(Stable) Matching

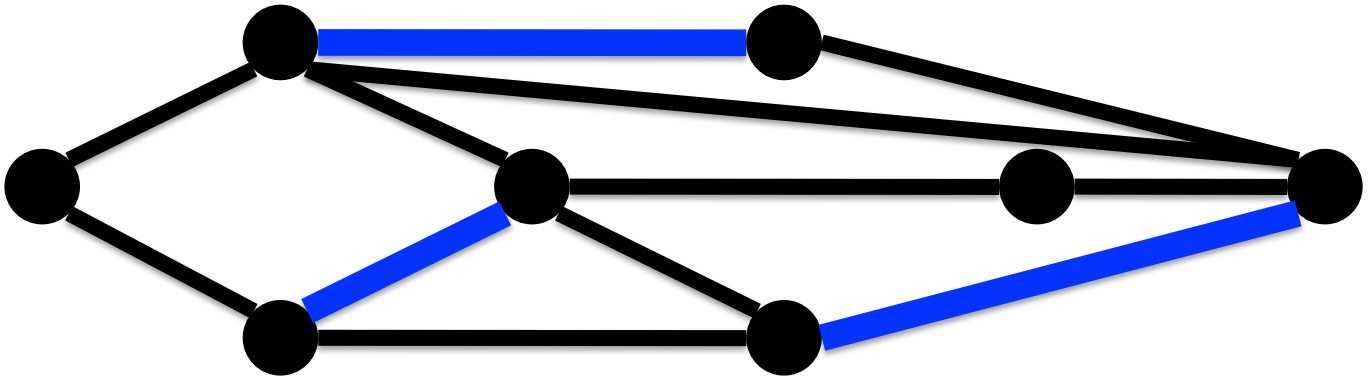
Or how to win a Nobel prize

Chapters 7 and 1

Marek Adamczyk, Stefano Leonardi,
Erik Jan van Leeuwen

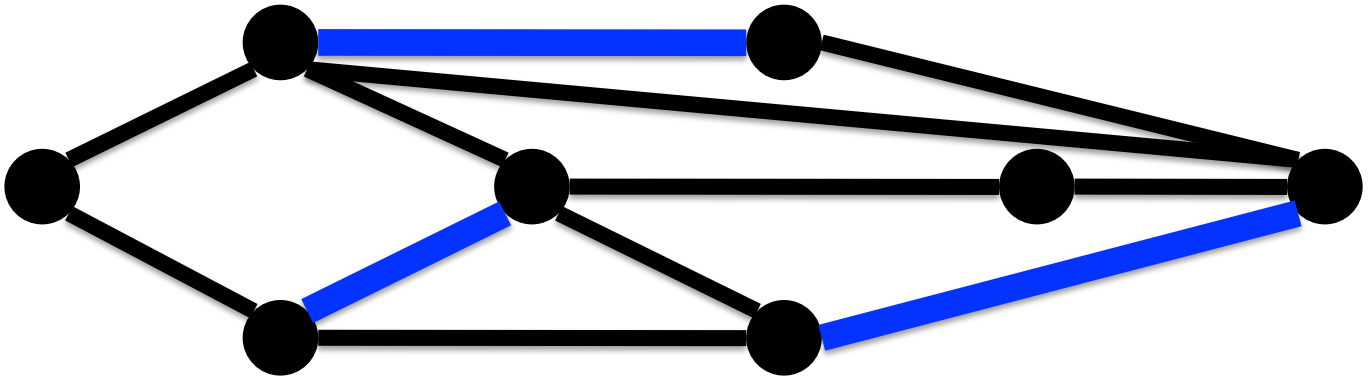
Recap Matching

- Set *M* of edges such that each node in at most one edge of *M*



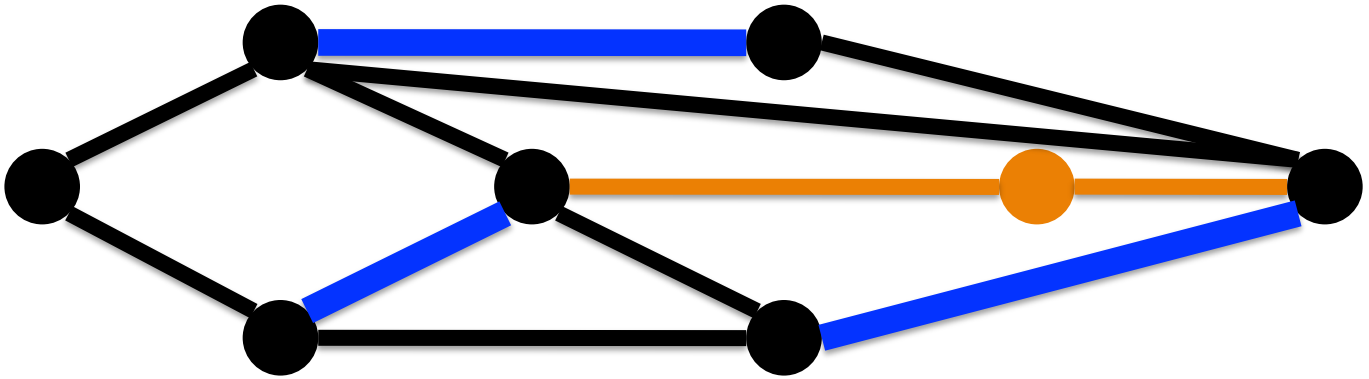
Recap Matching

- Set *M* of edges such that each node in at most one edge of *M*
- **Maximal** if no edge can be added to *M*



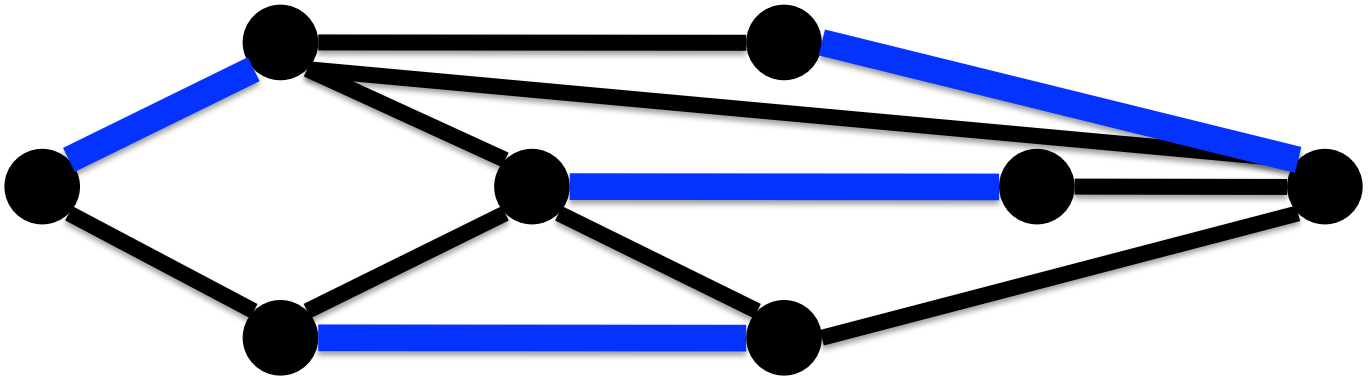
Recap Matching

- Set *M* of edges such that each node in at most one edge of *M*
- **Maximal** if no edge can be added to *M*



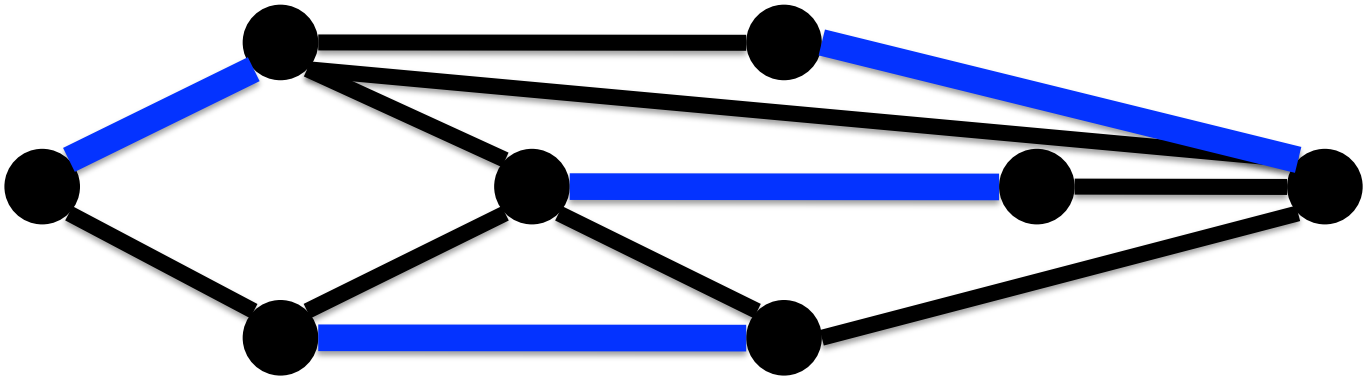
Recap Matching

- Set *M* of edges such that each node in at most one edge of *M*
- **Maximum** if no matching of larger size exists



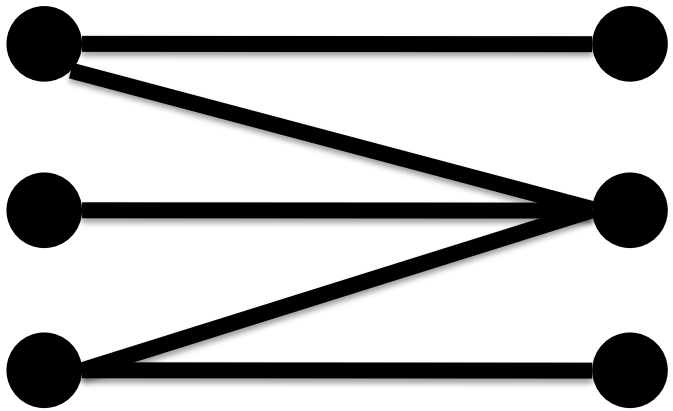
Recap Matching

- Set *M* of edges such that each node in at most one edge of *M*
- **Perfect** if each node is in exactly one edge of M



Recap Matching

- Maximum matching in **bipartite network**: use maximum flow, or direct algorithm (Hopcroft-Karp)



Matching in general

- How to compute a maximum matching in a general network?

Matching in general

- How to compute a maximum matching in a general network?
- Polynomial time using Edmond's blossom algorithm
  - Resembles algorithm for bipartite case
  - Curious? Wikipedia 'Blossom algorithm'

Matching with costs

- Suppose putting edge *e* in *M* has cost *c(e)*. Can we find matching with a maximum #edges, but minimum cost?

Matching with costs

- Suppose putting edge *e* in *M* has cost *c(e)*. Can we find matching with a maximum #edges, but minimum cost?
- Bipartite: minimum-cost flows
- General: more complex

Applications of Matching

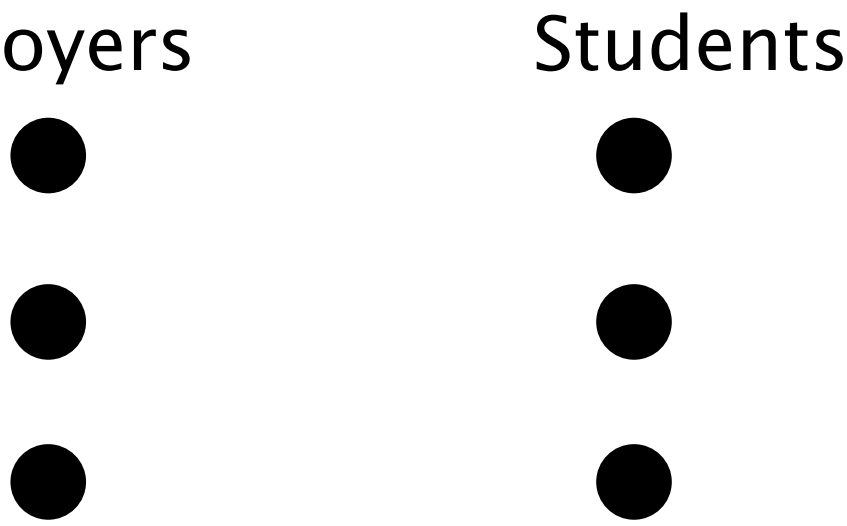
Enough with the theory

Apply bipartite matching

- Great for **assignment** problems

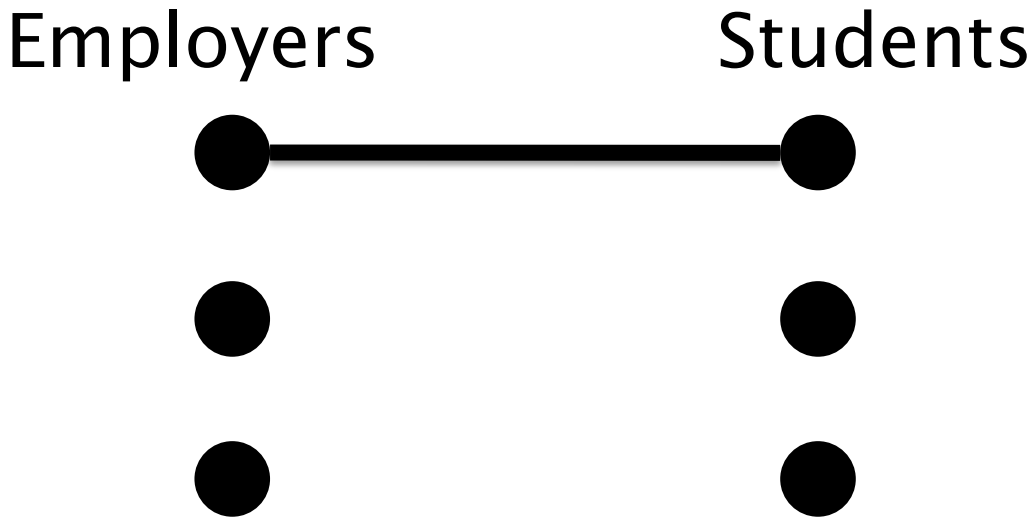
Apply bipartite matching

- Great for **assignment** problems
- Example: summer jobs



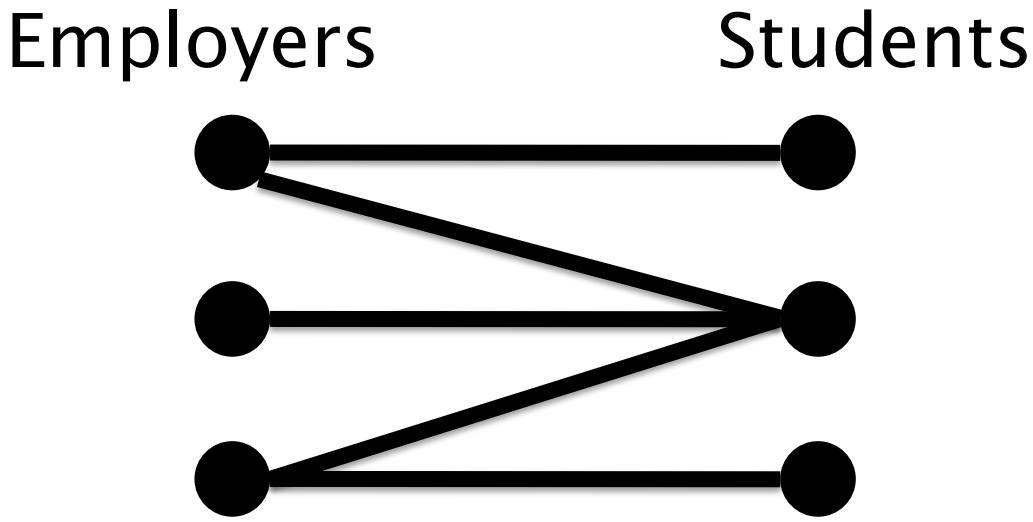
Apply bipartite matching

- Great for **assignment** problems
- Example: summer jobs



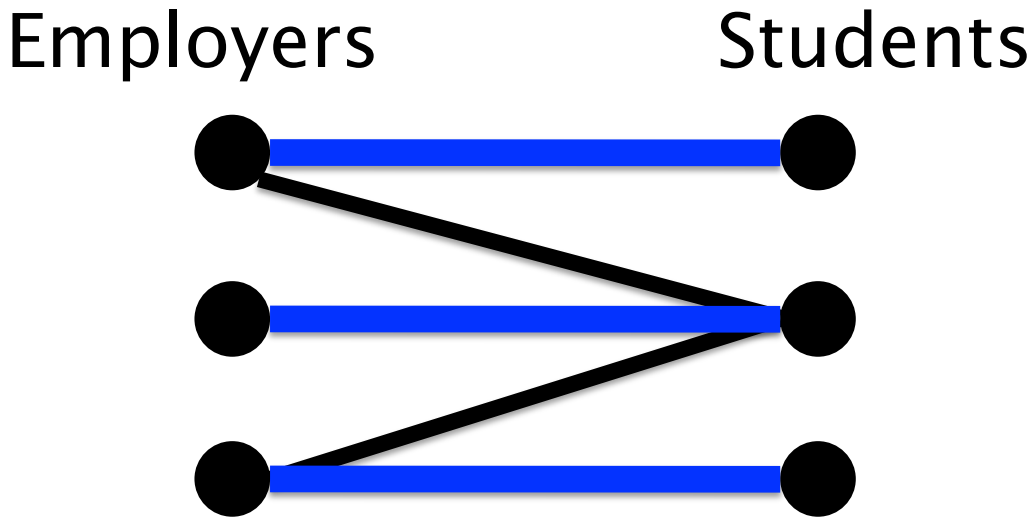
Apply bipartite matching

- Great for **assignment** problems
- Example: summer jobs



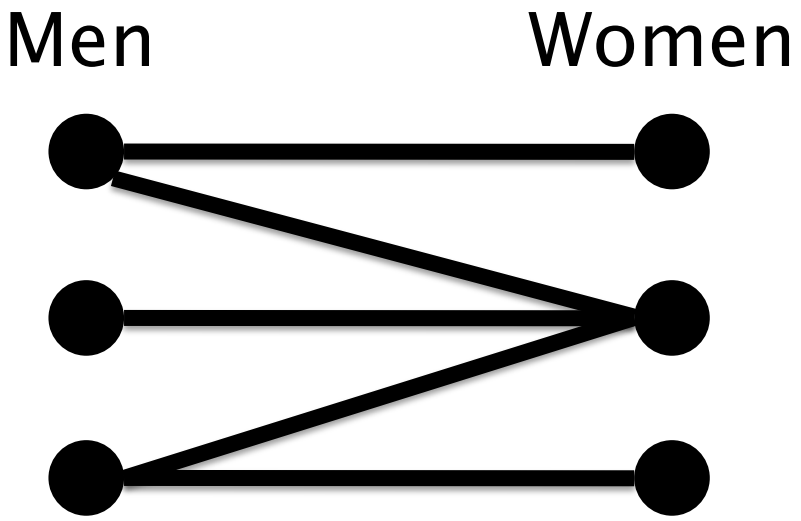
Apply bipartite matching

- Great for **assignment** problems
- Example: summer jobs



Apply bipartite matching

- Great for **assignment** problems
- Example: marriage (**dating**)



Stable Matching

Setup

- *n* men, *n* women, send *n* pairs on a date, paid by dating service
- Idea 1: Find any pairing

Setup

- *n* men, *n* women, send *n* pairs on a date, paid by dating service
- ~~Idea 1: Find any pairing~~

Setup

- Men rank the women
  - Man m orders $w_1 \leq w_2 \leq \ldots \leq w_n$
  - Man m' orders $w_2 \leq w_1 \leq w_4 \leq \ldots$
- Similarly, women rank all the men
- m–w and w'–m': what if w' prefers m to m' and m prefers w' to w? **Instable** matching

Stable Matching

- Idea 2: find a **stable matching**
  - If *m* matched to *w*, then there is no *w'* such that *m* prefers *w'* and *w'* prefers *m* to date *m'* of *w'*
- No incentive to switch dates
  - Related field of **game theory**

Stable Matching

- Can we find a stable matching in polynomial time?

Stable Matching

- Can we find a stable matching in polynomial time?
- Yes! Using Gale-Shapley algorithm
  - L.S. Shapley (together with A. Roth) received **2012 Nobel prize** in economics

"for the theory of stable allocations and the practice of market design"

Gale–Shapley algorithm

- **while** exists *m* without date and *m* has not asked every woman
  - Let *w* be highest-rank woman that *m* has not yet asked
  - If *w* is free or *w* prefers *m* to her current date *m'*, then *m* and *w* date and *m'* becomes dateless
  - Otherwise (if *w* prefers current date to *m*), goto **while**

Analysis: time

- Algorithm finishes in n^2 iterations
- Worst-case: every man tries every woman
- **Polynomial time**

Analysis: matching

- Once woman gets a date, she has a date during rest of algorithm
- If *m* has no date, then *m* has not asked every woman yet
  - Otherwise every woman has a date already, so *m* has a date
- Algorithm gives everyone a date

Analysis: stability

- Once woman gets a date, she has a date during rest of algorithm
- Date of a woman only improves

Analysis: stability

- Let m–w, m'–w', but m prefers w' to w and w' prefers m to m'
- Did m ask w' and if so before w?
  - No, then m prefers w to w', contradiction.
  - Yes and w' accepted. Since dates for women improve, m' is preferred to m.
  - Yes and w' declined. Then date of w' at that time preferred to m, so m' is preferred to m.

Is stable matching good?

- Every woman only gets better dates
- Every man only get worse dates

Is stable matching good?

- Might be many stable matchings, algorithm returns *M*
- *M* is ideal for men: in any stable matching man is matched to less preferred woman than in *M*
- *M* is terrible for women: woman is always matched with worst man

End of Matchings

Now You!

- Exercise 7.7: given n clients with a position, given k base stations with a position, a range R, and a maximum load L.
- Give polynomial-time algorithm that decides if every client can connect to a base station

Now You!

- A network is ***k*-regular** if every node has degree exactly *k*
- Show that any *k*-regular, bipartite network has a perfect matching