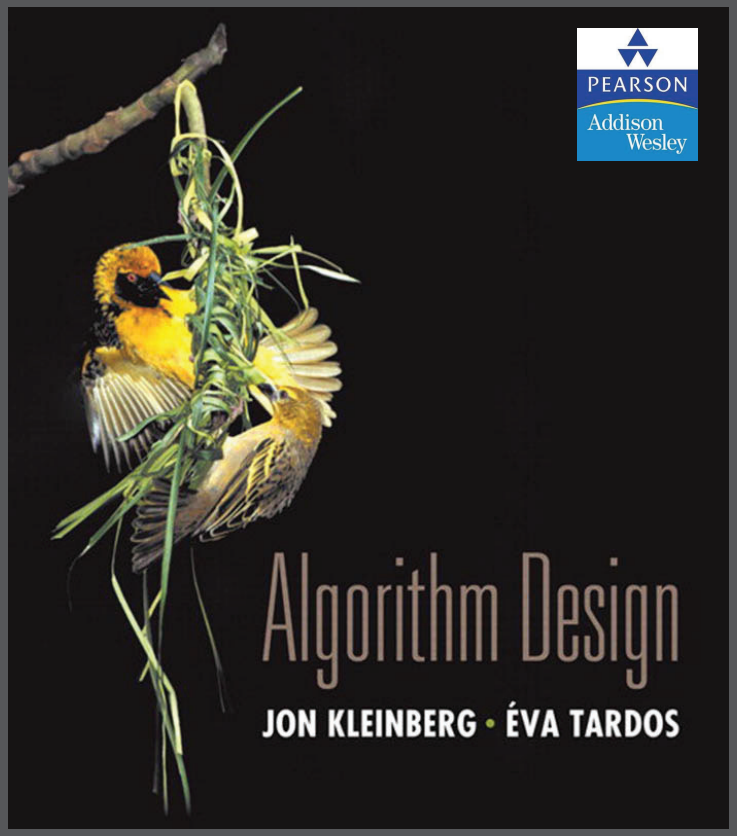


7. Network Flow II

- *bipartite matching*
- *disjoint paths*
- *extensions to max flow*
- *survey design*
- *airline scheduling*
- *image segmentation*
- *project selection*
- *baseball elimination*

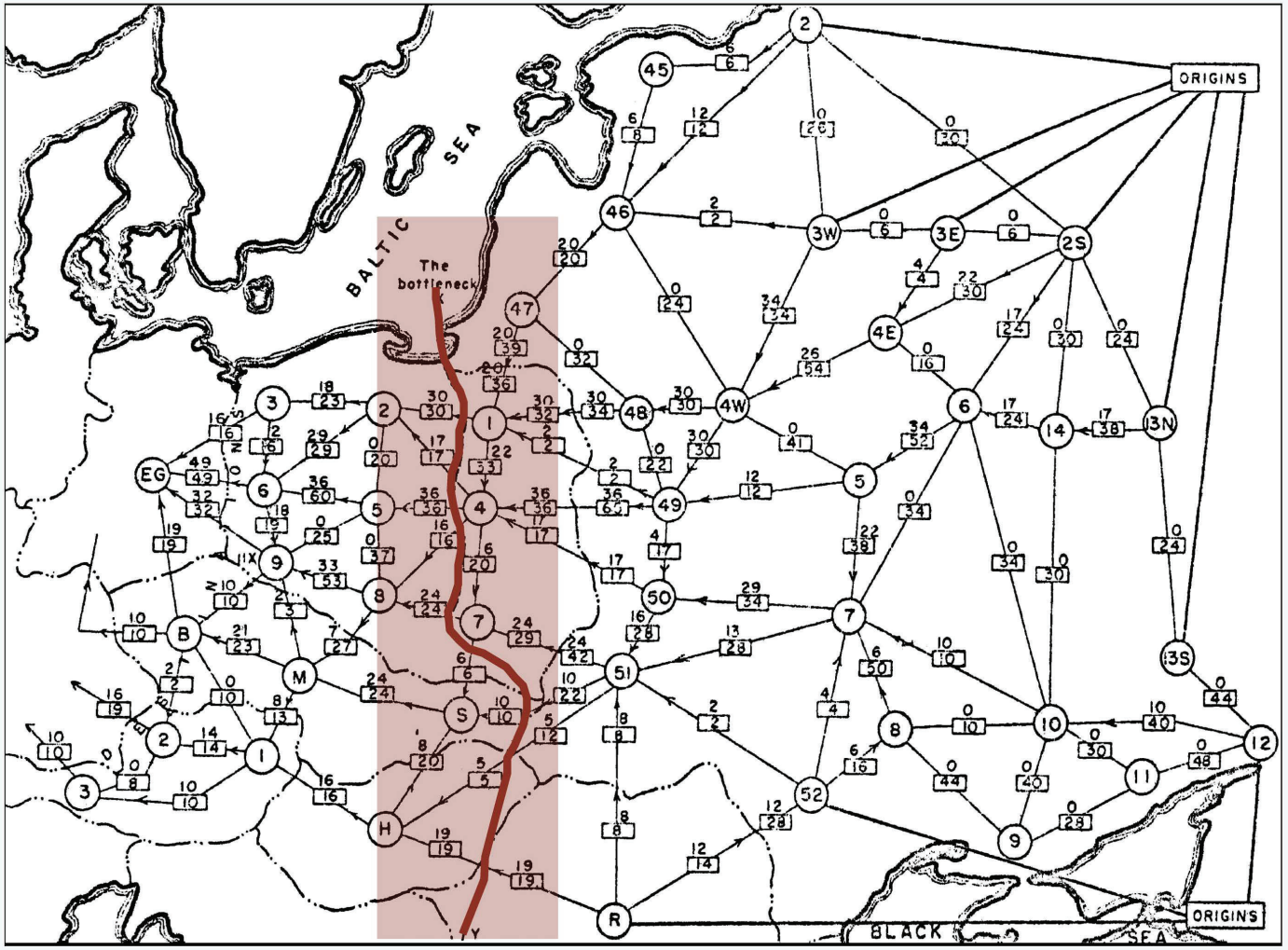
Lecture slides by Kevin Wayne

http://www.cs.princeton.edu/~wayne/kleinberg-tardos

Last updated on 1/14/20 2:20 PM

Minimum cut application (RAND 1950s)

"Free world" goal. Cut supplies (if Cold War turns into real war).

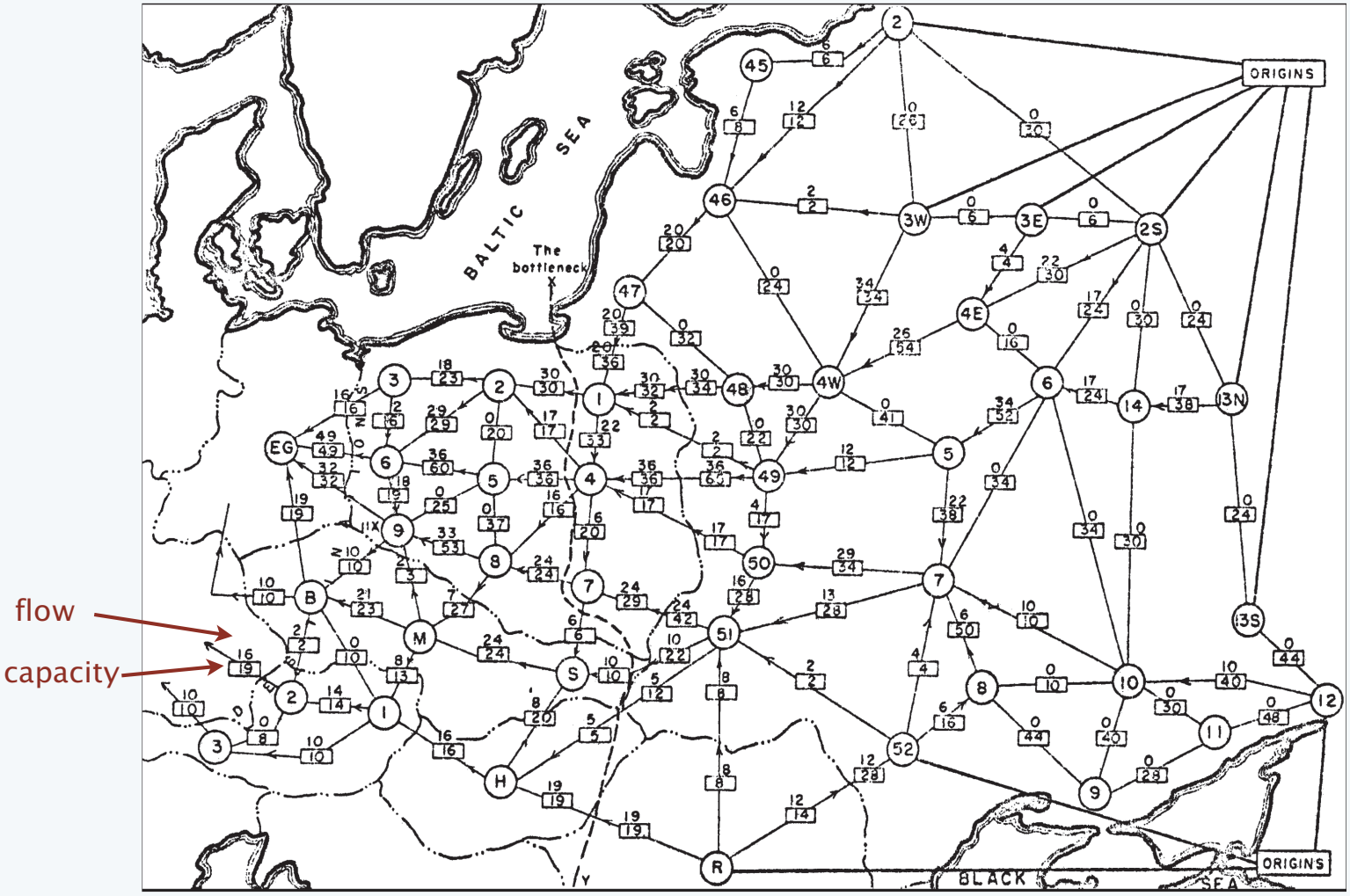


rail network connecting Soviet Union with Eastern European countries

(map declassified by Pentagon in 1999)

Maximum flow application (Tolstoĭ 1930s)

Soviet Union goal. Maximize flow of supplies to Eastern Europe.



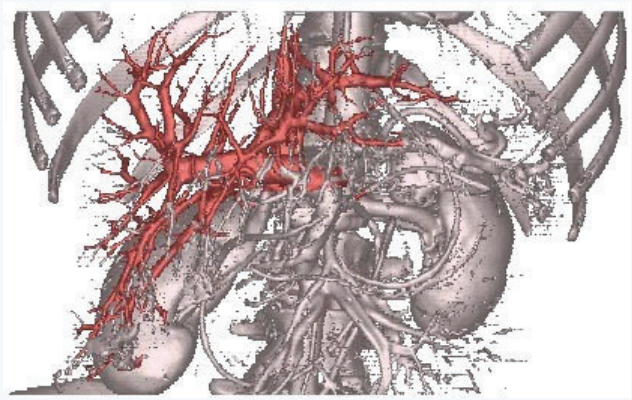
rail network connecting Soviet Union with Eastern European countries

(map declassified by Pentagon in 1999)

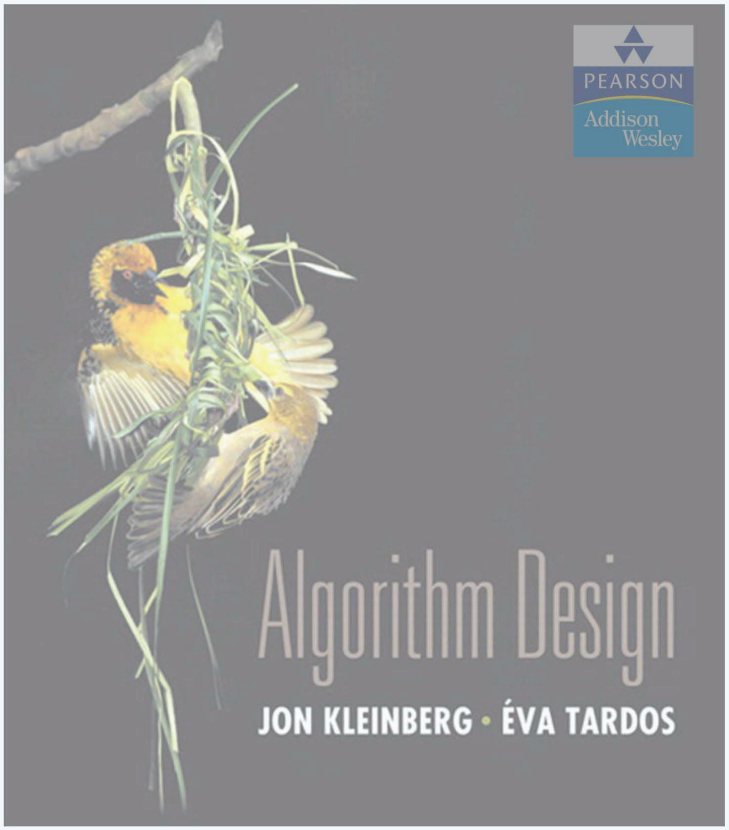
Max-flow and min-cut applications

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.



liver and hepatic vascularization segmentation



SECTION 7.5

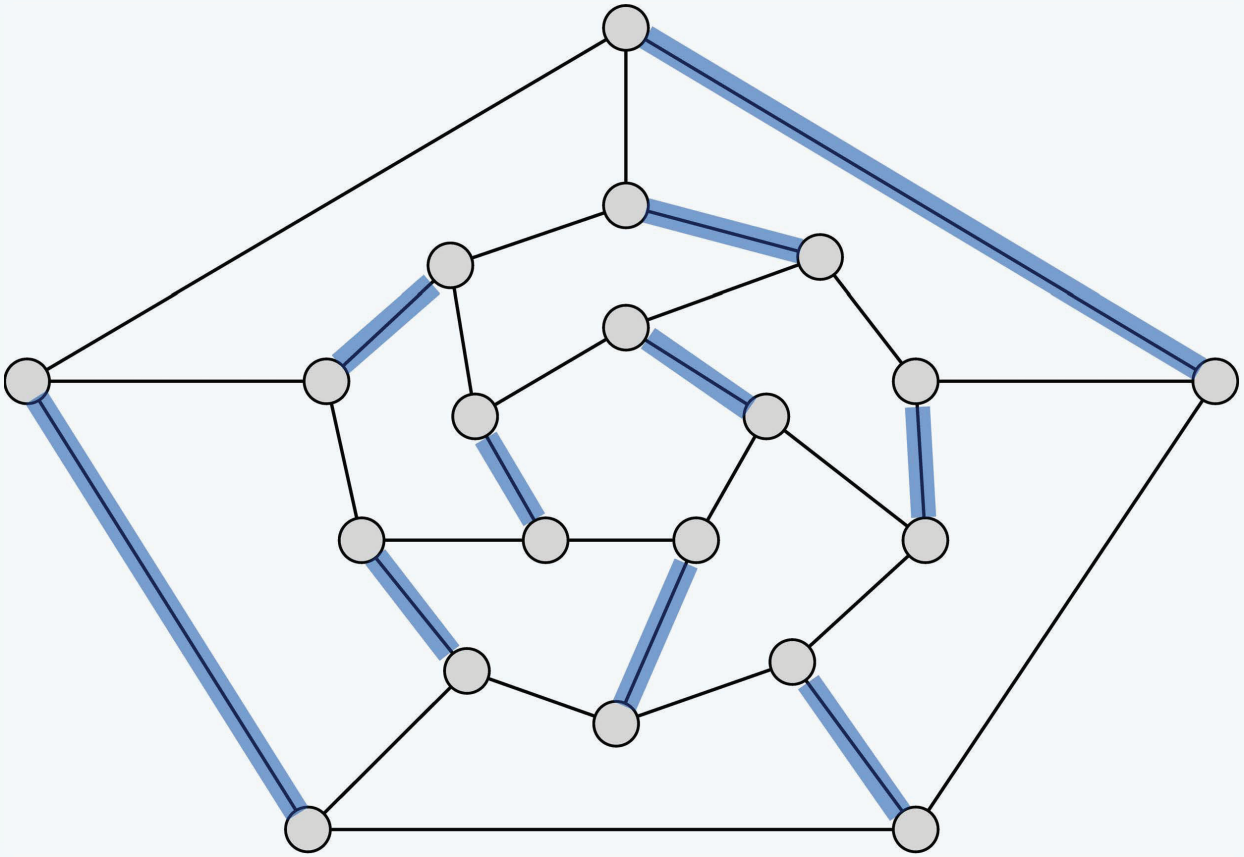
7. NETWORK FLOW II

- *bipartite matching*
- *disjoint paths*
- *extensions to max flow*
- *survey design*
- *airline scheduling*
- *image segmentation*
- *project selection*
- *baseball elimination*

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.

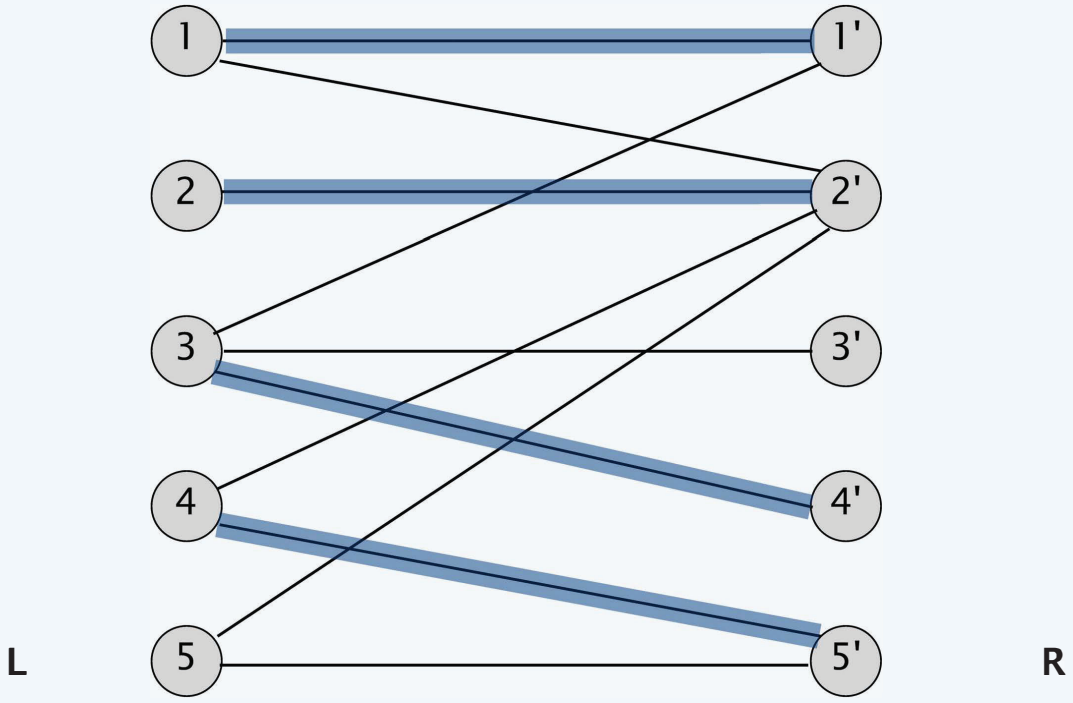
Max matching. Given a graph G, find a max-cardinality matching.



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.



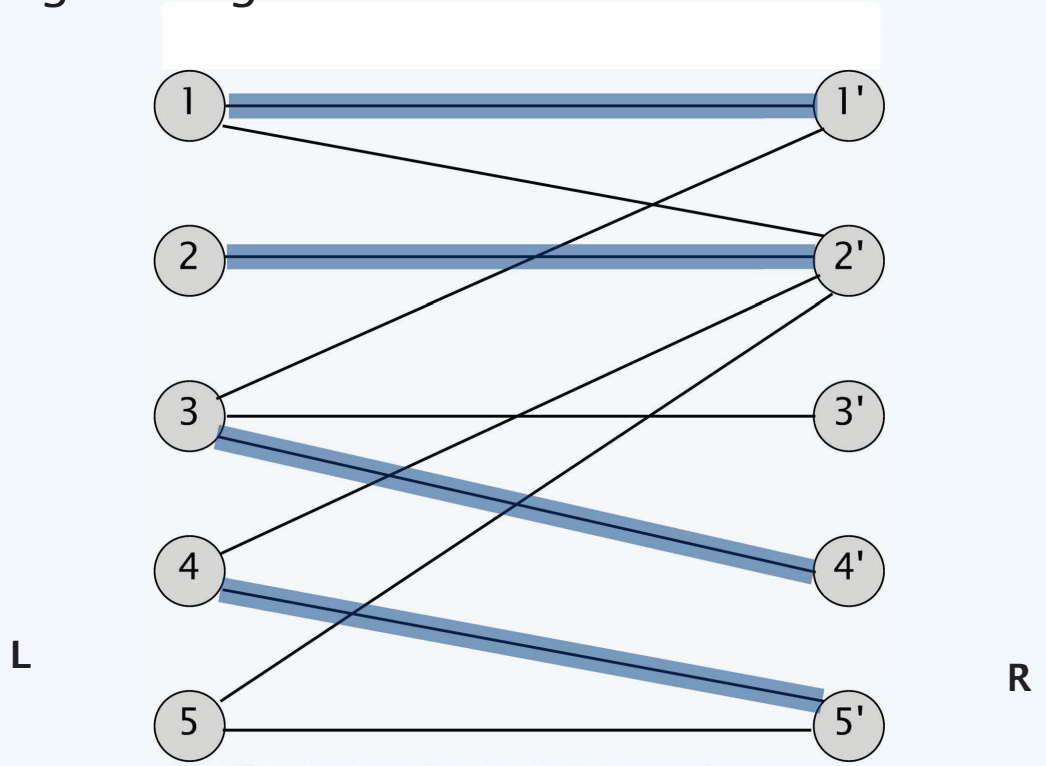
matching: 1-1', 2-2', 3-4', 4-5'

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.

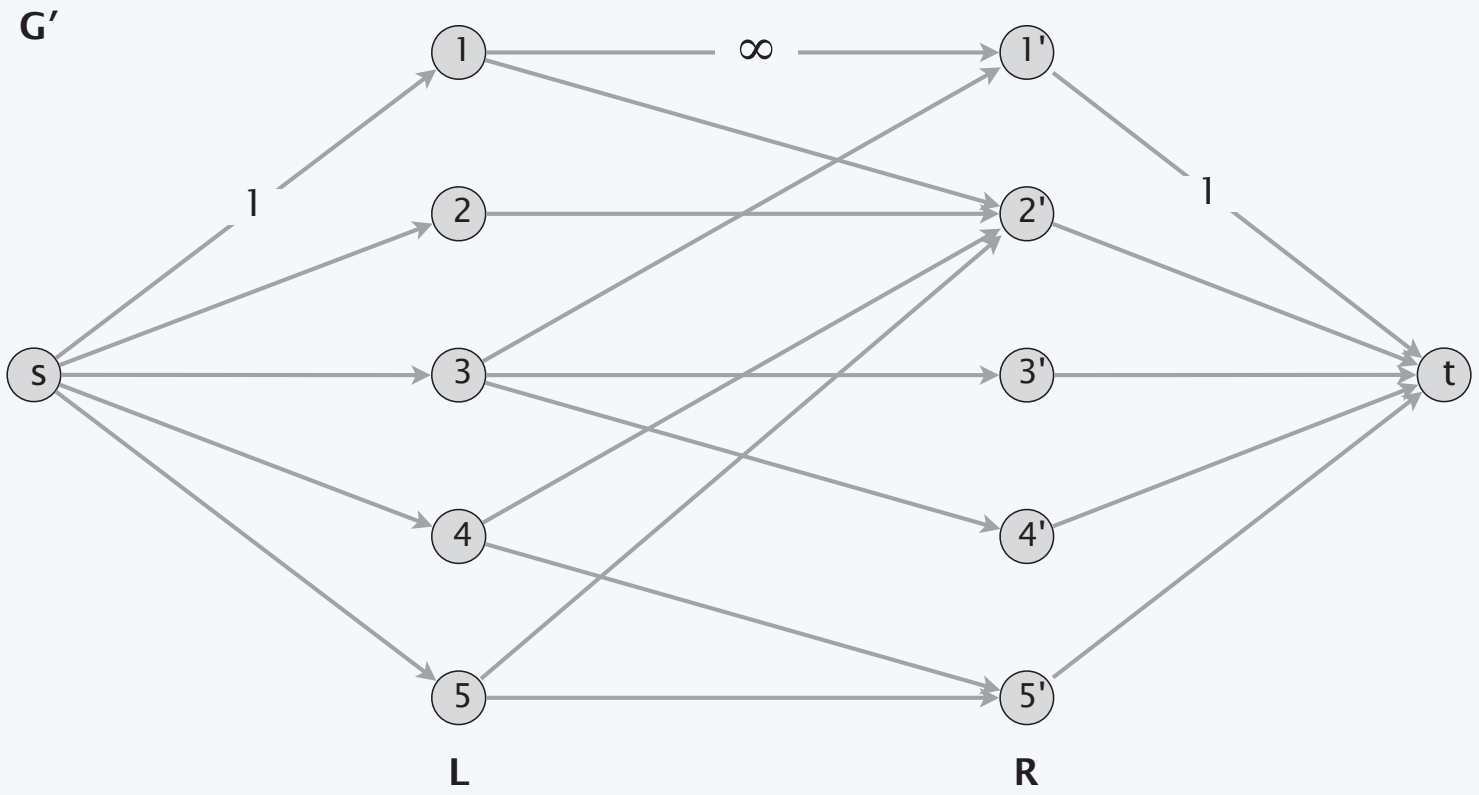
Challenge: Design an algorithm that finds a max-cardinality matching.



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: proof of correctness

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

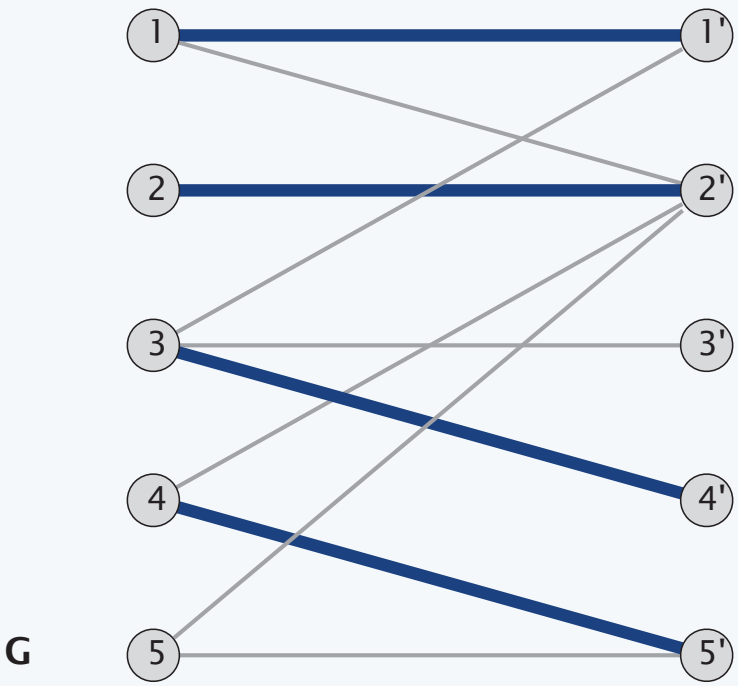
Pf. for each edge e: $f(e) \in \{ 0, 1 \}$

Max-flow formulation: proof of correctness

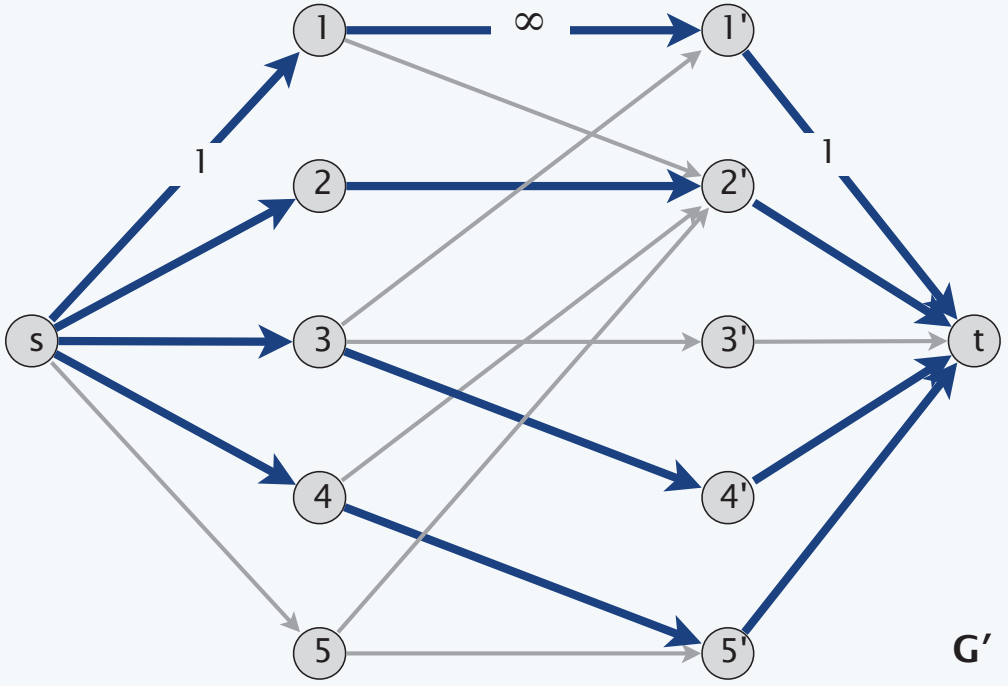
Theorem. 1–1 correspondence between matchings of cardinality k in G and integral flows of value k in G'.

Pf. $\Rightarrow$ for each edge e: $f(e) \in \{ 0, 1 \}$

- 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. ▪



G



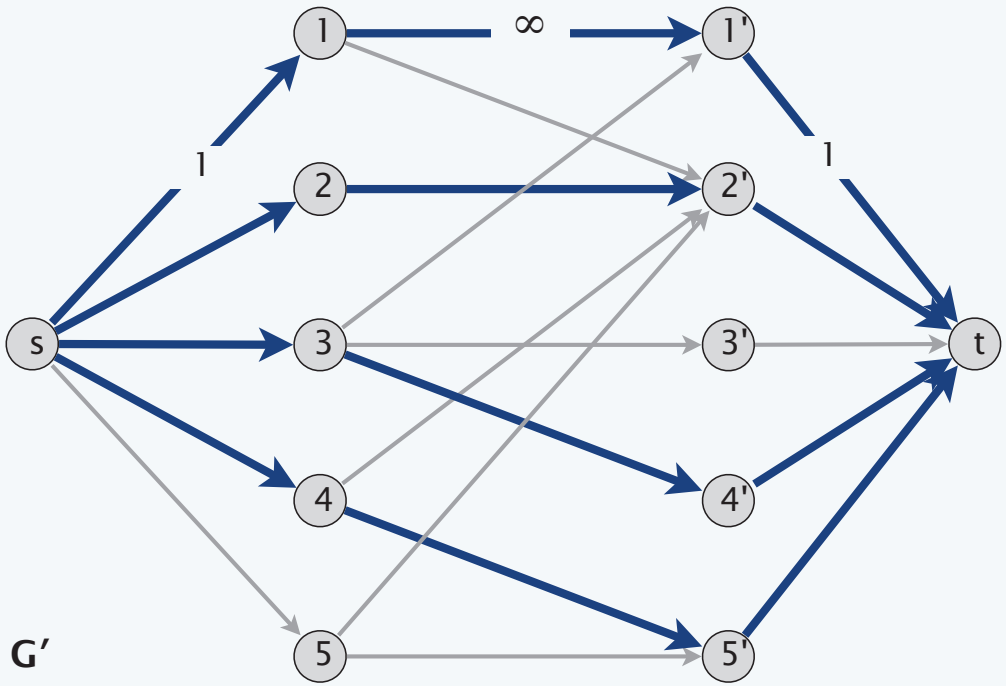
G′

Max-flow formulation: proof of correctness

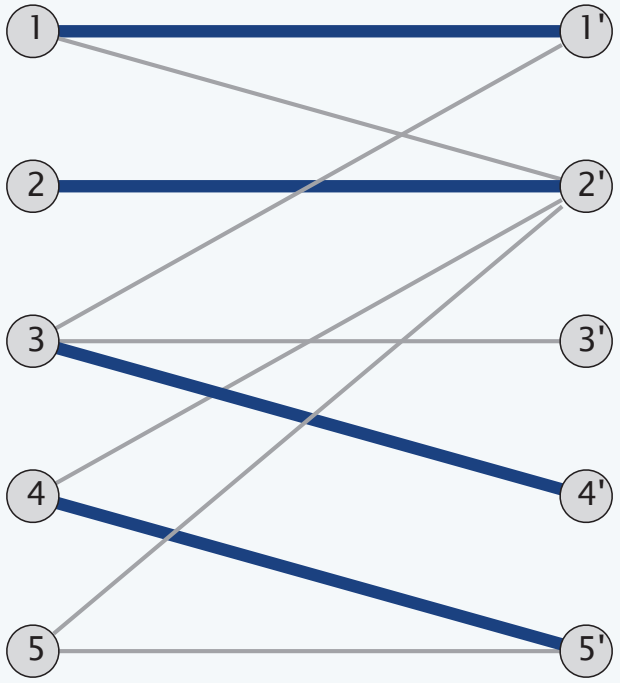
Theorem. 1–1 correspondence between matchings of cardinality k in G and integral flows of value k in G'.

Pf. $\Leftarrow$ (for each edge e: $f(e) \in \{ 0, 1 \}$)

- 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
  - $|M| = k$: apply flow-value lemma to cut $(L \cup \{s\}, R \cup \{t\})$ ▪



G′



G

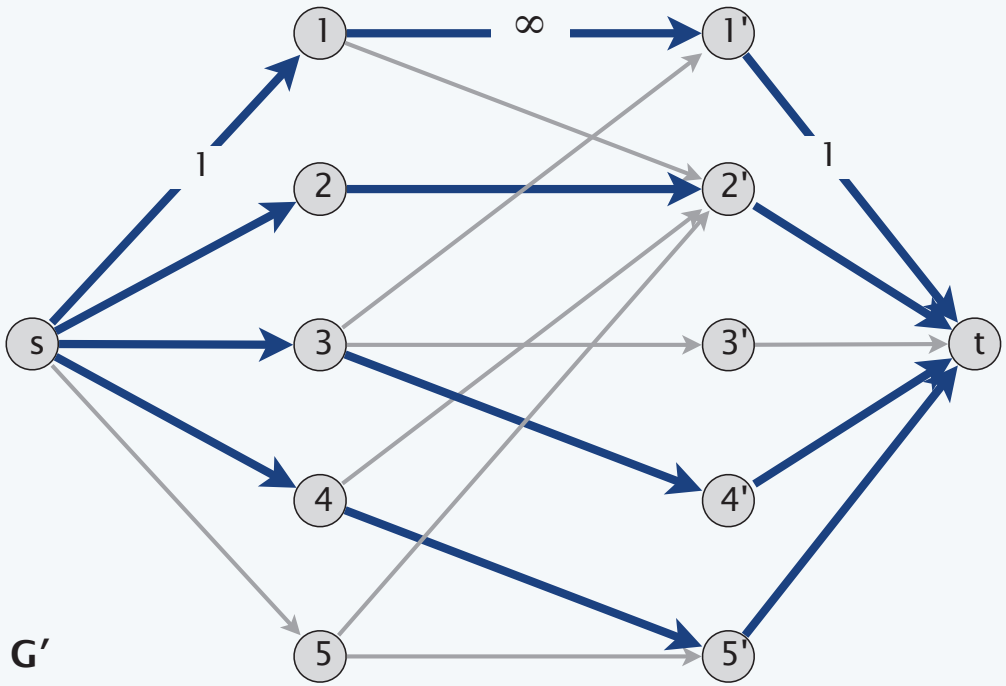
Max-flow formulation: proof of correctness

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

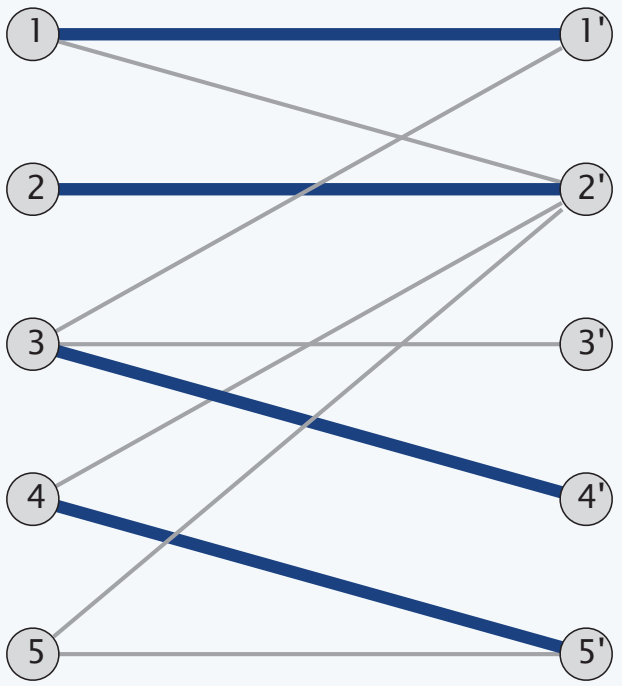
Corollary. Can solve bipartite matching problem via max-flow formulation.

Pf. *Recall integrality theorem: if all capacities in a flow network are integers, then there is a maximum flow where every value is an integer.*

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



G′



G

Network flow II: quiz 1

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. $O(m^2 n)$

Perfect matchings in bipartite graphs

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 $|L| = |R|$.
- Which other conditions are necessary?
- Which other conditions are sufficient?

Perfect matchings in bipartite graphs

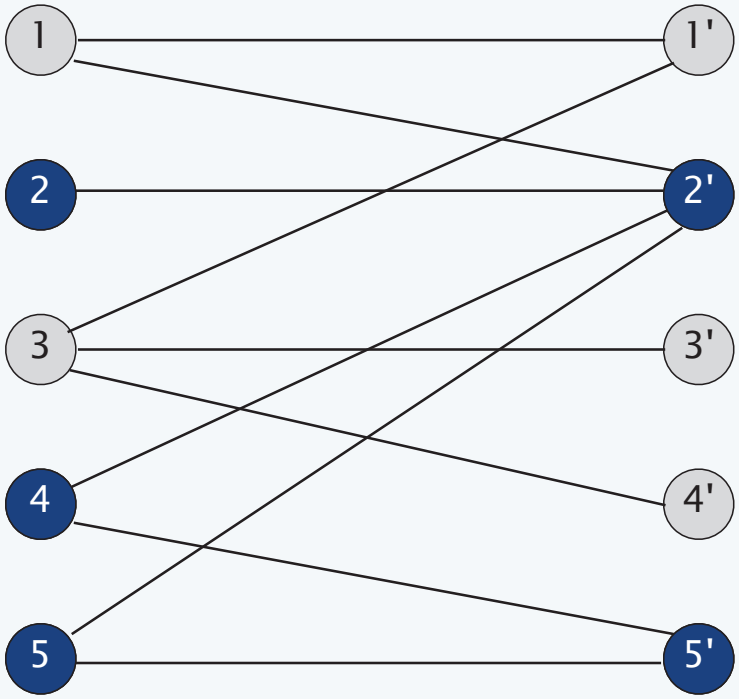
Notation. Let S be a subset of nodes, and let $N(S)$ be the set of nodes adjacent to nodes in S.

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 $N(S)$. ▪

S = { 2, 4, 5 }

N(S) = { 2', 5' }



no perfect matching

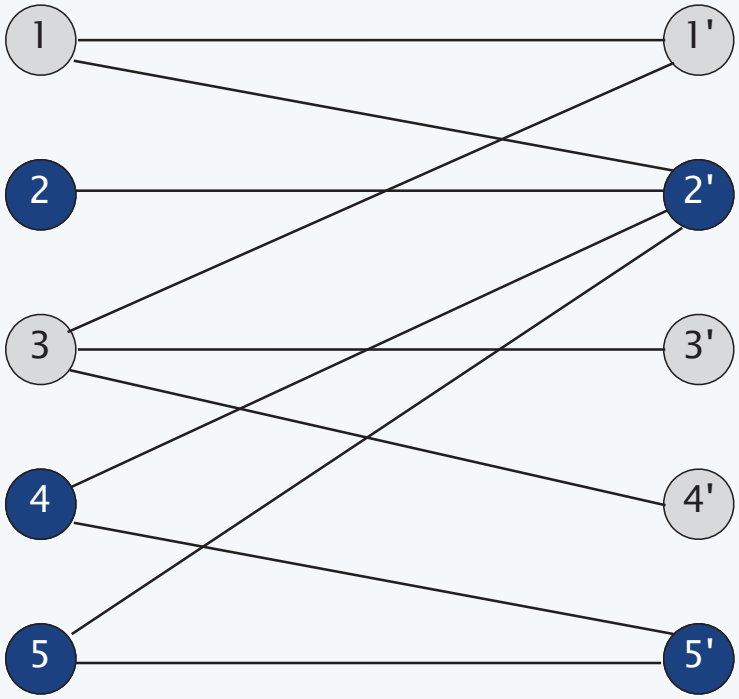
Hall's marriage theorem

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 $S \subseteq L$.

Proof?

S = { 2, 4, 5 }

N(S) = { 2', 5' }



no perfect matching

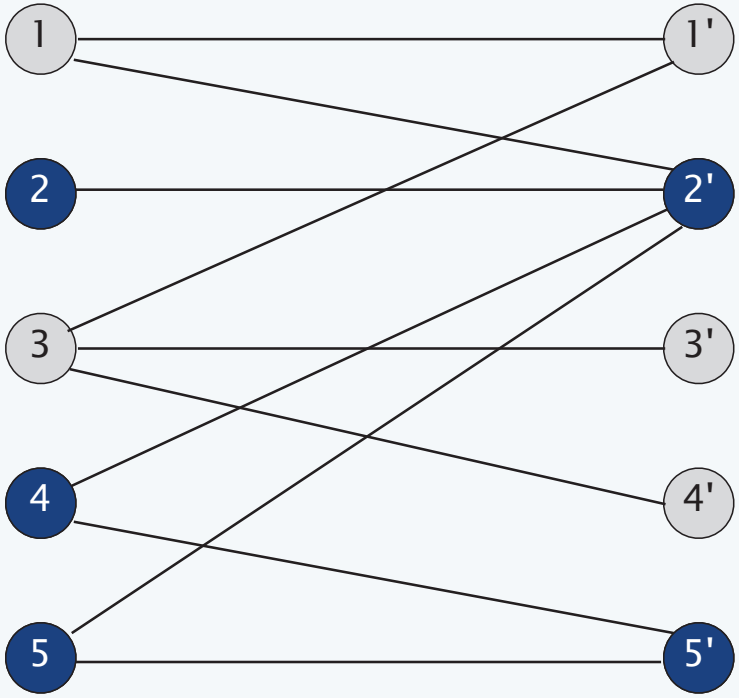
Hall's marriage theorem

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 was the previous observation.

S = { 2, 4, 5 }

N(S) = { 2', 5' }



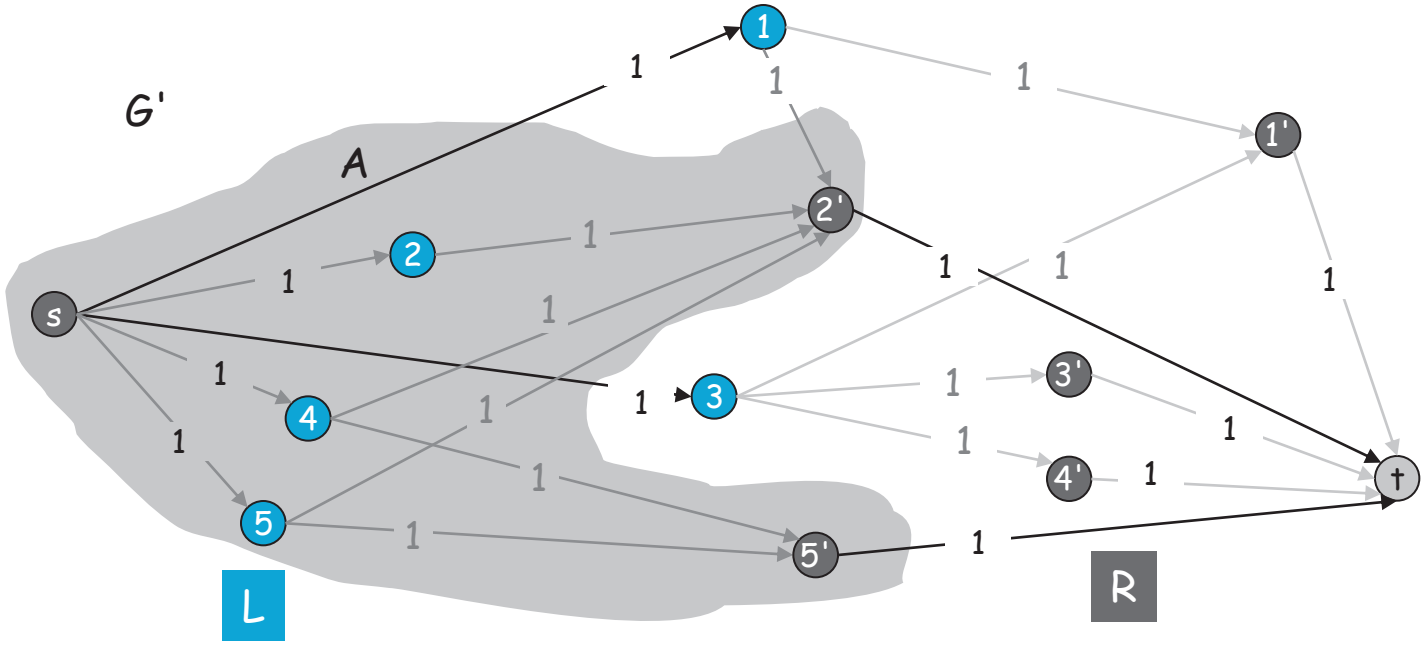
no perfect matching

Hall's marriage theorem

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

L-R edges have infinite capacity => not part of the cut.

- Formulate as a max flow problem and let (A, B) be min cut in G'.
- Define $L_A = L \cap A$, $L_B = L \cap B$, $R_A = R \cap A$.
- Not perfect, so $v(f) < |L|$, so by max-flow min-cut, $cap(A, B) < |L|$.
- No L-R edges from A to B in min cut: $cap(A, B) = |L_B| + |R_A|$.
- No L-R edges from A to B in min cut: $N(L_A) \subseteq R_A$.
- So $|N(L_A)| \leq |R_A| = cap(A, B) - |L_B| < |L| - |L_B| = |L_A|$.
- Choose $S = L_A$. Then $|N(S)| < |S|$. ▪



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

Bipartite matching

Problem. Given a bipartite graph, find a max-cardinality matching.

year	worst case	technique	discovered by
1955	$O(m\,n)$	**augmenting path**	Ford–Fulkerson
1973	$O(m\,n^{1/2})$	**blocking flow**	Hopcroft–Karp, Karzanov
2004	$O(n^{2.378})$	**fast matrix multiplication**	Mucha–Sankowsi
2013	$\tilde{O}(m^{10/7})$	**electrical flow**	Mądry
20xx			

running time for finding a max-cardinality matching in a bipartite graph with n nodes and m edges

Network flow II: quiz 2

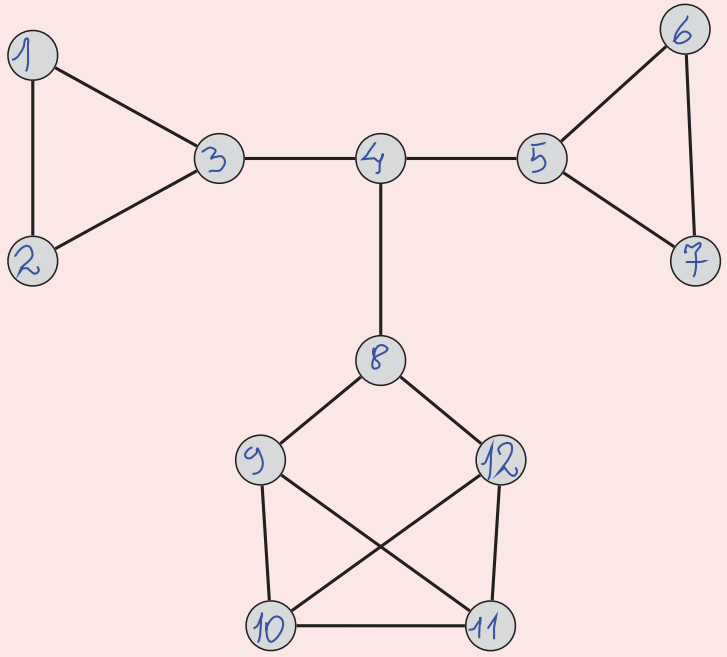
Which of the following are properties of the graph G = (V, E)?

A. G has a perfect matching.

B. Hall's condition is satisfied: $|N(S)| \geq |S|$ for all subsets $S \subseteq V$.

C. Both A and B.

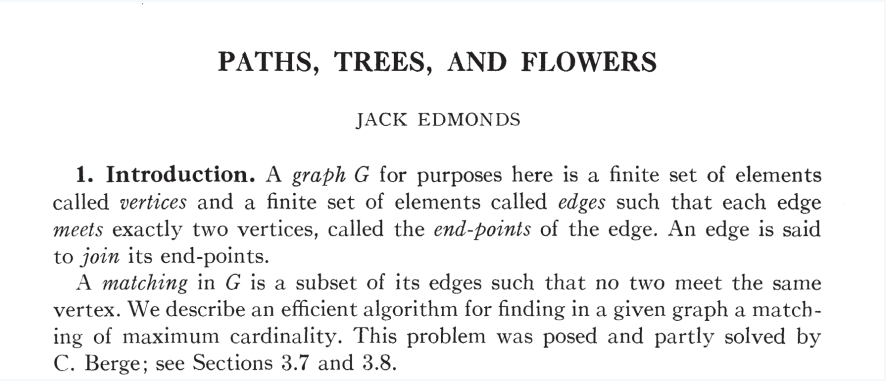
D. Neither A nor B.



Nonbipartite matching

Problem. Given an undirected graph, find a max-cardinality matching.

- Structure of nonbipartite graphs is more complicated.
- But well understood. [Tutte–Berge formula, Edmonds–Gallai]
- Blossom algorithm: $O(n^4)$. [Edmonds 1965]
- Best known: $O(m\, n^{1/2})$. [Micali–Vazirani 1980, Vazirani 1994]

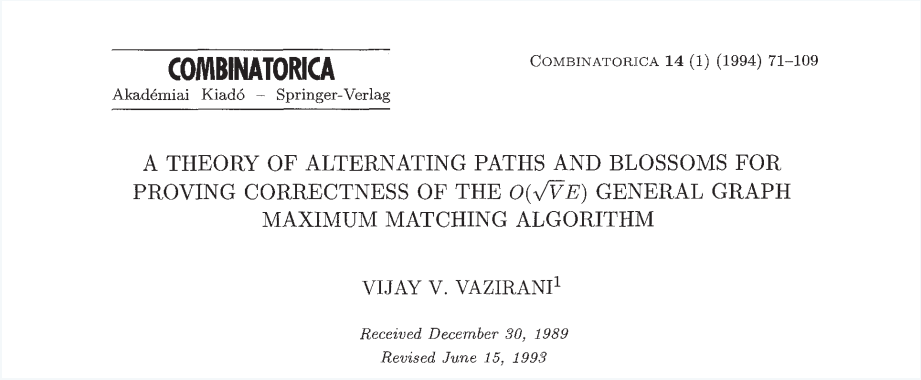


PATHS, TREES, AND FLOWERS

JACK EDMONDS

1. Introduction. A *graph* G for purposes here is a finite set of elements called *vertices* and a finite set of elements called *edges* such that each edge *meets* exactly two vertices, called the *end-points* of the edge. An edge is said to *join* its end-points.

A *matching* in G is a subset of its edges such that no two meet the same vertex. We describe an efficient algorithm for finding in a given graph a matching of maximum cardinality. This problem was posed and partly solved by C. Berge; see Sections 3.7 and 3.8.



COMBINATORICA
Akadémiai Kiadó – Springer-Verlag

COMBINATORICA **14** (1) (1994) 71–109

A THEORY OF ALTERNATING PATHS AND BLOSSOMS FOR PROVING CORRECTNESS OF THE $O(\sqrt{V}E)$ GENERAL GRAPH MAXIMUM MATCHING ALGORITHM

VIJAY V. VAZIRANI[1]

Received December 30, 1989
Revised June 15, 1993

Historical significance (Jack Edmonds 1965)

2. Digression. An explanation is due on the use of the words "efficient algorithm." First, what I present is a conceptual description of an algorithm and not a particular formalized algorithm or "code."

For practical purposes computational details are vital. However, my purpose is only to show as attractively as I can that there is an efficient algorithm. According to the dictionary, "efficient" means "adequate in operation or performance." This is roughly the meaning I want—in the sense that it is conceivable for maximum matching to have no efficient algorithm. Perhaps a better word is "good."

I am claiming, as a mathematical result, the existence of a *good* algorithm for finding a maximum cardinality matching in a graph.

There is an obvious finite algorithm, but that algorithm increases in difficulty exponentially with the size of the graph. It is by no means obvious whether *or not* there exists an algorithm whose difficulty increases only algebraically with the size of the graph.

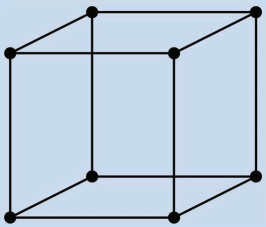
HACKATHON PROBLEM

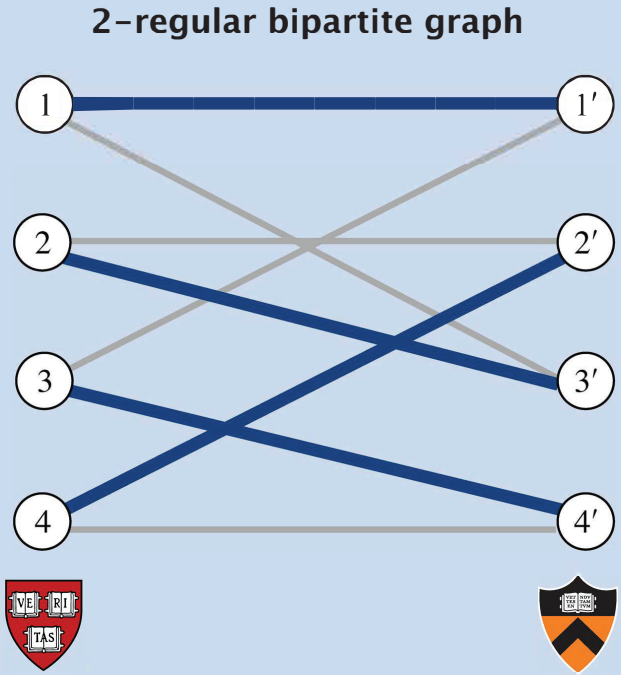
Hackathon problem.

- Hackathon attended by n Harvard students and n Princeton students.
- Each Harvard student is friends with exactly $k > 0$ Princeton students; each Princeton student is friends with exactly k Harvard students.
- Is it possible to arrange the hackathon so that each Princeton student pair programs with a different friend from Harvard?

Mathematical reformulation. Does every k-regular bipartite graph have a perfect matching?

Ex. Boolean hypercube.



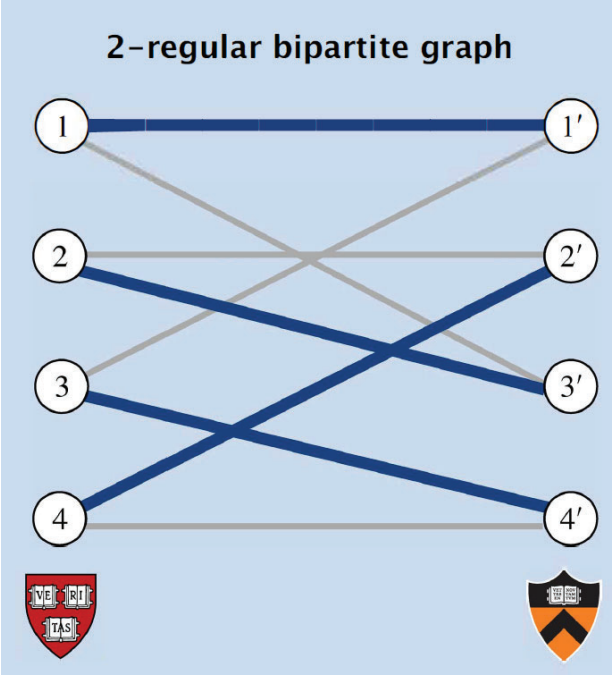


HACKATHON PROBLEM

Given k-regular bipartite graph. Does it have a perfect matching?

Proof: Let L and R be the left and right side of the bipartite graph.

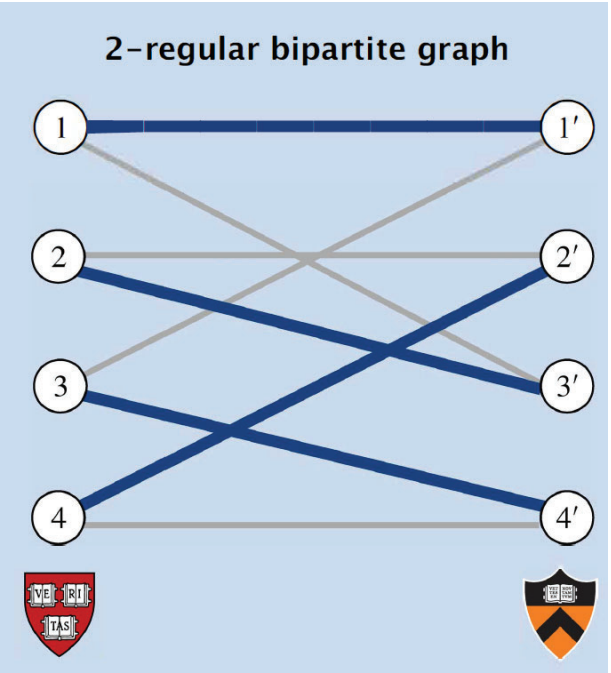
How do we proceed?



HACKATHON PROBLEM

Given k-regular bipartite graph. Does it have a perfect matching?

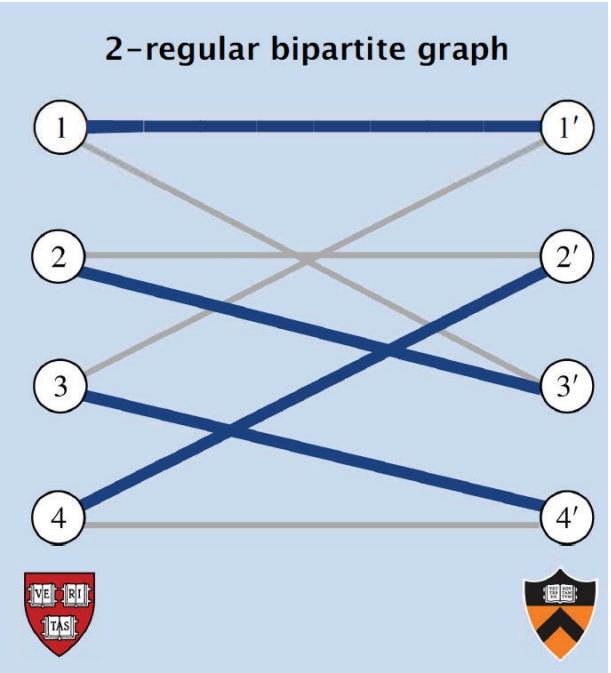
Proof: Let L and R be the left and right side of the bipartite graph.
The number of edges outgoing from L is k|L| and from R is k|R| => k|L| = k |R|. Thus |L| = |R| := n.



HACKATHON PROBLEM

Given k-regular bipartite graph. Does it have a perfect matching?

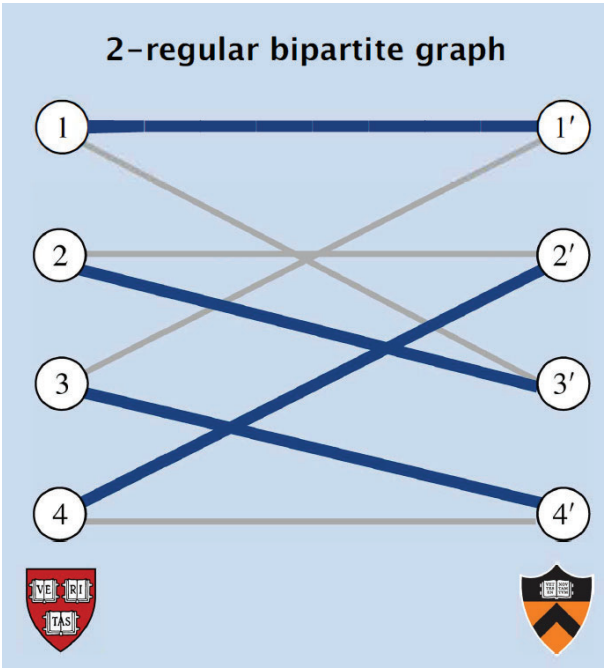
Proof: Let L and R be the left and right side of the bipartite graph.
The number of edges outgoing from L is k|L| and from R is k|R| => k|L| = k |R|. Thus |L| = |R| := n.
Let S be a subset of L. The number of edges adjacent to S is k|S|. These edges are adjacent to vertices in N(S) in R.



HACKATHON PROBLEM

Given k-regular bipartite graph. Does it have a perfect matching?

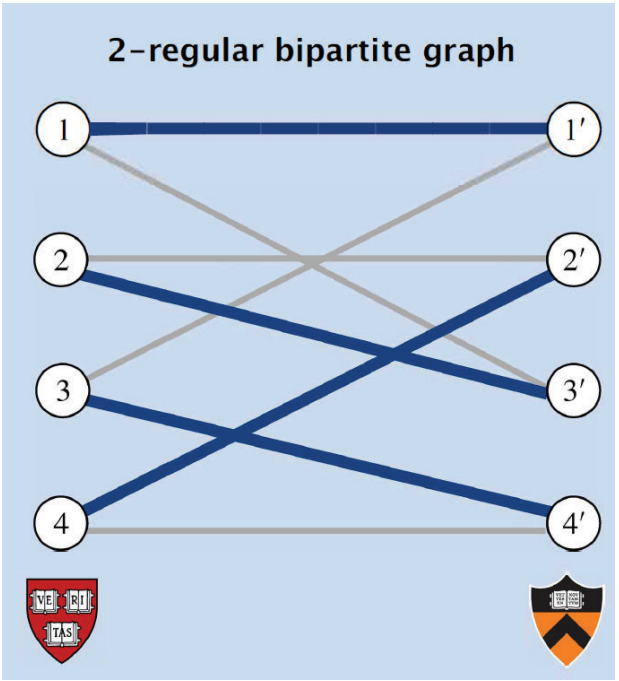
Proof: Let L and R be the left and right side of the bipartite graph.
The number of edges outgoing from L is k|L| and from R is k|R| => k|L| = k |R|. Thus |L| = |R| := n.
Let S be a subset of L. The number of edges adjacent to S is k|S|. These edges are adjacent to vertices in N(S) in R.
If |N(S)| < |S|, then ?

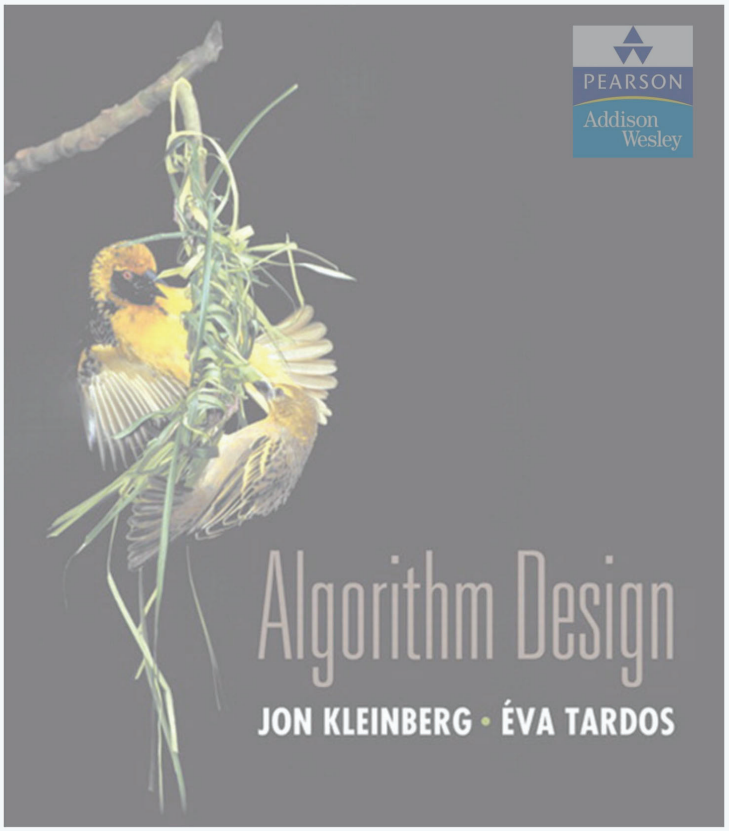


HACKATHON PROBLEM

Given k-regular bipartite graph. Does it have a perfect matching?

Proof: Let L and R be the left and right side of the bipartite graph.
The number of edges outgoing from L is k|L| and from R is k|R| => k|L| = k |R|. Thus |L| = |R| := n.
Let S be a subset of L. The number of edges adjacent to S is k|S|. These edges are adjacent to vertices in N(S) in R.
If |N(S)| < |S|, then some vertex in N(S) would have degree strictly more than k, contradicting k-regularity.
Thus |N(S)| >= |S| for each subset S of L.





SECTION 7.6

7. NETWORK FLOW II

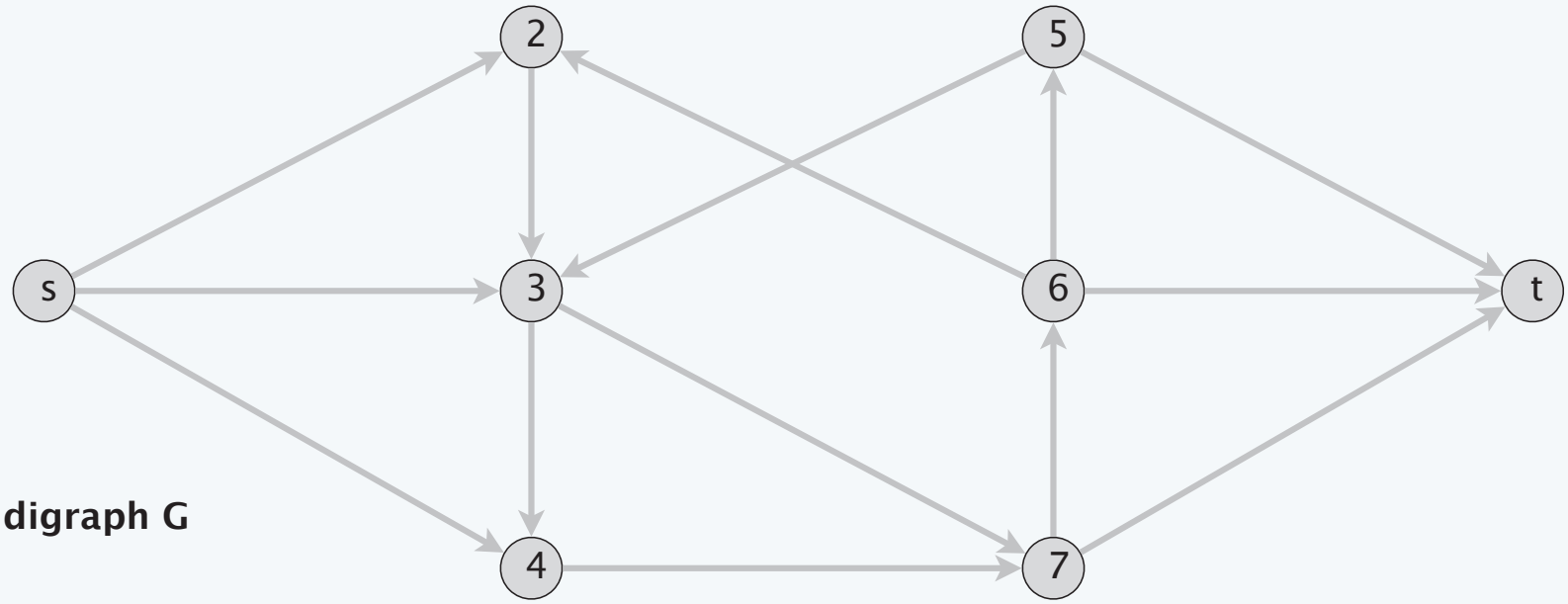
- *bipartite matching*
- *disjoint paths*
- *extensions to max flow*
- *survey design*
- *airline scheduling*
- *image segmentation*
- *project selection*
- *baseball elimination*

Edge-disjoint paths

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

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

Ex. Communication networks.



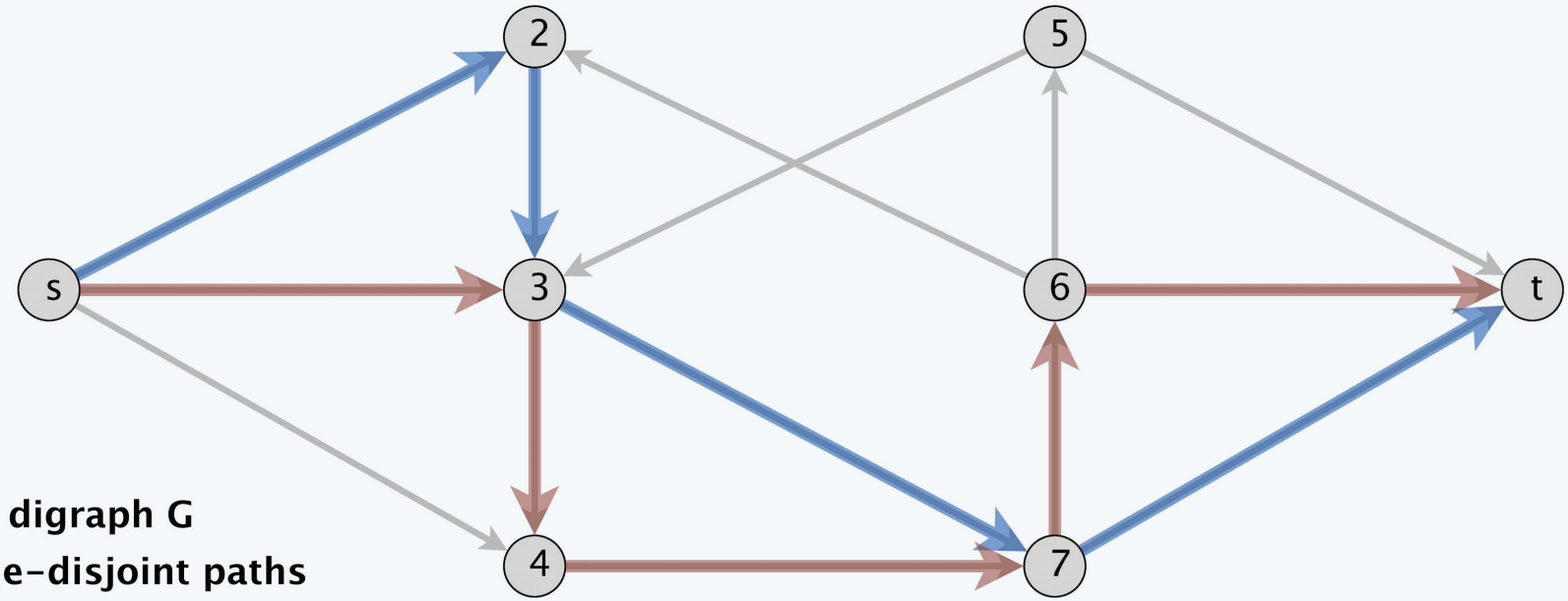
digraph G

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.



digraph G
2 edge-disjoint paths

Edge-disjoint paths

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

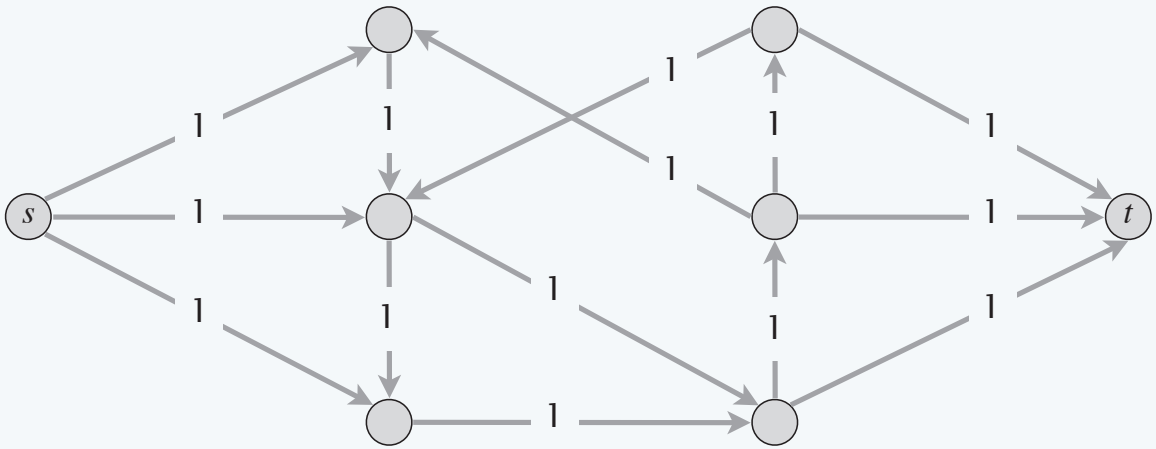
How can we use max flow to learn the number of edge disjoint paths?

Edge-disjoint paths

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

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

Proof?



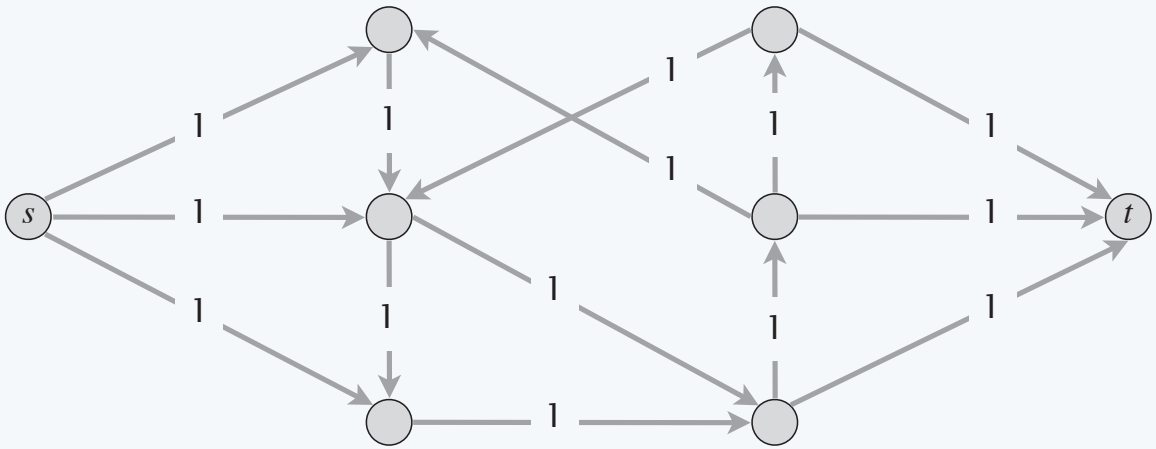
Edge-disjoint paths

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.



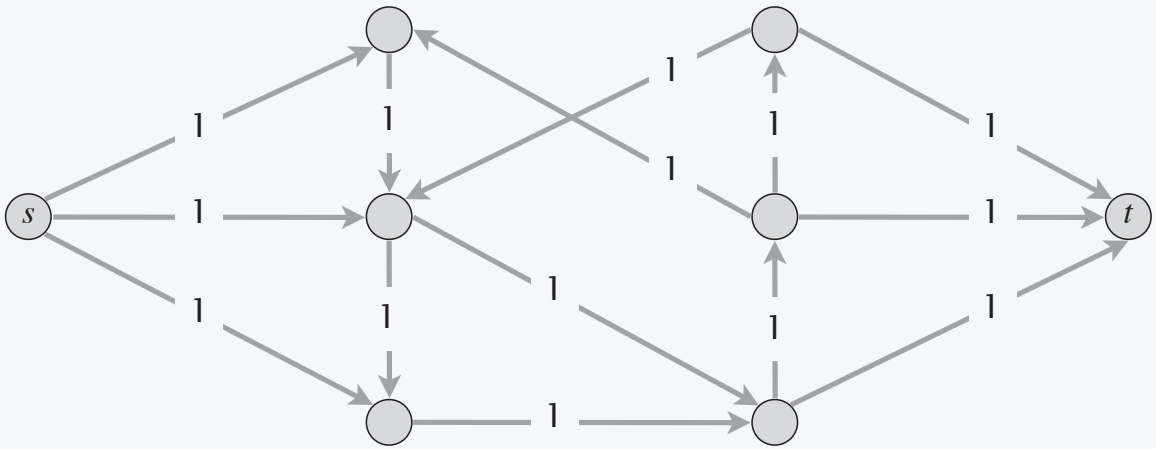
Edge-disjoint paths

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 $f(e) = \begin{cases} 1 & \text{edge } e \text{ participates in some path } P_j \\ 0 & \text{otherwise} \end{cases}$



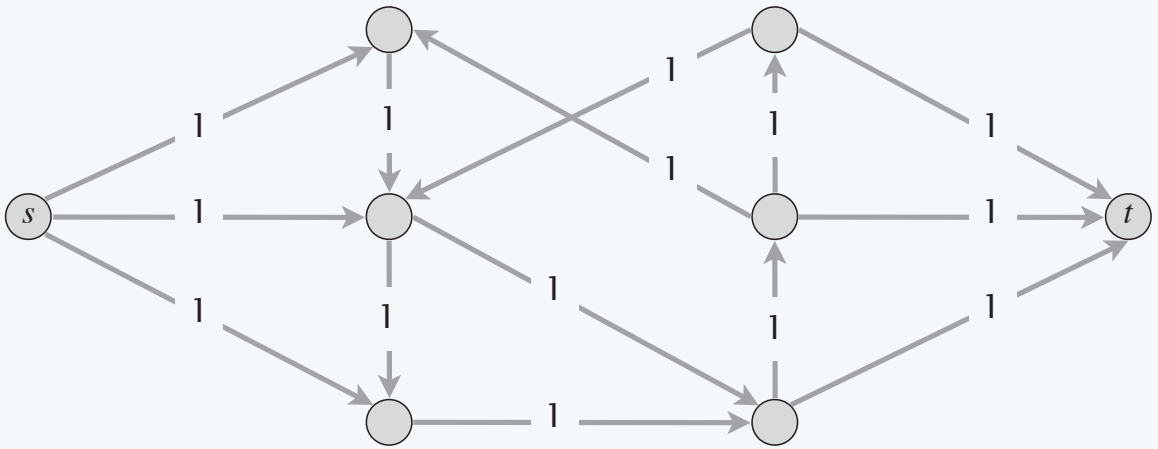
Edge-disjoint paths

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 $f(e) = \begin{cases} 1 & \text{edge } e \text{ participates in some path } P_j \\ 0 & \text{otherwise} \end{cases}$
- Since paths are edge-disjoint, f is a flow of value k. ▪



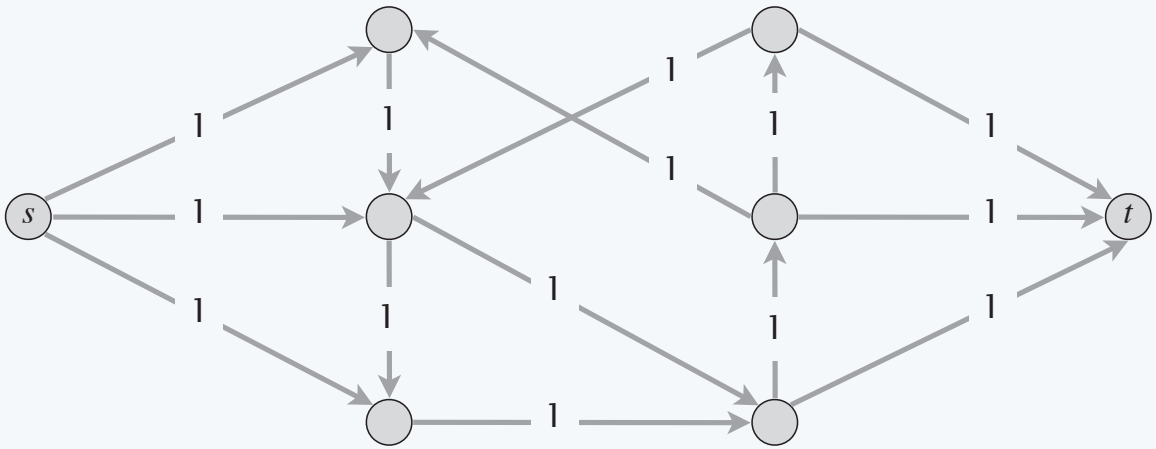
Edge-disjoint paths

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. •



Edge-disjoint paths

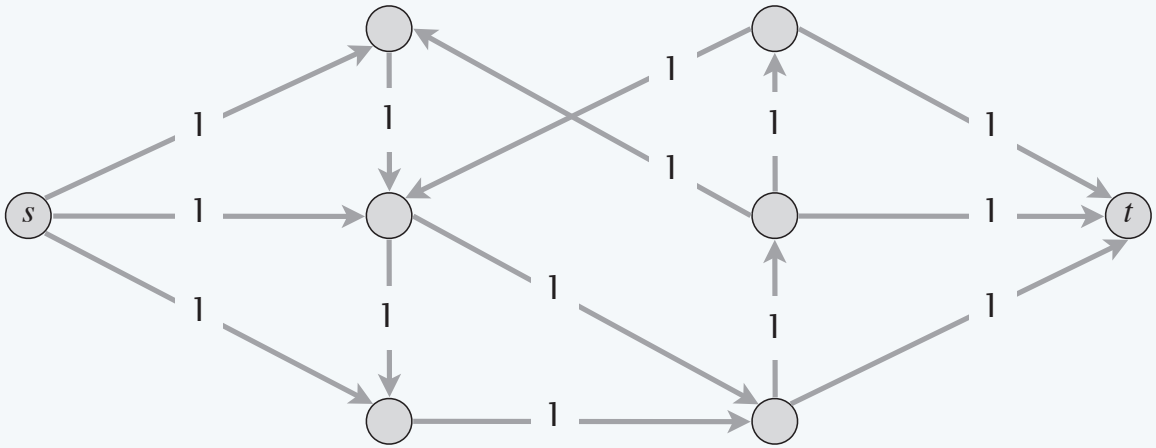
Max-flow formulation. Assign unit capacity to every edge.

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. ▪

can eliminate cycles to get simple paths in $O(mn)$ time if desired (flow decomposition)



Edge-disjoint paths

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'.

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 $s \rightsquigarrow t$ paths in G. ▪