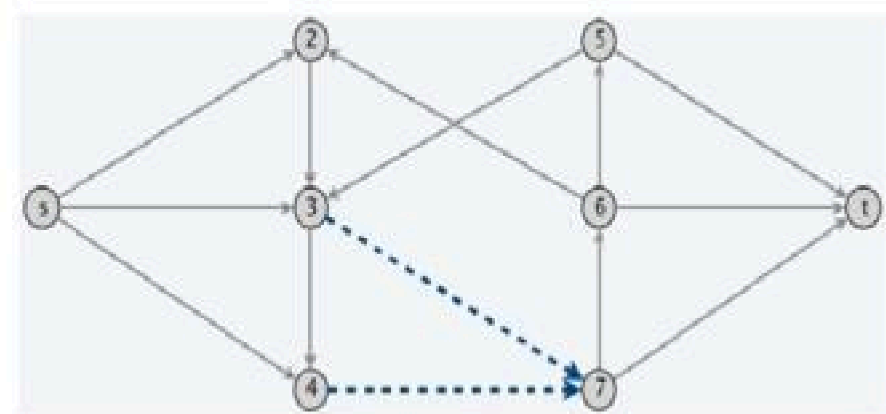
Menger's theorem

Theorem. [Menger 1927] The max number of edge-disjoint $s \rightsquigarrow t$ paths equals the min number of edges whose removal disconnects t from s.

Pf. $\leq$

- Suppose the removal of $F \subseteq E$ disconnects t from s, and $|F| = k$.
- Every $s \rightsquigarrow t$ path uses at least one edge in F.
- Hence, the number of edge-disjoint paths is $\leq k$.



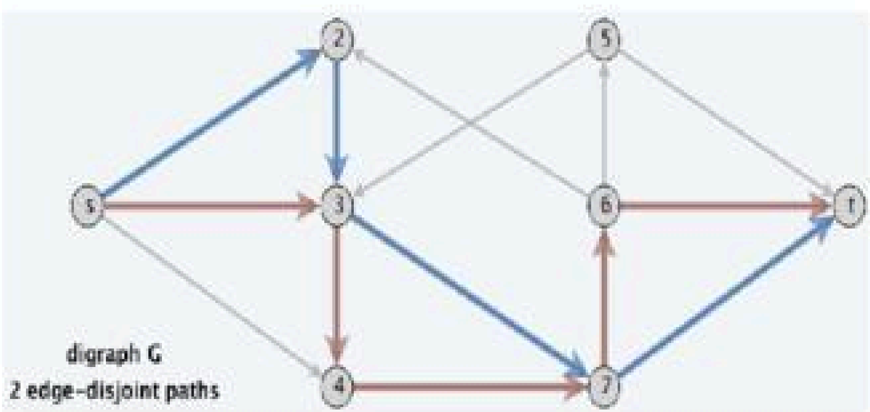


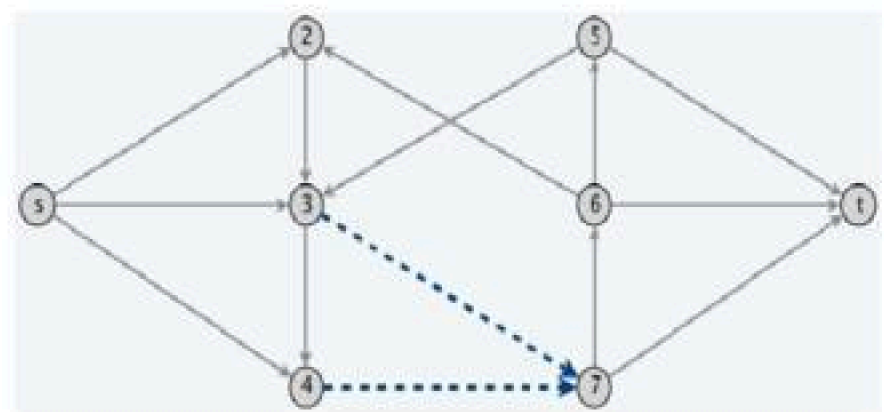
Menger's theorem (cont.)

Theorem. [Menger 1927] The max number of edge-disjoint $s \rightsquigarrow t$ paths equals the min number of edges whose removal disconnects t from s.

Pf. $\geq$

- Suppose max number of edge-disjoint $s \rightsquigarrow t$ paths is k.
- Then value of max flow = k.
- Max-flow min-cut theorem $\Rightarrow$ there exists a cut (A, B) of capacity k.
- Let F be set of edges going from A to B.
- $|F| = k$ and disconnects t from s.





Quiz: edge-disjoint paths

How to find the max number of edge-disjoint paths in an undirected graph?

A. Solve the edge-disjoint paths problem in a digraph (by replacing each undirected edge with two antiparallel edges).
B. Solve a max flow problem in an undirected graph.
C. Both A and B.
D. Neither A nor B.

Quiz: edge-disjoint paths

How to find the max number of edge-disjoint paths in an undirected graph?

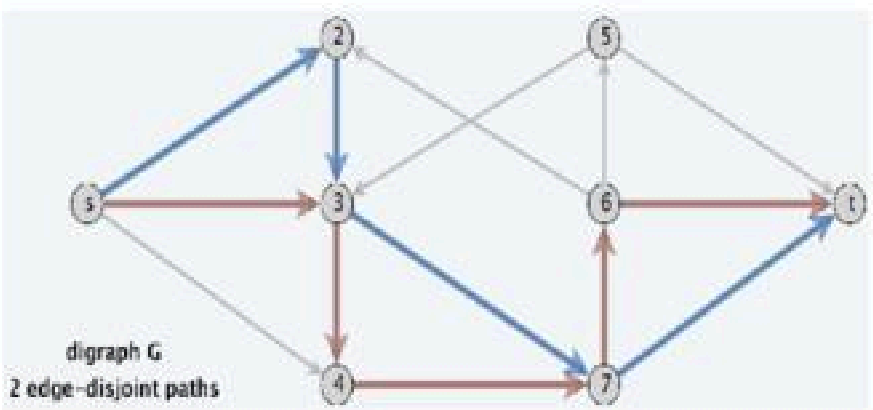
A. Solve the edge-disjoint paths problem in a digraph (by replacing each undirected edge with two antiparallel edges).
B. Solve a max flow problem in an undirected graph.
C. Both A and B.
D. Neither A nor B.

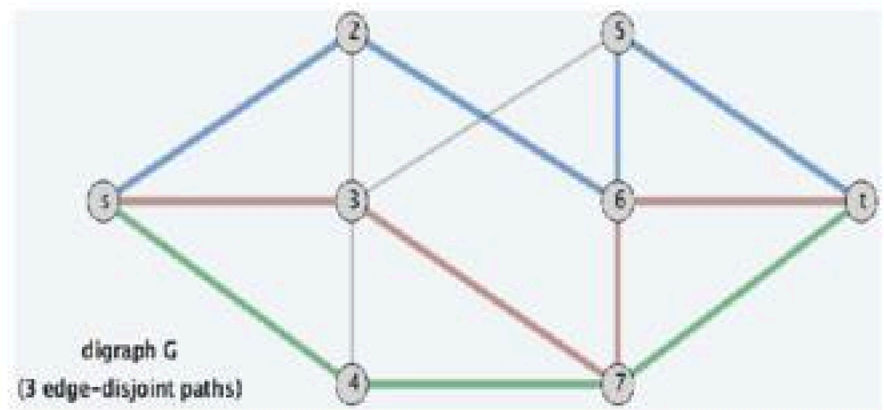
C. both are fine.

Edge-disjoint: undirected graphs

Def. Two paths are **edge-disjoint** if they have no edge in common.

Edge-disjoint paths problem in undirected graphs. Given a graph $G = (V, E)$ and two nodes s and t, find the max number of edge-disjoint $s - t$ paths.



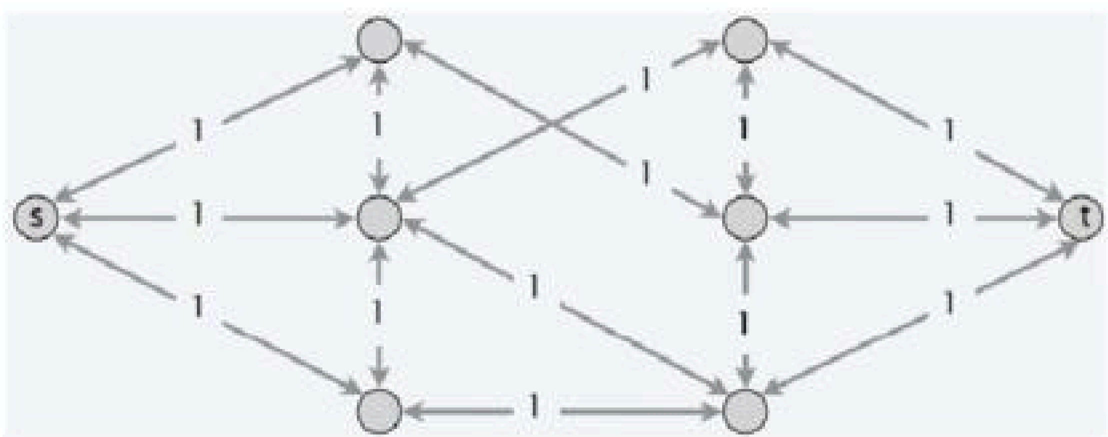


Undirected Edge-disjoint: Max-flow

Max-flow formulation. Replace each edge with two antiparallel edges and assign unit capacity to every edge.

Observation. Two paths P_1 and P_2 may be edge-disjoint in the digraph but not edge-disjoint in the undirected graph.

- if P_1 uses edge (u, v) and P_2 uses its antiparallel edge (v, u)

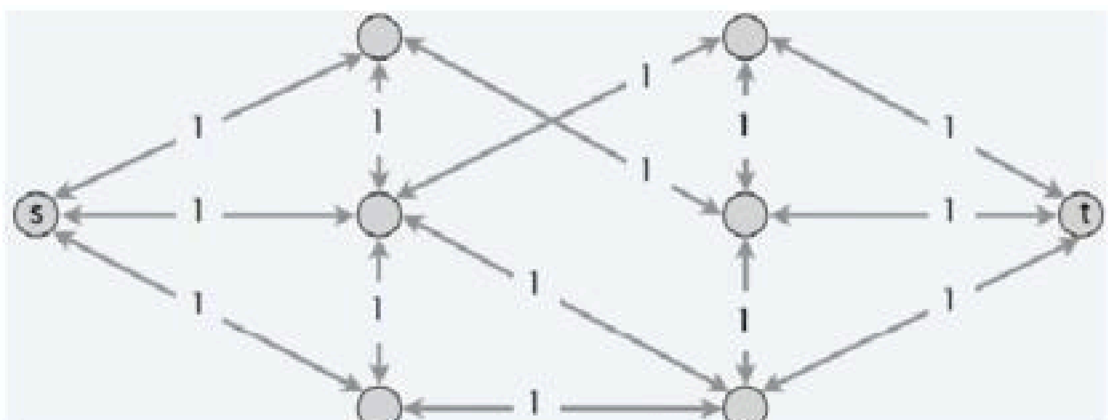


Undirected Menger's theorem

Lemma. In any flow network, there exists a maximum flow f in which for each pair of antiparallel edges e and e': either $f(e) = 0$ or $f(e') = 0$ or both. Moreover, integrality theorem still holds.

Pf. [by induction on number of such pairs]

- Suppose $f(e) > 0$ and $f(e') > 0$ for a pair of antiparallel edges e and e'.
- Set $f(e) = f(e) - \delta$ and $f(e') = f(e') - \delta$, where $\delta = \min\{f(e), f(e')\}$.
  - they cancel each other
- f is still a flow of the same value but has one fewer such pair.



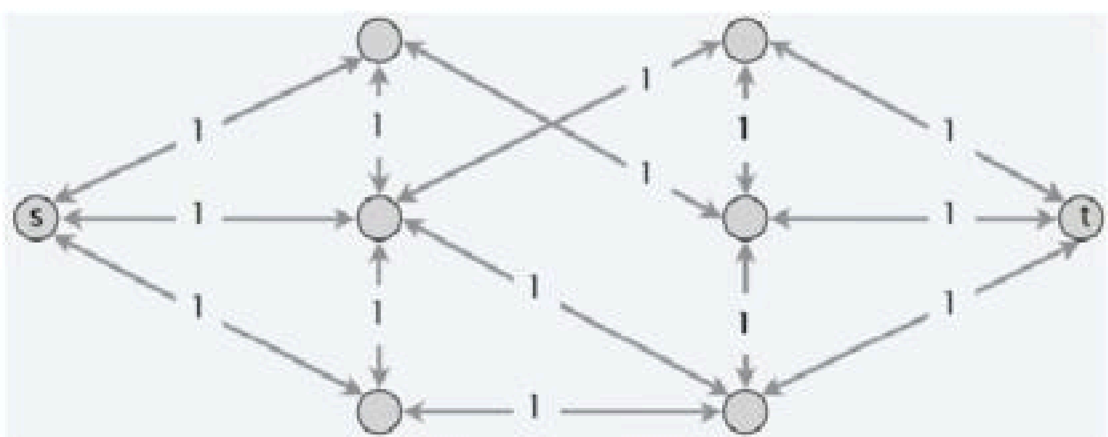
Undirected Menger's theorem (cont.)

Max-flow formulation. Replace each edge with two antiparallel edges and assign unit capacity to every edge.

Lemma. In any flow network, there exists a maximum flow f in which for each pair of antiparallel edges e and e': either $f(e) = 0$ or $f(e') = 0$ or both. Moreover, integrality theorem still holds.

Theorem. Max number of edge-disjoint $s \rightsquigarrow t$ paths = value of max flow.

Pf. Similar to proof in digraphs; use lemma.



More Menger theorems

Theorem. Given an *undirected* graph and two nodes s and t, the max number of *edge-disjoint* s-t paths equals the min number of edges whose removal disconnects s and t.

Theorem. Given an *undirected* graph and two nonadjacent nodes s and t, the max number of internally *node-disjoint* s-t paths equals the min number of internal nodes whose removal disconnects s and t.

Theorem. Given a *directed* graph with two nonadjacent nodes s and t, the max number of internally *node-disjoint* $s \rightsquigarrow t$ paths equals the min number of internal nodes whose removal disconnects t from s.

Extensions to max flow

Quiz: Extensions to max flow

Which extensions to max flow can be easily modeled?

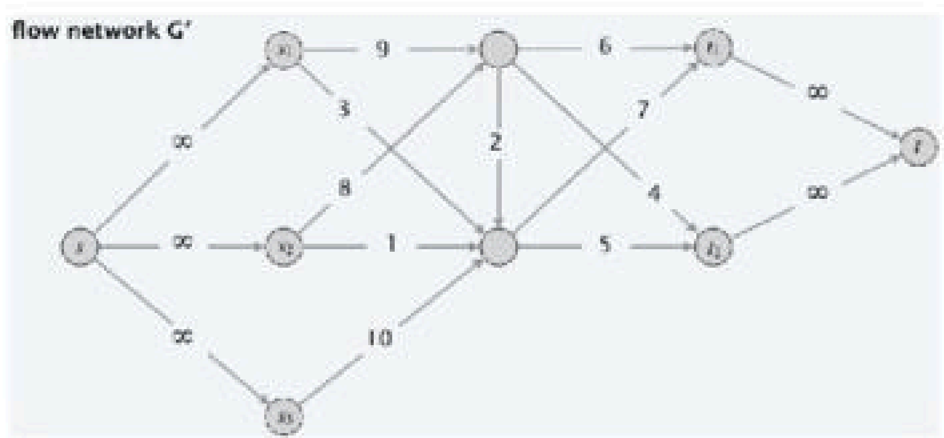
A. Multiple sources and multiple sinks.
B. Undirected graphs.
C. Lower bounds on edge flows.
D. All of the above.

Multiple sources & sinks

Def. Given a digraph $G = (V, E)$ with edge capacities $c(e) \geq 0$ and multiple source nodes and multiple sink nodes, find max flow that can be sent from the source nodes to the sink nodes.

Max-flow formulation.

- Add a new source node s and sink node t.
- For each original source node s_i add edge (s, s_i) with capacity ∞.
- For each original sink node t_i, add edge (t_i, t) with capacity ∞.



≡

Claim. 1-1 correspondence between flows in G and G'.

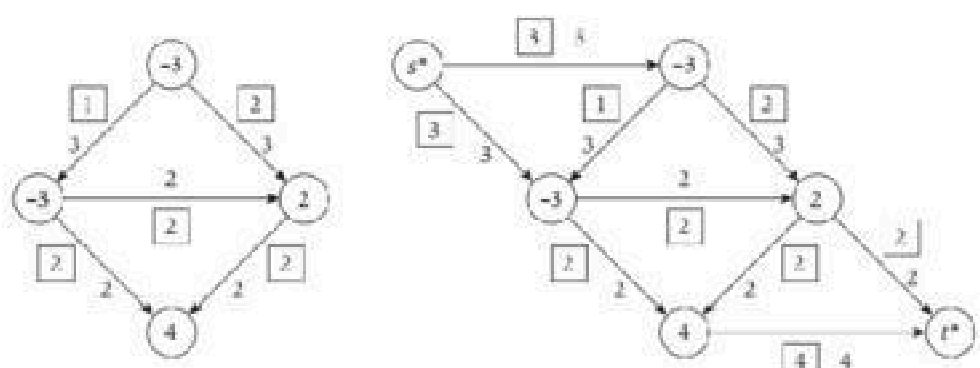
Circulation w/ supplies & demands

Def. Given a digraph $G = (V, E)$ with edge capacities $c(e) \geq 0$ and node demands $d(v)$, a **circulation** is a function $f(e)$ that satisfies:

- [capacity] For each $e \in E : 0 \leq f(e) \leq c(e)$
- [conservation] For each $v \in V : \sum_{e \text{ into } v} f(e) - \sum_{e \text{ out } v} f(e) = d(v)$

Max-flow formulation.

- Add new source s and sink t.
- For each v with $d(v) < 0$, add edge (s, v) with capacity $-d(v)$.
- For each v with $d(v) > 0$, add edge (v, t) with capacity $d(v)$.



Claim. G has circulation iff G' has max flow of value $D = \sum_{v:d(v)>0} d(v) = \sum_{v:d(v)<0} -d(v)$

- ie., saturates all edges leaving s and entering t

Circulation w/ S & D

Integrality theorem. If all capacities and demands are integers, and there exists a circulation, then there exists one that is integer-valued.
Pf. Follows from max-flow formulation + integrality theorem for max flow.

Theorem. Given (V, E, c, d), there does *not* exist a circulation iff there exists a node partition (A, B) such that $\sum_{v \in B} d(v) > cap(A, B)$.

- ie., demand by nodes in B exceeds supply of nodes in B plus max capacity of edges going from A to B

Pf sketch. Look at min cut in G'.

Circulation w/ S & D & lower bounds

Def. Given a digraph $G = (V, E)$ with edge capacities $c(e) \geq 0$, lower bounds $l(e) \geq 0$, and node demands $d(v)$, a circulation $f(e)$ is a function that satisfies:

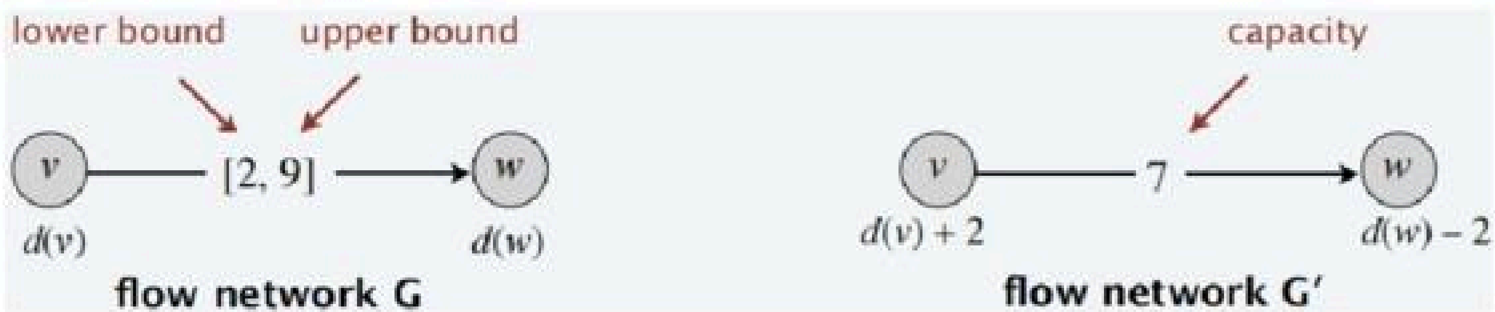
- [capacity] For each $e \in E : l(e) \leq f(e) \leq c(e)$
- [conservation] For each $v \in V : \sum_{e \text{ into } v} f(e) - \sum_{e \text{ out } v} f(e) = d(v)$

Circulation problem with lower bounds. Given (V, E, l, c, d), does there exist a feasible circulation?

Circulation w/ S & D & LB

Max-flow formulation. Model lower bounds as circulation with demands.

- Send $l(e)$ units of flow along edge e.
- Update demands of both endpoints.



Theorem. There exists a circulation in G iff there exists a circulation in G'. Moreover, if all demands, capacities, and lower bounds in G are integers, then there exists a circulation in G that is integer-valued.

Pf sketch. $f(e)$ is a circulation in G iff $f'(e) = f(e) - l(e)$ is a circulation in G'.

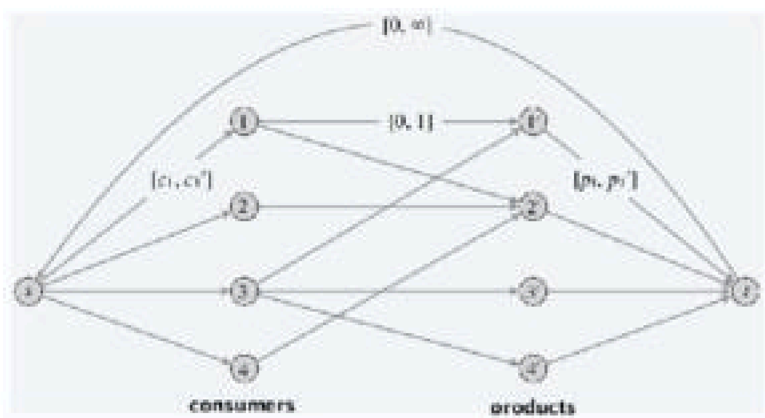
Survey design

Survey Design Problem

Goal. Design a survey that meets following specs, if possible.

- Design survey asking n_1 consumers about n_2 products.
- Can survey consumer i about product j only if they own it.
- Ask consumer i between c_i and c_i' questions.
- Ask between p_j and p_j' consumers about product j.

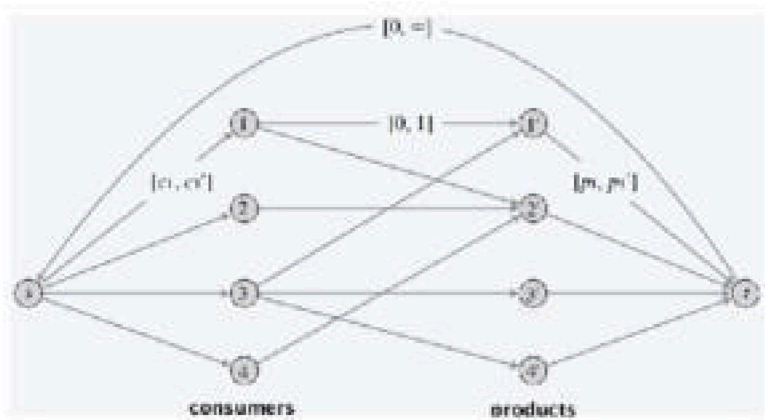
Bipartite perfect matching. Special case when $c_i = c_i' = p_j = p_j' = 1$.



Survey Design: Max-flow

Max-flow formulation. Model as a circulation problem with lower bounds.

- Add edge (i, j) if consumer j owns product i.
- Add edge from s to consumer j.
- Add edge from product i to t.
- Add edge from t to s.
- All demands = 0.
- Integer circulation ⇔ feasible survey design.



Airline scheduling

Airline Scheduling Problem

Airline scheduling.

- Complex computational problem faced by airline carriers.
- Must produce schedules that are efficient in terms of equipment usage, crew allocation, and customer satisfaction.
  - even in presence of unpredictable events, such as weather and breakdowns
- One of largest consumers of high-powered algorithmic techniques.

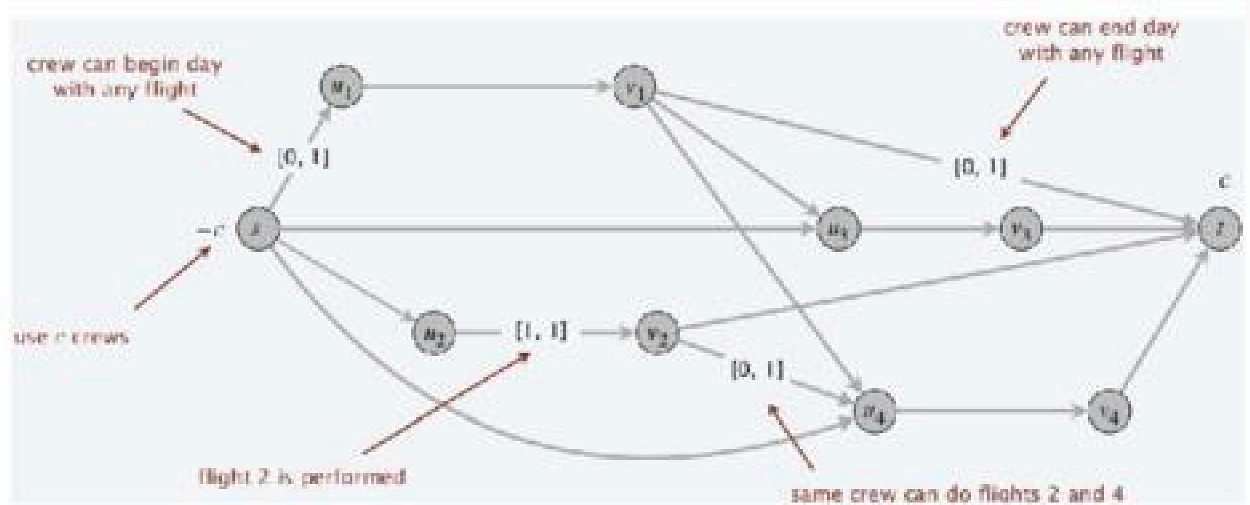
"Toy problem".

- Manage flight crews by reusing them over multiple flights.
- Input: set of k flights for a given day.
- Flight i leaves origin o_i at time s_i and arrives at destination d_i at time f_i.
- Minimize number of flight crews.

Airline Scheduling: Circulation

Circulation formulation. [to see if c crews suffice]

- For each flight i, include two nodes u_i and v_i.
- Add source s with demand $-c$, and edges (s, u_i) with capacity 1.
- Add sink t with demand c, and edges (v_i, t) with capacity 1.
- For each i, add edge (u_i, v_i) with lower bound and capacity 1.
- if flight j reachable from i, add edge (v_i, uj) with capacity 1.



Airline Scheduling: analysis

Theorem. The airline scheduling problem can be solved in $O(k^3 \log k)$ time.
Pf.

- k = number of flights.
- c = number of crews (unknown).
- $O(k)$ nodes, $O(k^2)$ edges.
- At most k crews needed.
  - $\Rightarrow$ solve $log_2 k$ circulation problems.
    - binary search for min value c^*
- Value of any flow is between 0 and k.
  - $\Rightarrow$ at most k augmentations per circulation problem.
- Overall time = $O(k^3 \log k)$.

Airline Scheduling: analysis

Theorem. The airline scheduling problem can be solved in $O(k^3 \log k)$ time.

Pf.

- k = number of flights.
- c = number of crews (unknown).
- $O(k)$ nodes, $O(k^2)$ edges.
- At most k crews needed.
  - $\Rightarrow$ solve $log_2 k$ circulation problems.
    - binary search for min value c^*
- Value of any flow is between 0 and k.
  - $\Rightarrow$ at most k augmentations per circulation problem.
- Overall time = $O(k^3 \log k)$.

Remark. Can solve in $O(k^3)$ time by formulating as *minimum-flow* problem.

Airline Scheduling: practical discussion

Remark. We solved a toy version of a real problem.

Real-world problem models countless other factors:

- Union regulations: e.g., flight crews can fly only a certain number of hours in a given time window.
- Need optimal schedule over planning horizon, not just one day.
- Approaching deadhead has a cost.
- Flights don't always leave or arrive on schedule.
- Simultaneously optimize both flight schedule and fare structure.

Message.

- Our solution is a generally useful technique for efficient reuse of limited resources but trivializes real airline scheduling problem.
- Flow techniques useful for solving airline scheduling problems (and are widely used in practice).
- Running an airline efficiently is a very difficult problem.

Image segmentation

Image Segmentation Problem

Image segmentation.

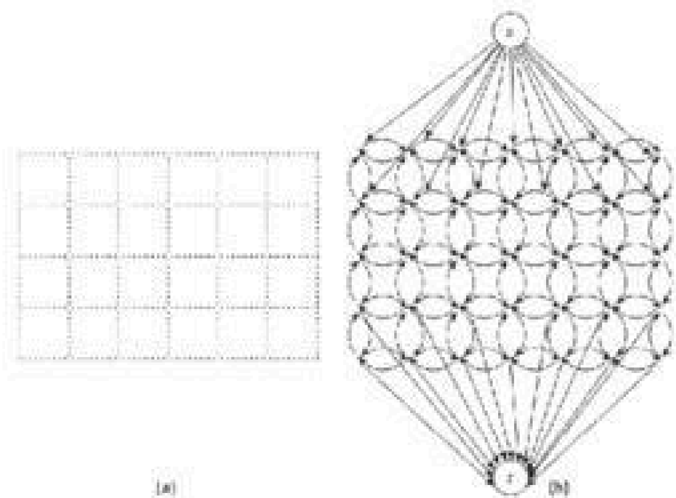
- Divide image into coherent regions.
- Central problem in image processing.

Ex. Separate human from background and reconstruct a new scene.

FG/BG segmentation

Foreground / background segmentation.

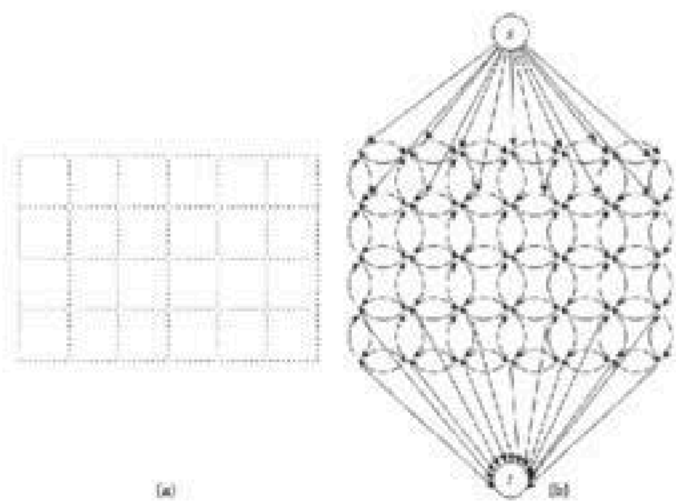
- Label each pixel as belonging to foreground or background.
- V = set of pixels, E = pairs of neighboring pixels.
  - $a_i \geq 0$ is likelihood pixel i in foreground.
  - $b_j \geq 0$ is likelihood pixel i in background.
  - $p_{ij} \geq 0$ is separation penalty for labeling one of neighboring i and j as foreground, and the other as background.



FG/BG segmentation: goals

- Accuracy: if $a_i > b_j$ in isolation, prefer to label i in foreground.
- Smoothness: if many neighbors of i are labeled foreground, we should be inclined to label i as foreground.
- Find partition (A, B) that maximizes:

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



FG/BG segmentation: min-cut?

Formulate as min-cut problem. Issues:

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

FG/BG segmentation: min-cut?

Formulate as min-cut problem. Issues:

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

Turn into minimization problem.

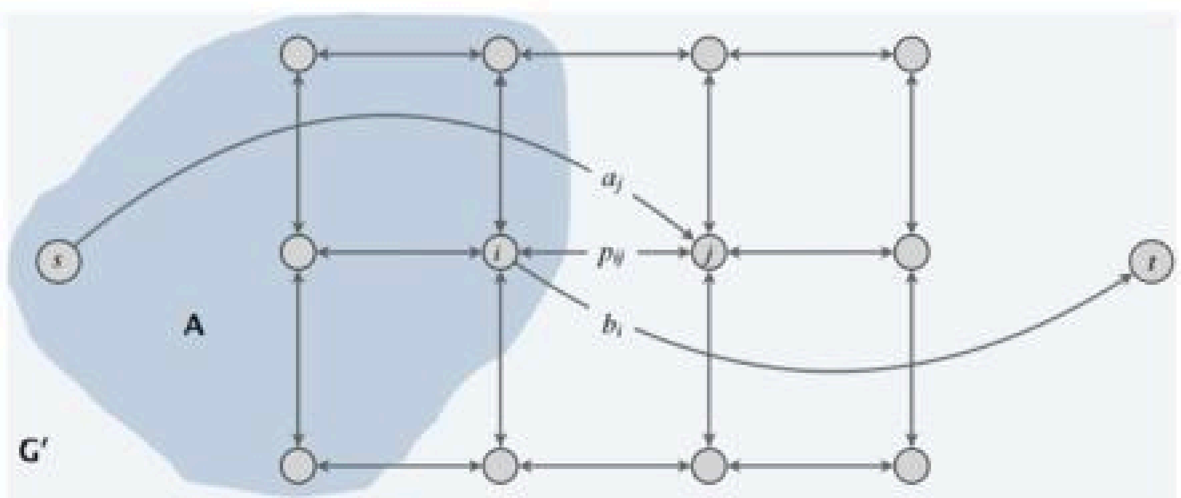
- Maximizing: $\sum_{i \in A} a_i + \sum_{j \in B} b_j - \sum_{(i,j) \in E, |A \cap \{i,j\}|=1} p_{ij}$
  - is equivalent to minimizing: $(\sum_{i \in V} a_i + \sum_{j \in V} b_j) - (\sum_{i \in A} a_i + \sum_{j \in B} b_j - \sum_{(i,j) \in E, |A \cap \{i,j\}|=1} p_{ij})$
  - or alternatively:

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

FG/BG segmentation: min-cut

Formulate as min-cut problem $G' = (V', E')$.

- Include node for each pixel.
- Use two antiparallel edges instead of undirected edge.
- Add source s to correspond to foreground.
- Add sink t to correspond to background.



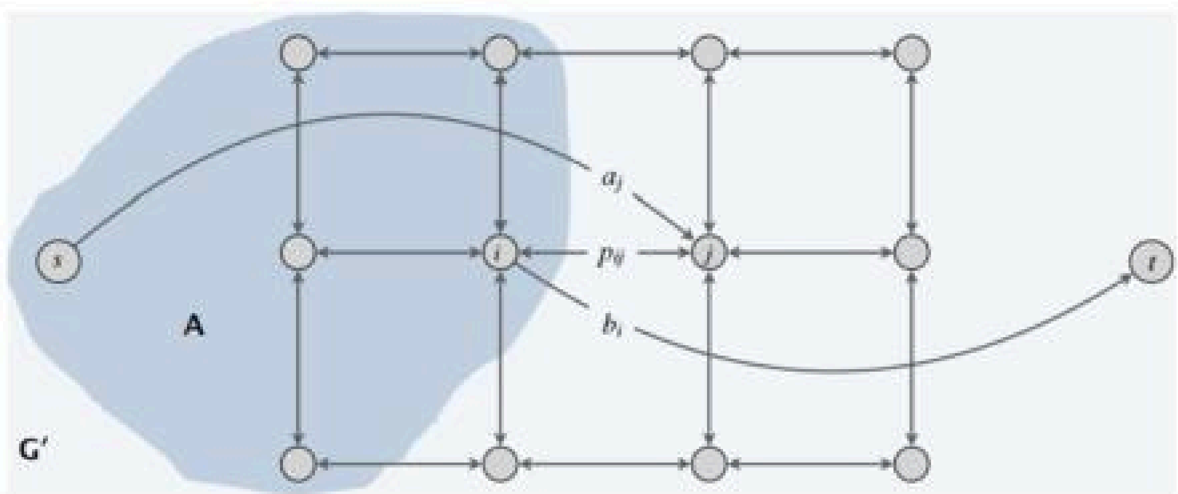
FG/BG segmentation: min-cut (cont.)

Consider min-cut (A, B) in G'.

- A = foreground.

$$cap(A,B) = \sum_{j \in B} a_j + \sum_{i \in A} b_i + \sum_{(i,j) \in E, i \in A, j \in B} p_{ij}$$

- Precisely the quantity we want to minimize.



Grabcut image segmentation

Grabcut. [Rother-Kolmogorov-Blake 2004]

"GrabCut" — Interactive Foreground Extraction using Iterated Graph Cuts

Carsten Rother* Vladimir Kolmogorov† Andrew Blake‡
Microsoft Research Cambridge, UK

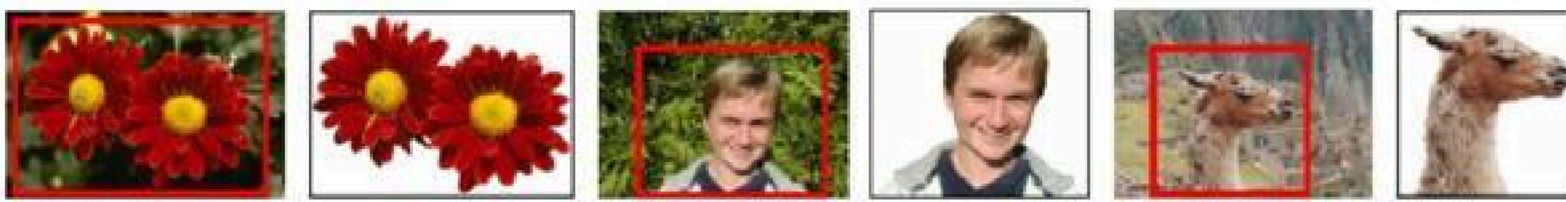


Figure 1: **Three examples of GrabCut .** The user drags a rectangle loosely around an object. The object is then extracted automatically.

- integrated in PowerPoint.

Project selection

Project Selection Problem

Projects with prerequisites.

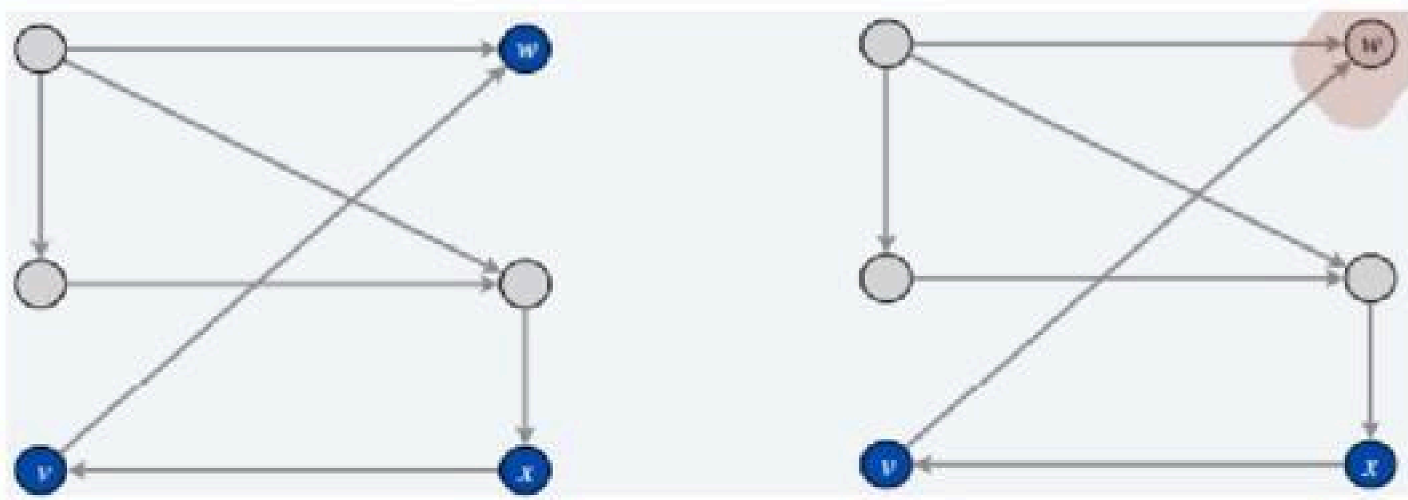
- Set of possible projects P: project v has associated revenue p_v.
  - value can be positive or negative
- Set of prerequisites E: $(v, w) \in E$ means w is a prerequisite for v.
- A subset of projects $A \subseteq P$ is **feasible** if the prerequisite of every project in A also belongs to A.

Project selection problem. Given a set of projects P and prerequisites E, choose a feasible subset of projects to maximize revenue.

- aka. Maximum Weight Closure Problem

Project selection: prerequisite graph

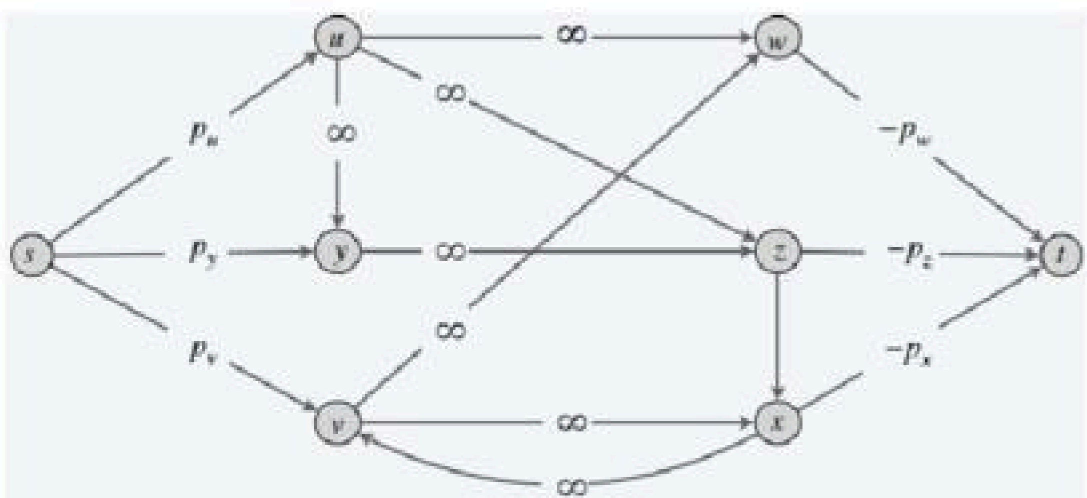
Prerequisite graph. Add edge (v, w) if w is a prerequisite for v.



Project selection: min-cut

Min-cut formulation.

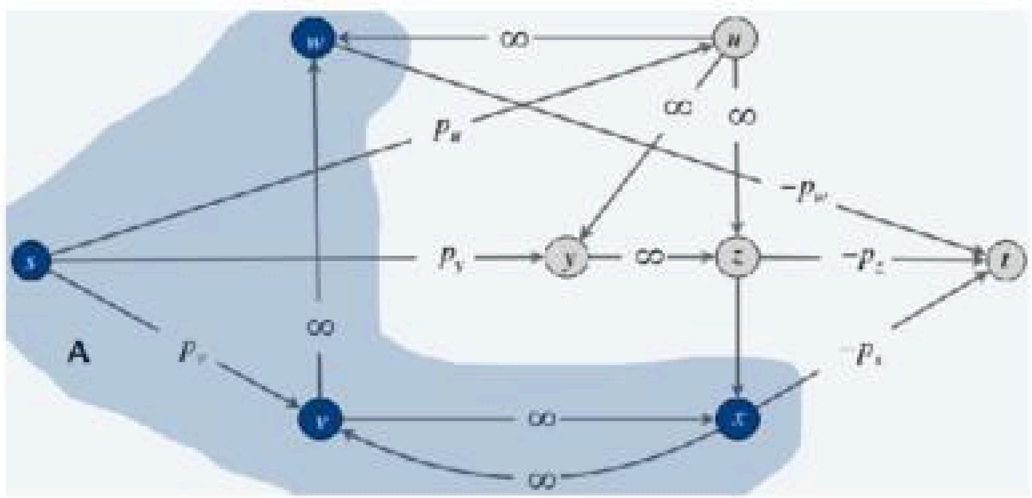
- Assign a capacity of ∞ to each prerequisite edge.
- Add edge (s, v) with capacity p_v if $p_v > 0$.
- Add edge (v, t) with capacity $-p_v$ if $p_v < 0$.
- For notational convenience, define $p_s = p_t = 0$.



Project selection: min-cut (cont.)

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

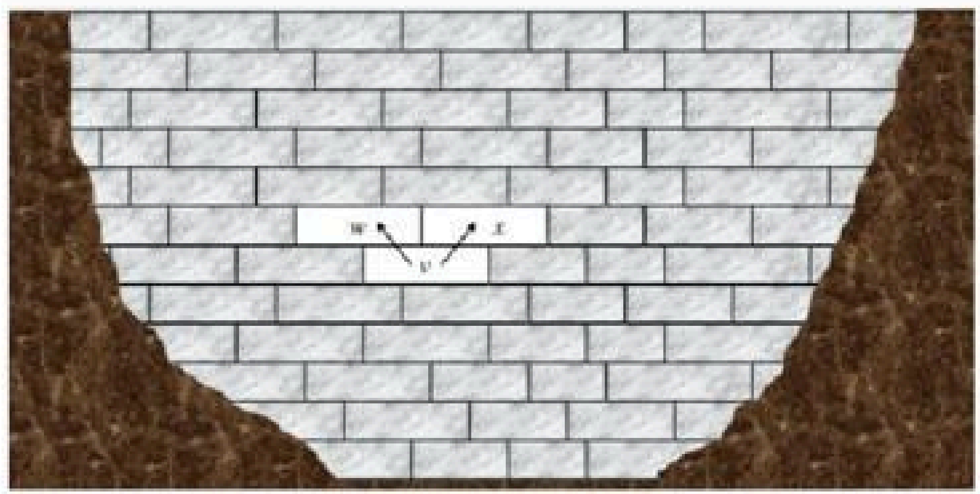
- Infinite capacity edges ensure $A - \{s\}$ is feasible.
  - cut never cross ∞: prerequisite must go together.
- Max revenue because:
  - $cap(A, B) = \sum_{v \in B: p_v > 0} p_v + \sum_{v \in A: p_v < 0} (-p_v)$



Open-pit mining

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 until both blocks w and x are removed.



Tournament elimination

Tournament: who is eliminated?

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

T	W	L	P	A	B	C	D
A	83	71	8	-	1	6	1
B	80	79	3	1	-	0	2
C	78	78	6	6	0	-	0
D	77	82	3	1	2	0	-

Tournament: who is eliminated?

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

T	W	L	P	A	B	C	D
A	83	71	8	-	1	6	1
B	80	79	3	1	-	0	2
C	78	78	6	6	0	-	0
D	77	82	3	1	2	0	-

D is mathematically eliminated.

- D finishes with ≤ 80 wins.
- A already has 83 wins.

Remark. This appear to be the only reasoning sports writers aware of.

Tournament: who is eliminated?

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

T	Win	Lose	to Play	A	B	C	D
A	83	71	8	-	1	6	1
B	80	79	3	1	-	0	2
C	78	78	6	6	0	-	0
D	77	82	3	1	2	0	-

B is mathematically eliminated.

- B finishes with ≤ 83 wins.
- Either C or A will finish with ≥ 84 wins.

Tournament: who is eliminated?

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

T	Win	Lose	to Play	A	B	C	D
A	83	71	8	-	1	6	1
B	80	79	3	1	-	0	2
C	78	78	6	6	0	-	0
D	77	82	3	1	2	0	-

B is mathematically eliminated.

- B finishes with ≤ 83 wins.
- Either C or A will finish with ≥ 84 wins.

Observation. Answer depends not only on how many games already won and left to play, but on whom they're against.

Tournament Elimination Problem

Current standings.

- Set of teams S.
- Distinguished team $z \in S$.
- Team x has won w_x games already.
- Teams x and y play each other r_{xy} additional times.

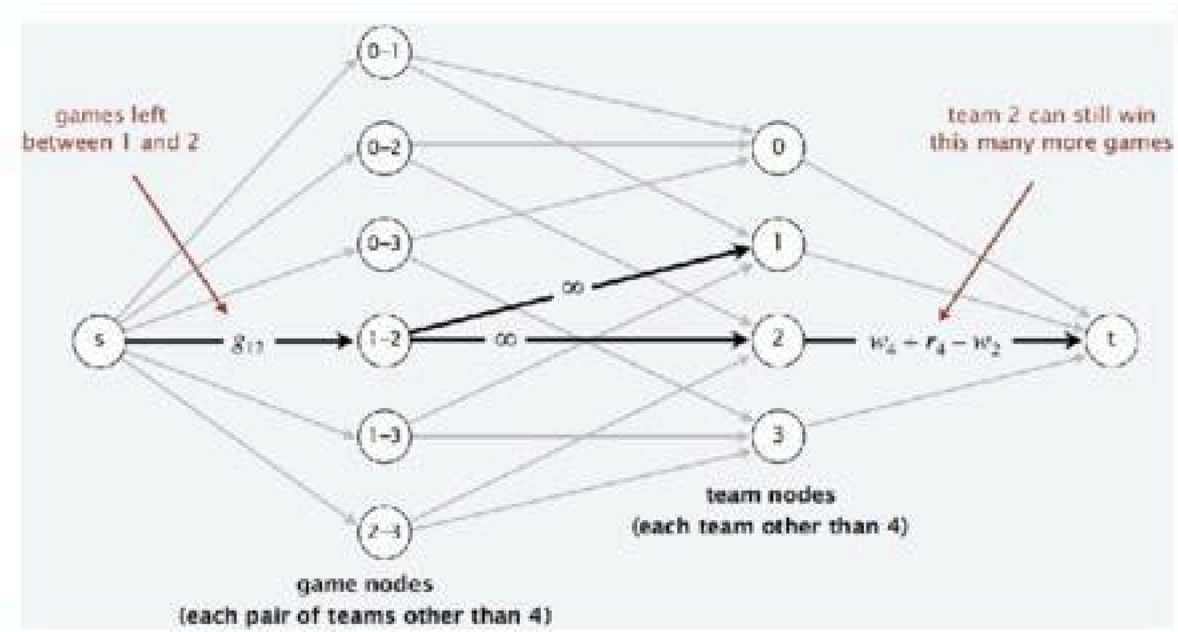
Tournament elimination problem. Given the current standings, is there any outcome of the remaining games in which team z finishes with the most (or tied for the most) wins?

- [Schwartz 1966] Possible winners in partially completed tournaments

Tournament Elimination: max-flow

Can team 4 finish with most wins?

- Assume team 4 wins all remaining games $\Rightarrow w_4 + r_4$ wins.
- Arrange remaining games so that all teams have $\leq w_4 + r_4$ wins.

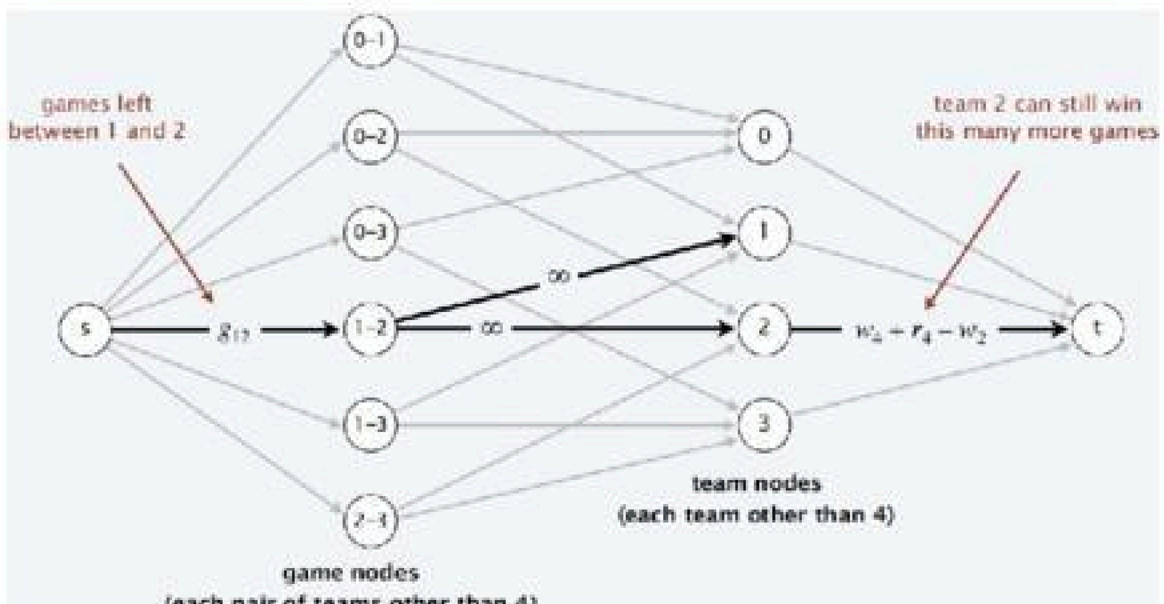


Tournament Elimination: max-flow (cont.)

Theorem. Team 4 not eliminated iff max flow saturates all edges leaving s.

Pf.

- Integrality theorem $\Rightarrow$ each remaining game between x and y added to number of wins for team x or team y.
- Capacity on (x, t) edges ensure no team wins too many games.



An explanation for sports writers

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

T	Win	Lose	to Play	A	B	C	D	E
A	75	59	28	-	3	8	7	3
B	71	63	28	3	-	2	7	4
C	69	66	27	8	2	-	0	0
D	63	72	27	7	7	0	-	0
E	49	86	27	3	4	0	0	-

An explanation for sports writers

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

T	Win	Lose	to Play	A	B	C	D	E
A	75	59	28	-	3	8	7	3
B	71	63	28	3	-	2	7	4
C	69	66	27	8	2	-	0	0
D	63	72	27	7	7	0	-	0
E	49	86	27	3	4	0	0	-

E is mathematically eliminated.

- E finishes with $\leq 49 + 86 = 76$ wins.
- Wins for $R = \{A, B, C, D\} = 75 + 71 + 69 + 63 = 278$.
- Remaining games among $\{A, B, C, D\} = 3 + 8 + 7 + 2 + 7 = 27$.
- Average team in R wins $305/4 = 76.25$ games.

Certificate of elimination

Theorem. [Hoffman-Rivlin 1967] Team z is eliminated iff there exists a subset T^*: $w_z + g_z < \frac{w(T^*)+g(T^*)}{|T^*|}$.

- # wins: $w(T) = \sum_{i \in T} w_i$; # remaining: $g(T) = \sum_{\{x,y\} \subseteq T} g_{xy}$

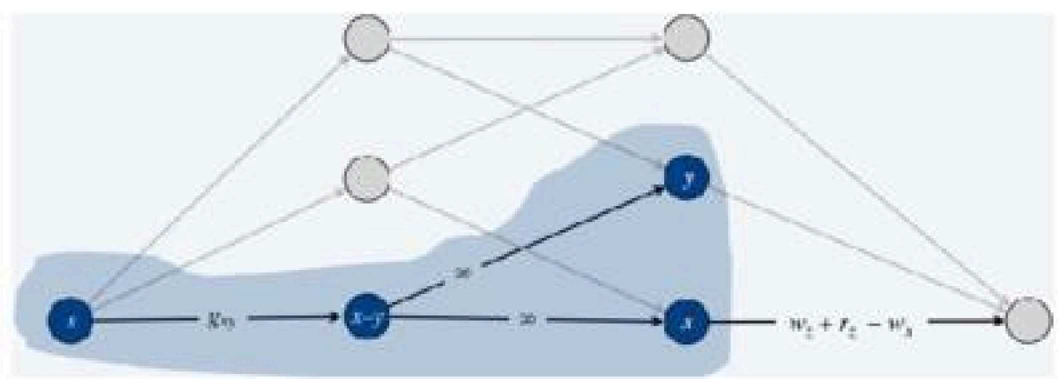
Pf. $\Leftarrow$

- Suppose there exists $T^* \subseteq S$ satisfy certificate.
- Then, teams in T^* win at least $(w(T^*) + g(T^*))/|T^*|$ games on average.
- This exceeds maximum number that team z can win.

Certificate of elimination (cont.)

Pf. $\Rightarrow$

- Use max-flow formulation, and consider min cut (A, B).
- Let T^* = team nodes on source side A of min cut.
- Observe that game node $x\text{-}y \in A$ iff both $x \in T^*$ and $y \in T^*$.
  - infinite capacity ensure $x\text{-}y \in A$, then both $x \in A$ and $y \in A$
  - if $x \in A$ and $y \in A$ but $x\text{-}y \notin A$, then adding $x\text{-}y$ to A decreases the capacity of the cut by g_{xy}



Certificate of elimination (cont.)

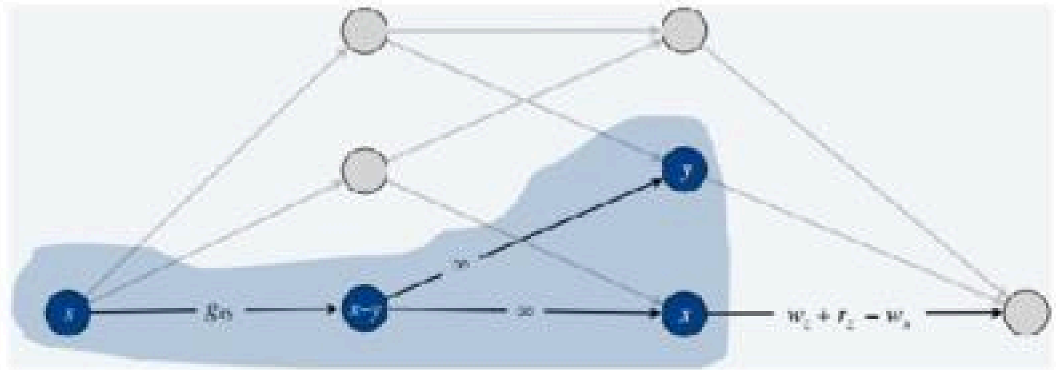
Pf. $\Rightarrow$

- Since team z is eliminated, by MF-MC theorem, $g(S - \{z\})$ is not saturated, so:

$$
\begin{aligned}
g(S - \{z\}) > cap(A, B) & \\
&= [g(S - \{z\}) - g(T^*)] + [\sum_{x \in T^*} (w_z + g_z - w_x)] \\
&= [g(S - \{z\}) - g(T^*)] + [w(T^*) + |T^*|(w_z + g_z)]
\end{aligned}
$$

- Rearranging terms: $w_z + g_z < \frac{w(T^*) + g(T^*)}{|T^*|}$

≡



y
∞
∞
x