DIS 2A

STABLE MATCHING

given n jobs, n candidates, and their preferences, how can we best match them?

TERMS

→ stable matching instance

a particular set of n jobs, n candidates, and their preference lists

→ a stable matching

a matching of jobs to candidates that does not contain any rogue couples

a matching is unstable if it contains at least one rogue couple

→ stable matching algorithm

the propose-and-reject algorithm

MORE TERMS

→ **rogue couple**

a job-candidate pair that prefers each other over who they're currently matched with.

J	$C > C'$
J'	$C' > C$

C	$J > J'$
C'	$J' > J$

$\{(J', C), (J, C')\}$

(J, C) is a rogue couple

→ **job-optimal matching**

each job is matched with the <u>best</u> possible candidate it could get out of all stable matchings

→ **job-pessimal matching**

each job is matched with the <u>worst</u> possible candidate it could get out of all stable matchings

candidate-optimal/pessimal defined similarly

PROPOSE-AND-REJECT

on each day:

1. jobs propose to the most preferred candidate still on their list

2. candidates put their favorite job offer "on a string" (like tentatively accepting) and reject the others

3. jobs cross off candidates who rejected them.

repeat until termination, when no offers are rejected.

→ always halts

→ produces a job-optimal stable matching

→ improvement lemma

if J proposes to C on day k, every day thereafter C has an offer at least as good as J

STABLE MATCHING PROOF TIPS

- pick out specific jobs and candidates to reason about (J, J', C, C', etc.)
- contradiction is your friend! flip the claim and show that it violates properties of the algorithm.
  - stability
  - optimality
  - halting
  - improvement lemma