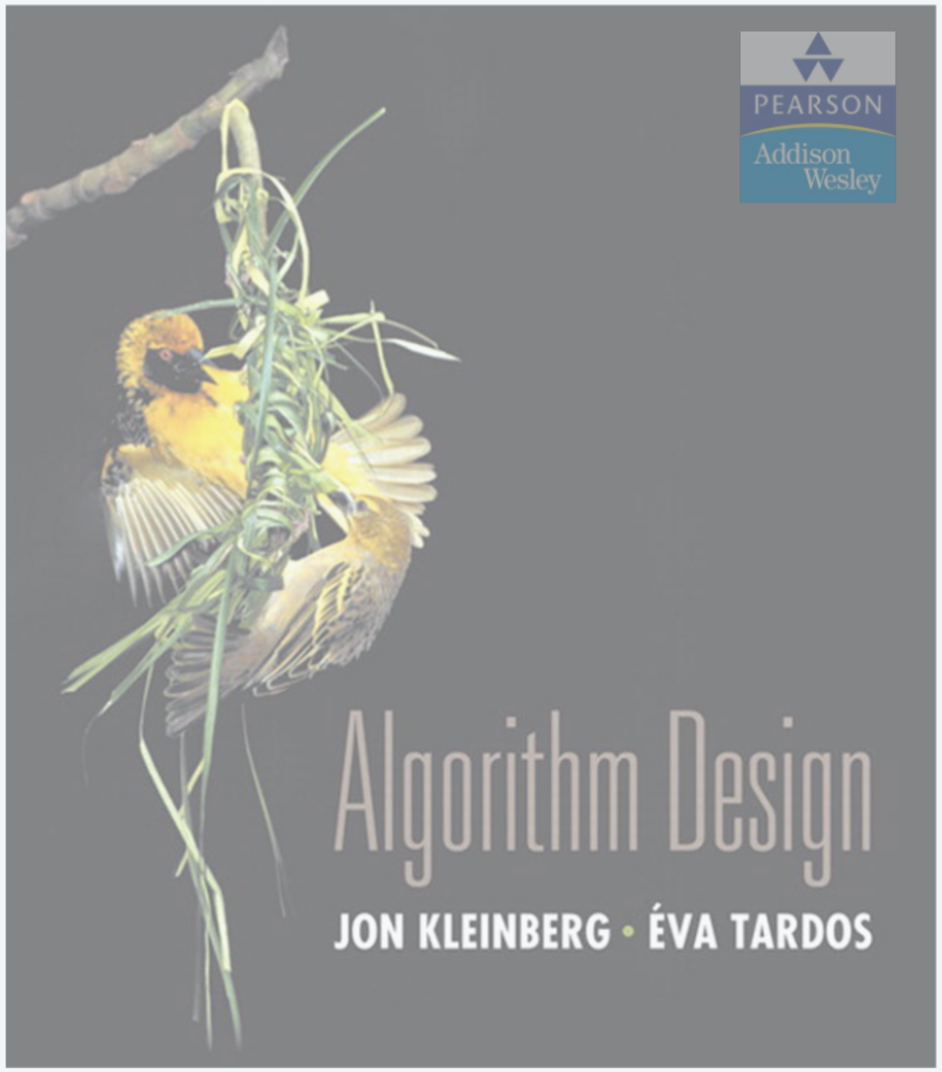


SECTION 7.5

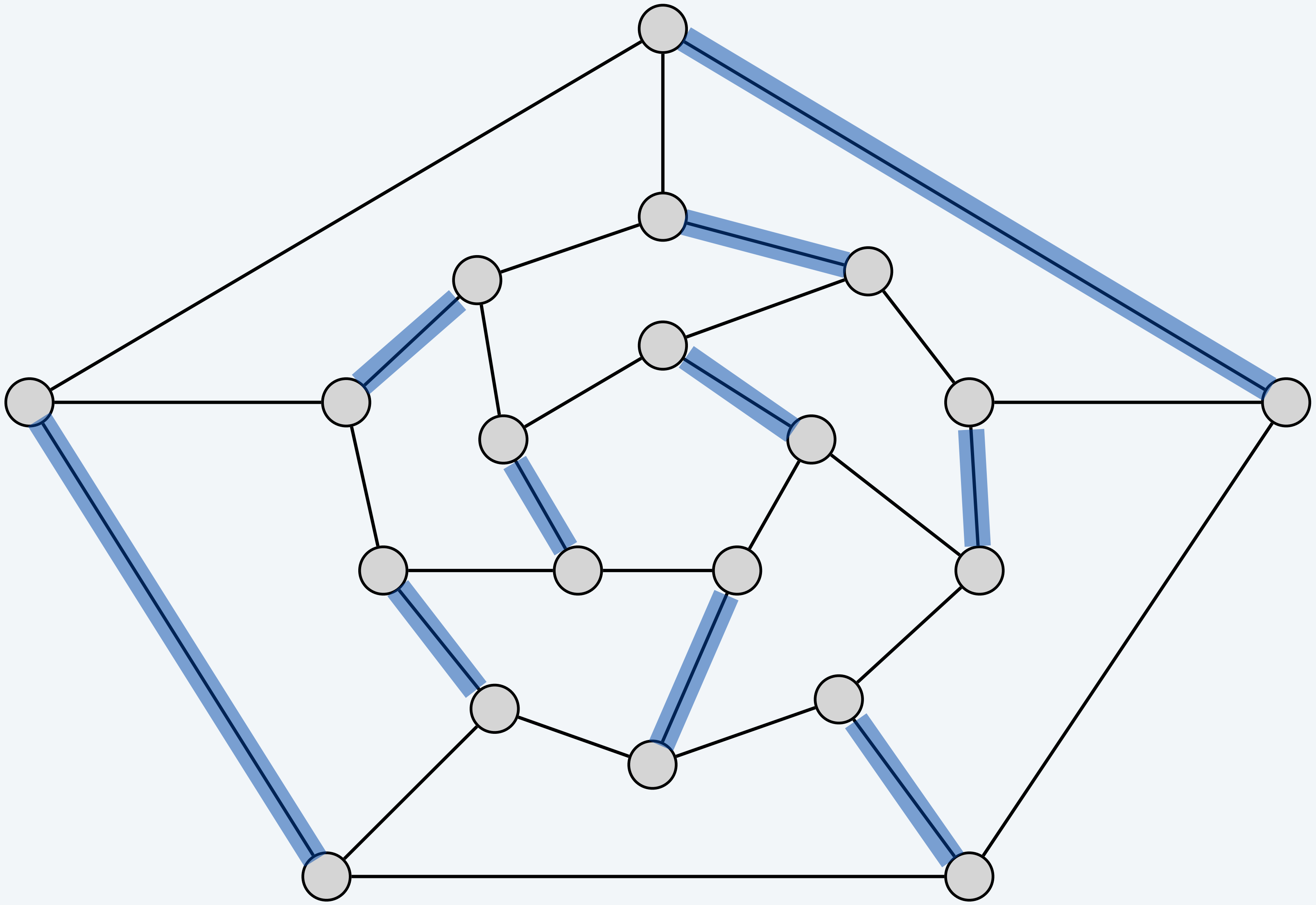
7. NETWORK FLOW II

- *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.

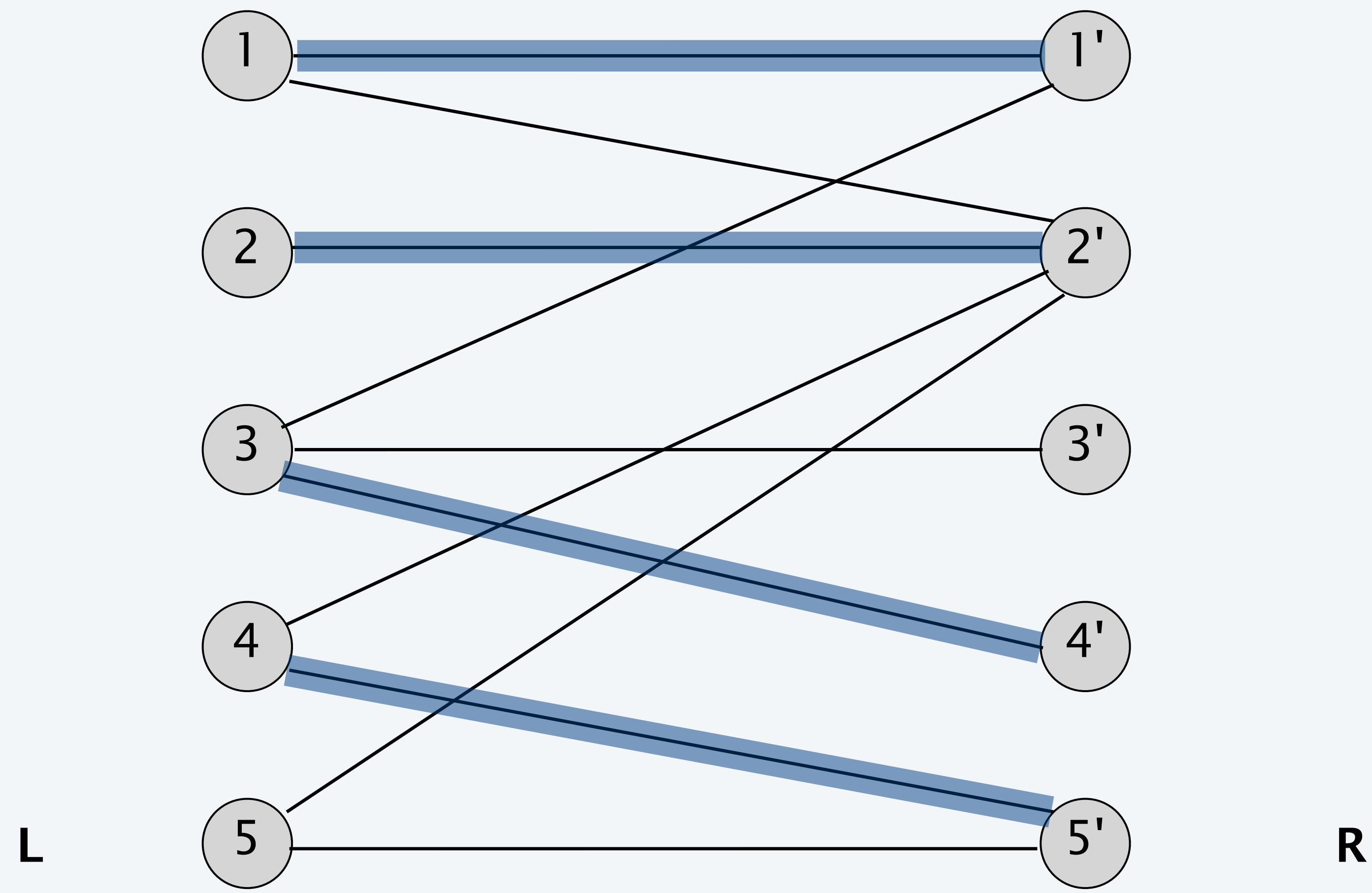
Max matching. Given a graph G, find a max-cardinality matching.



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.

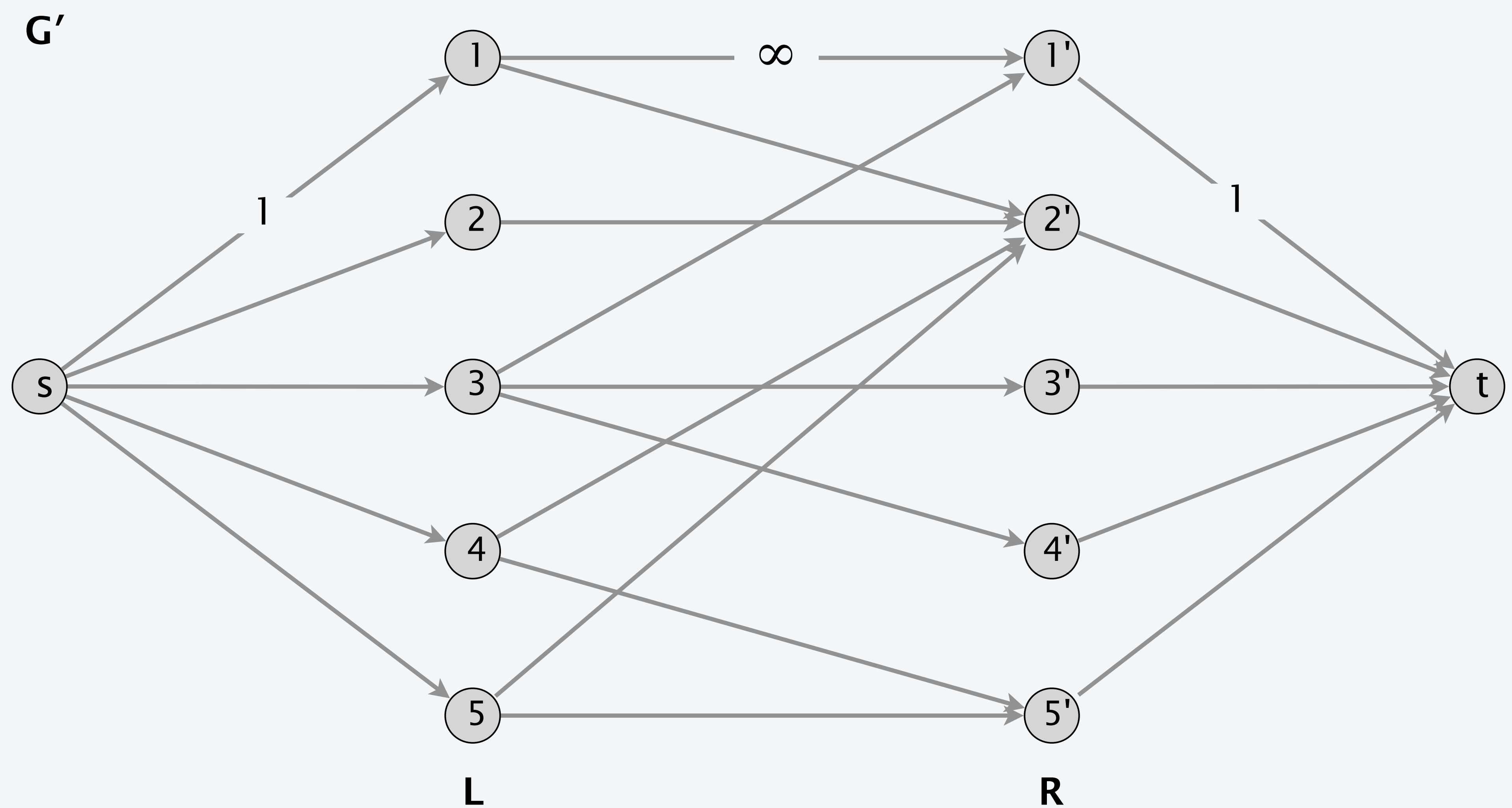


matching: 1–1', 2–2', 3–4', 4–5'

Bipartite matching: 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.



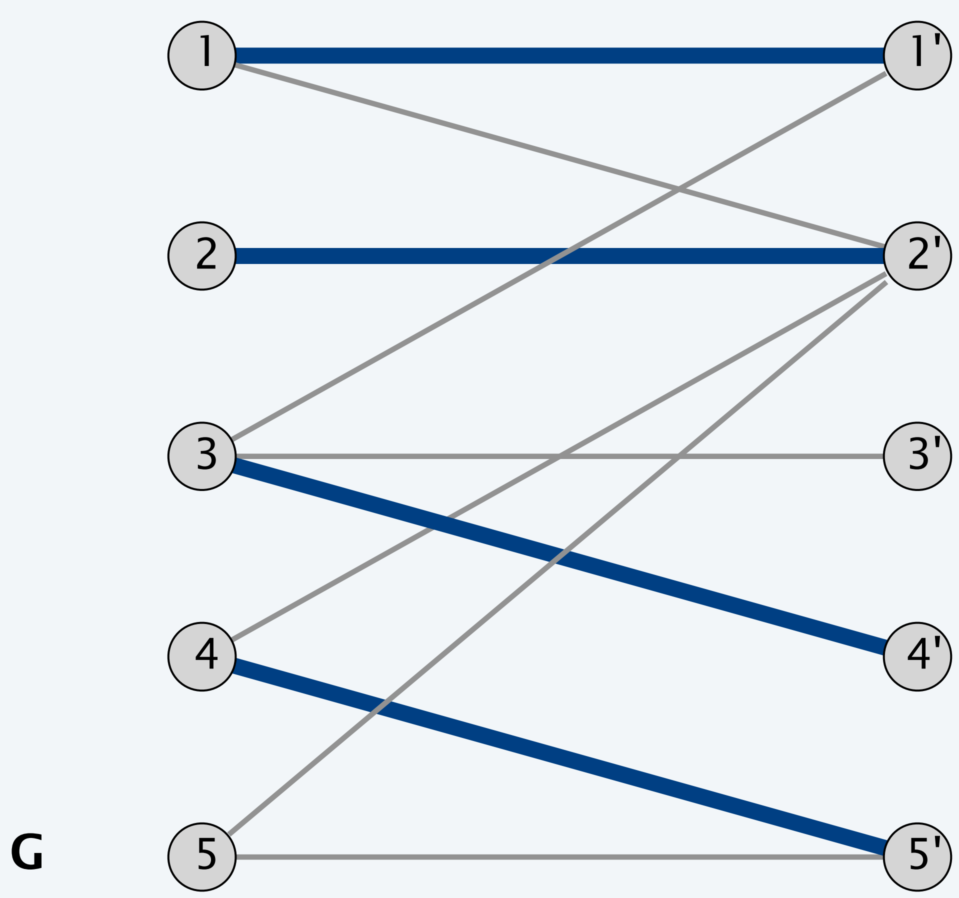
Max-flow formulation: proof of correctness

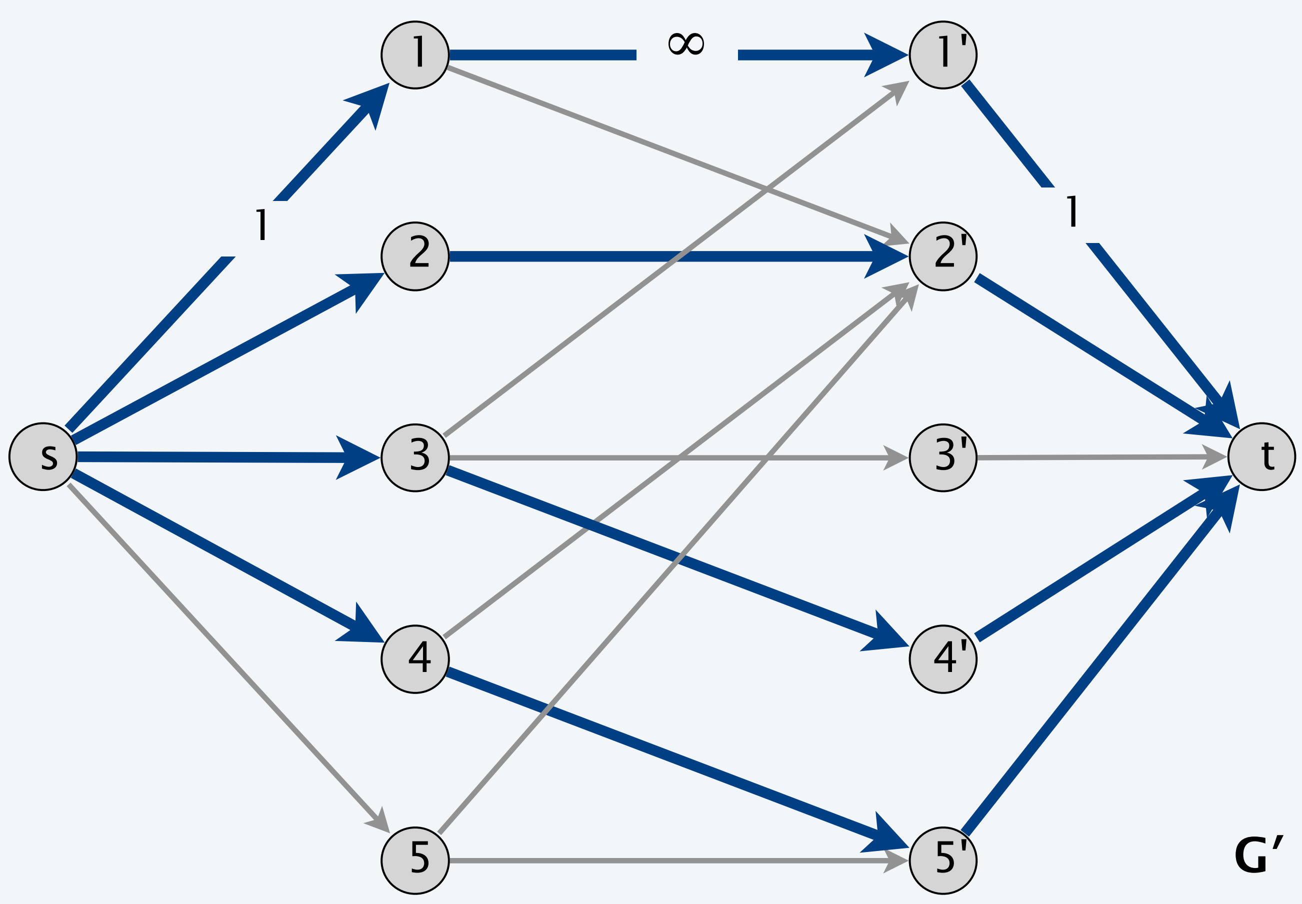
Theorem. 1–1 correspondence between matchings of cardinality k in G and integral flows of value k in G'.

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

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. ▪





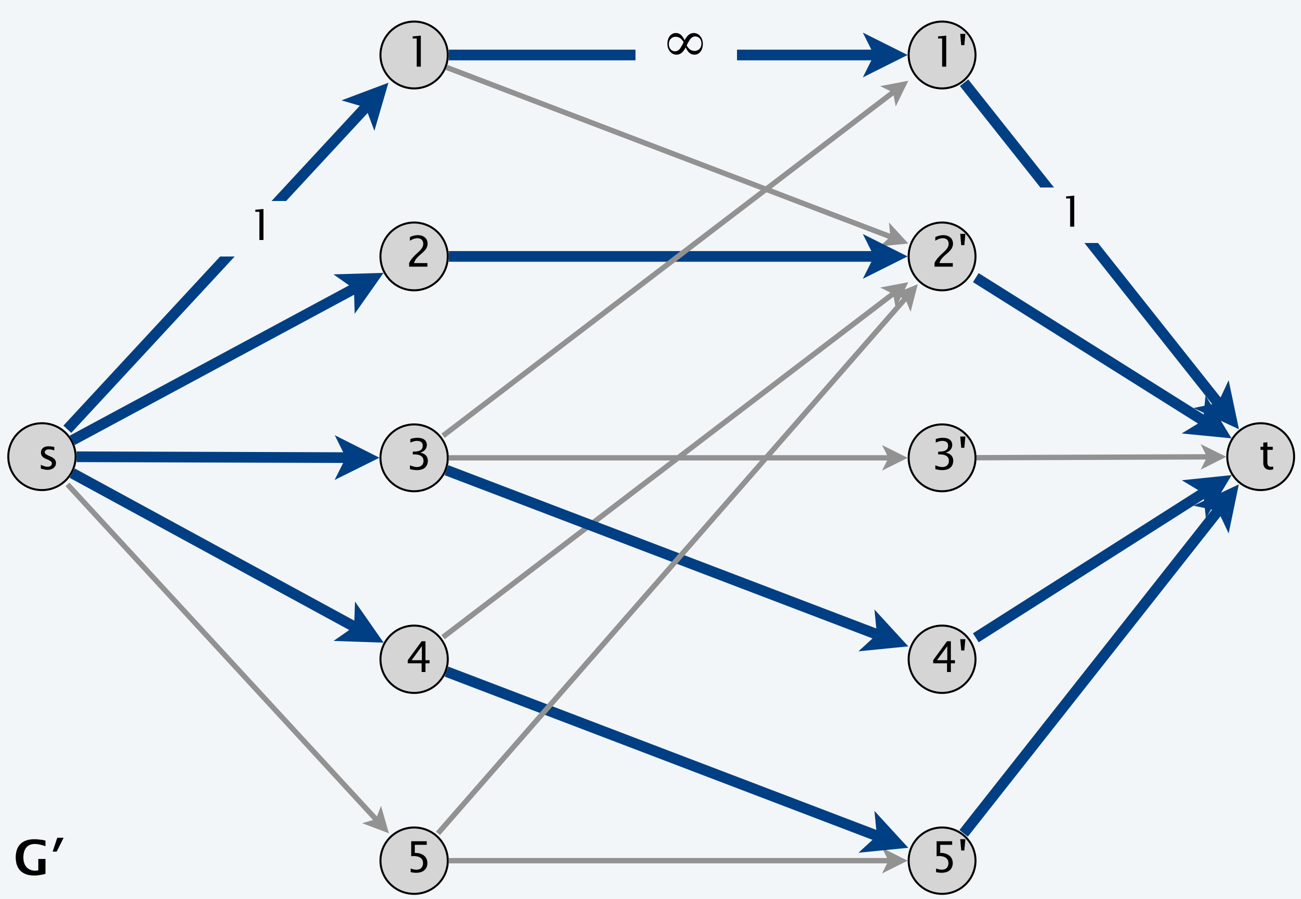
Max-flow formulation: proof of correctness

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

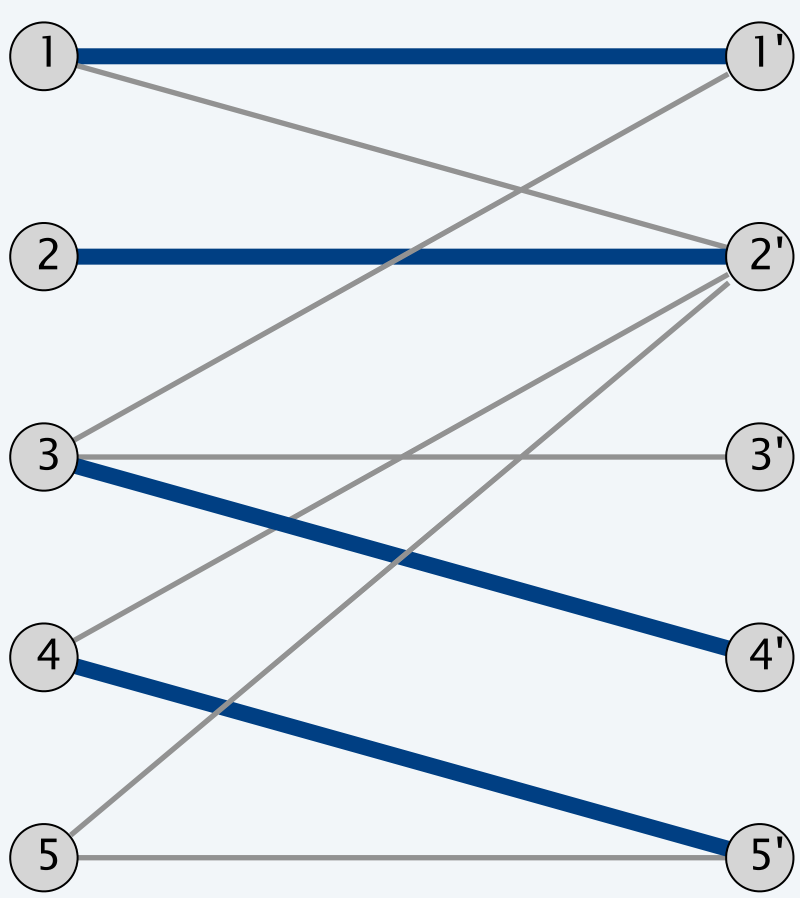
for each edge e: $f(e) \in \{ 0, 1 \}$

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



G′



G

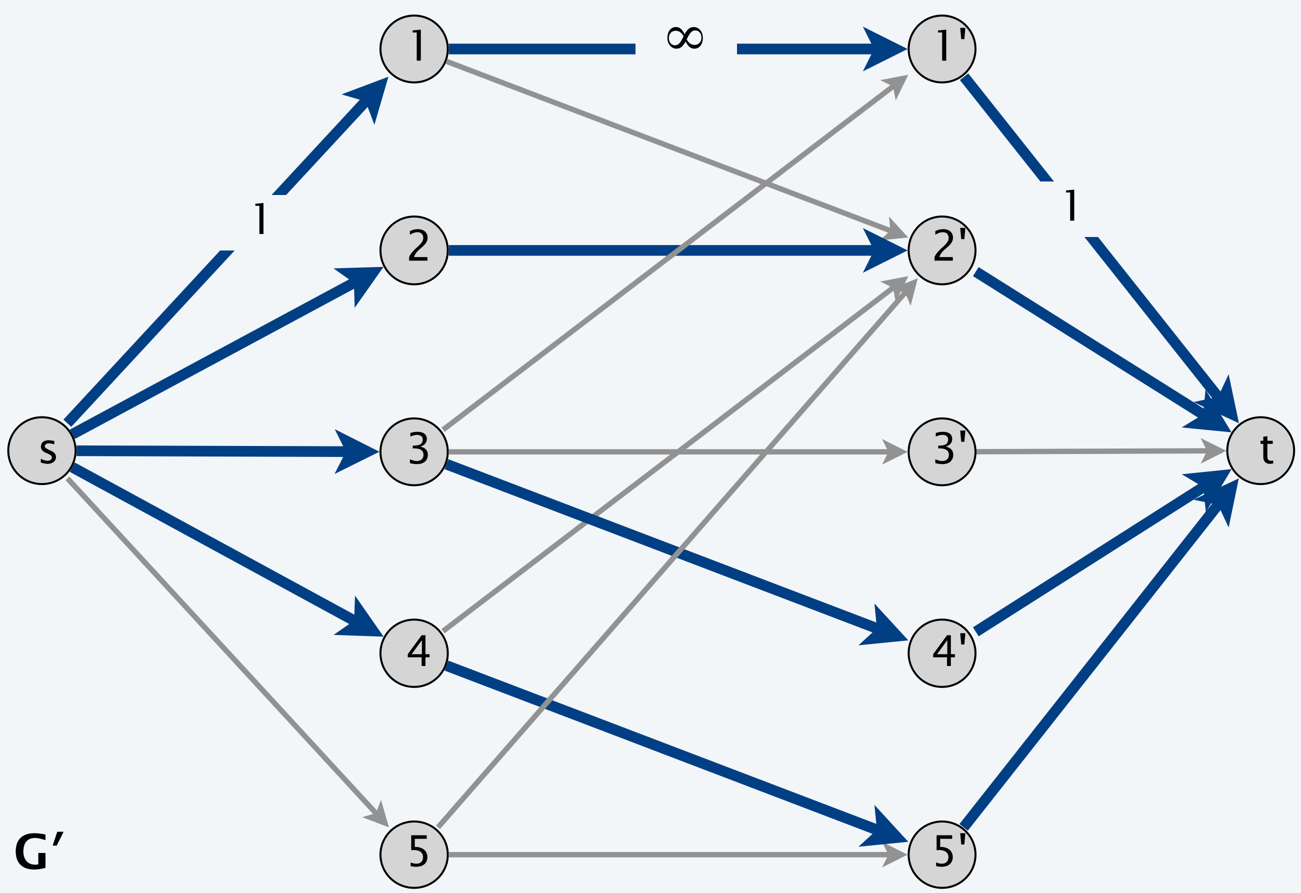
Max-flow formulation: proof of correctness

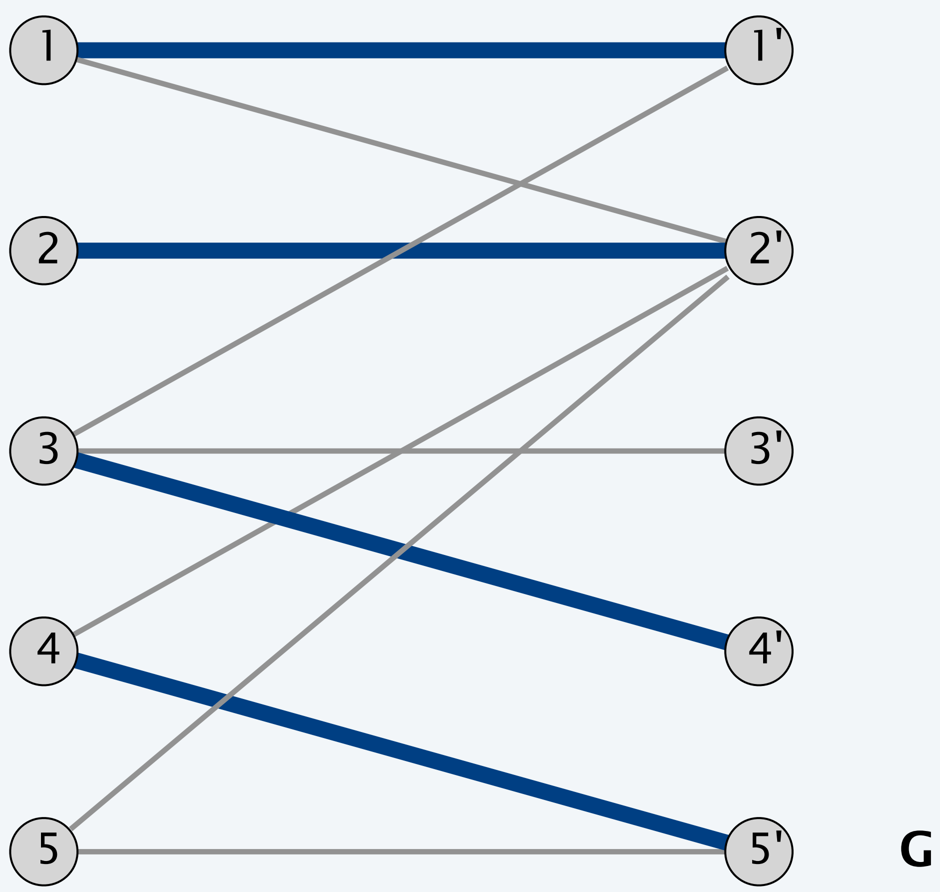
Theorem. 1–1 correspondence between matchings of cardinality k in G and integral flows of value k in G'.

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

Pf.

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





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

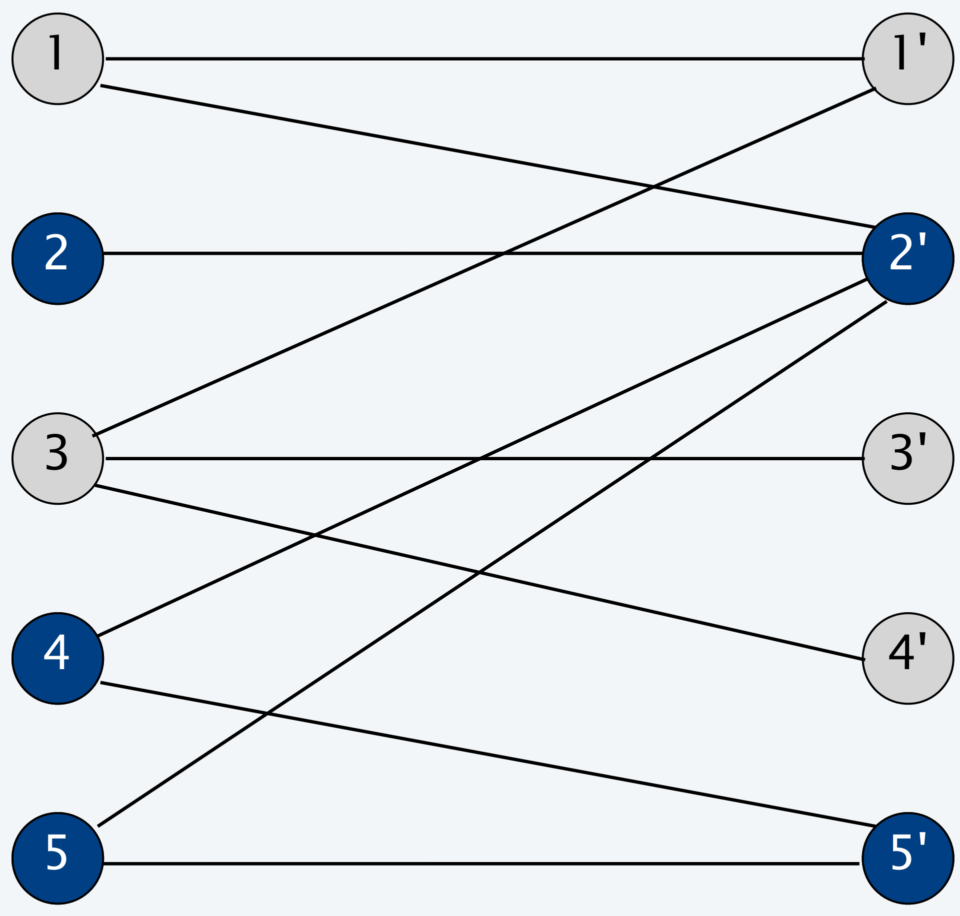
Notation. Let S be a subset of nodes, and let $N(S)$ be the set of nodes adjacent to nodes in S.

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

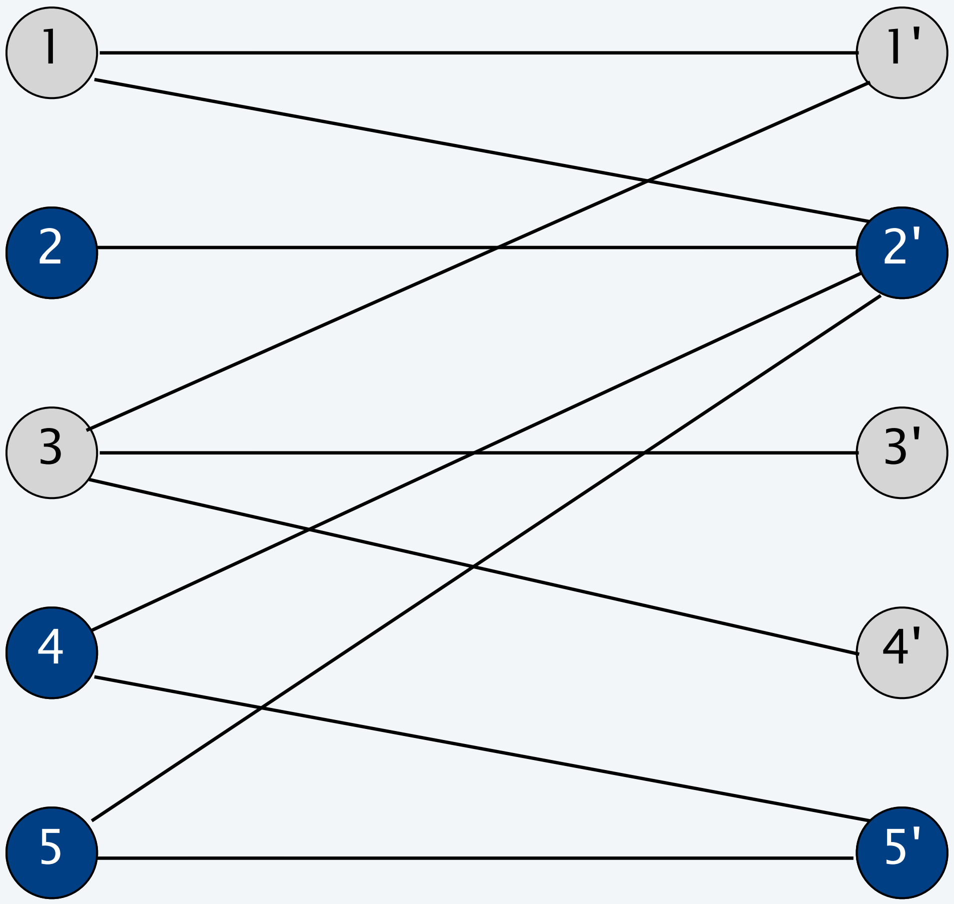
Hall's marriage theorem

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 $S \subseteq L$.

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

S = { 2, 4, 5 }

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

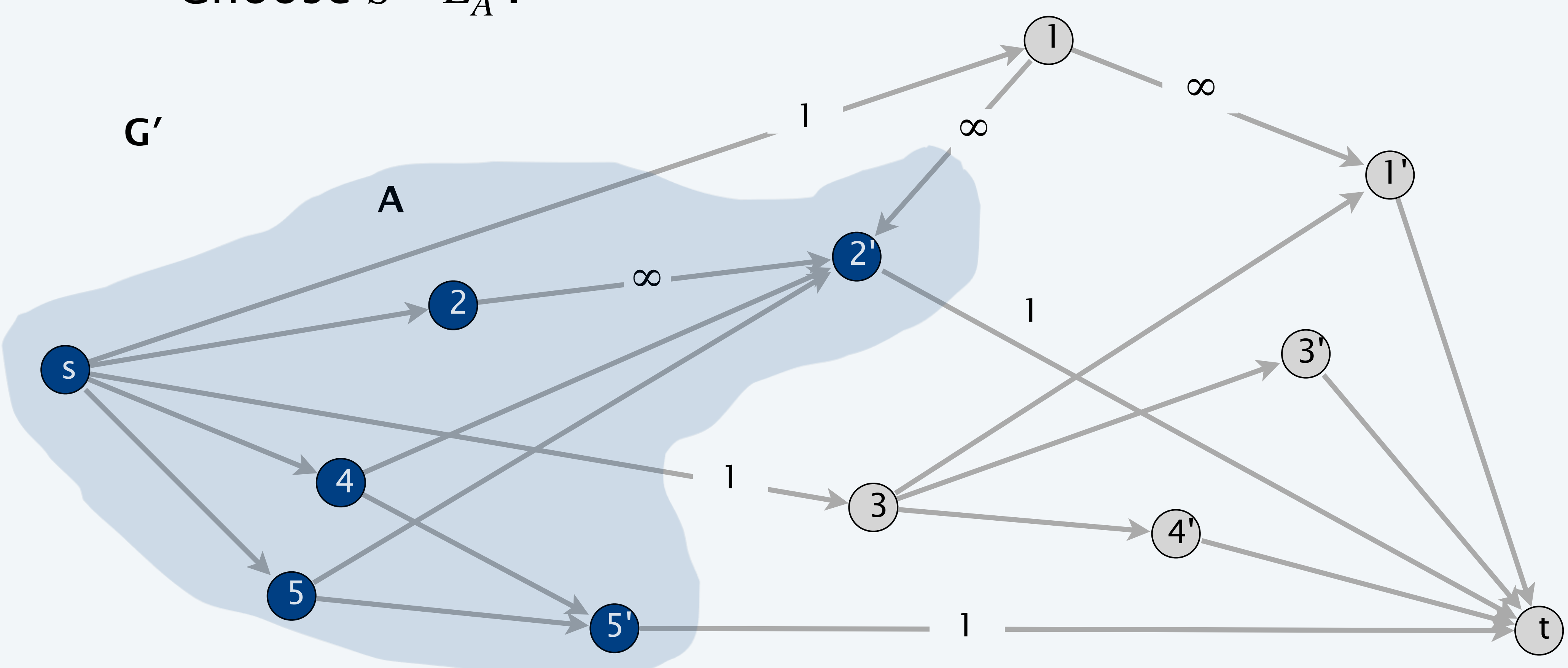


no perfect matching

Hall's marriage theorem

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 $L_A = L \cap A$, $L_B = L \cap B$, $R_A = R \cap A$.
- $cap(A, B) = |L_B| + |R_A| \Rightarrow |R_A| < |L_A|$.
- Min cut can't use ∞ edges $\Rightarrow N(L_A) \subseteq R_A$.
- $|N(L_A)| \le |R_A| < |L_A|$.
- Choose $S = L_A$. ▪



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

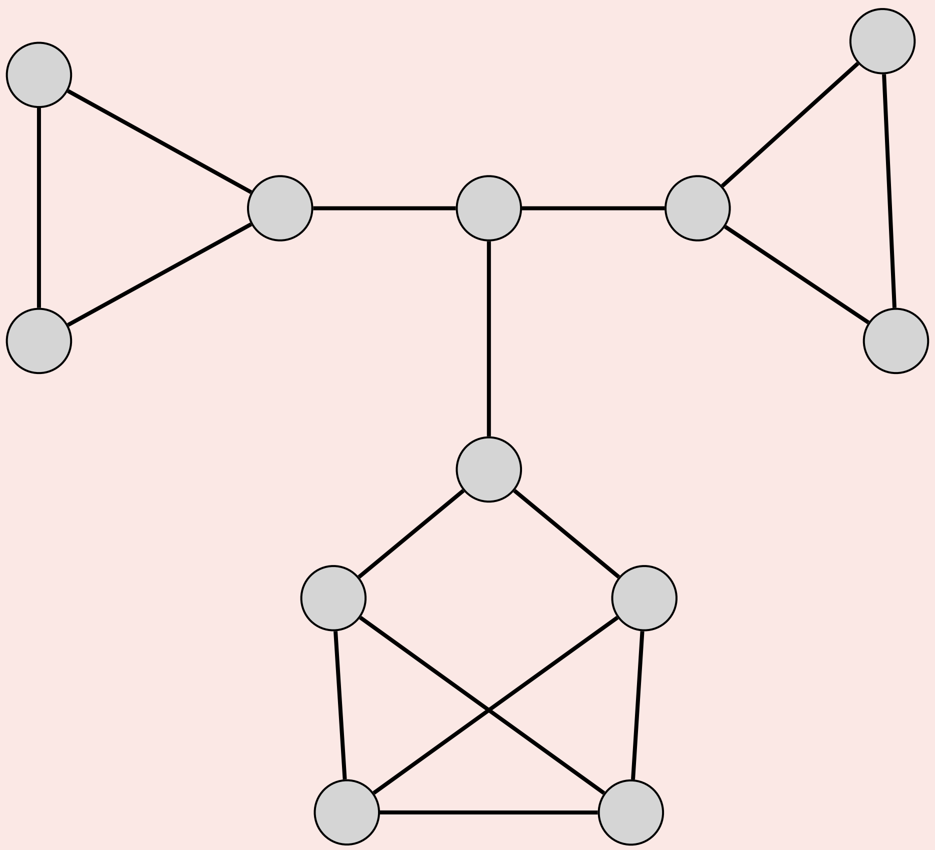
Which of the following are properties of the graph G = (V, E)?

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.

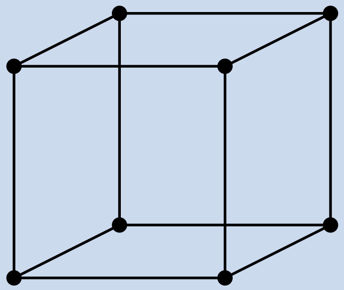
Hackathon problem

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.



2-regular bipartite graph