CS 331, Fall 2025

Lecture 10 (9/29)

Today: — Stable matching

Stable matching (Part IV, Section 5)

Setup: n applicants {Alice, Bob, ...}

n job openings {Google, Apple, ...}

Input: Preference lists

$a: \alpha > \gamma > \beta$

$b: \gamma > \alpha > \beta$

$c: \alpha > \beta > \gamma$

$\alpha: b > a > c$

$\beta: c > a > b$

$\gamma: a > b > c$

Output: Stable matching

(a, α) (b, β) (c, γ)

unstable

(b, α) (c, β) (a, γ)

stable

What's the difference?

(b, α) unstable pair:

b prefers $\alpha > \beta$

α prefers $b > a$

$\boxed{\text{backroom deal...}}$

Stability notion protects against "first-order" deviations, similar concept to Nash equilibrium

How to design a greedy algo?

Idea: fix instability (like inversions)

Issue: cycles

$$\begin{matrix}(a,\alpha)\\(b,\beta)\\(c,\gamma)\end{matrix} \xrightarrow{(b,\alpha)} \begin{matrix}(b,\alpha)\\(a,\beta)\\(c,\gamma)\end{matrix} \xrightarrow{(b,\gamma)} \begin{matrix}(c,\alpha)\\(a,\beta)\\(b,\gamma)\end{matrix}$$

$$\xrightarrow{(a,\gamma)} \begin{matrix}(c,\alpha)\\(b,\beta)\\(a,\gamma)\end{matrix} \xrightarrow{(a,\alpha)} \begin{matrix}(a,\alpha)\\(b,\beta)\\(c,\gamma)\end{matrix}$$

Just because (b,α) unstable doesn't mean we should pair (a,β)!

Key idea: job offers/reneges (unmatch a matched pair)

- Maintain pool of temporary matches
- If (a, α) in the pool, a prefers β to α, and β makes offer to a, can renege and pool $\leftarrow$ pool $\cup (a, \beta) \setminus (a, \alpha)$

Gale-Shapley also

- Hugely influential in practice
  - National Resident Matching Program
  - Faculty recruiting
  - Public schools in NYC, Boston
  - Assignments in US Navy
  - Kidney exchange programs
- Nobel Prize in Economics, 2012

StableMatching$(\{A_a\}_{a\in[n]}, \{J_\alpha\}_{\alpha\in[n]})$:

$M \Leftarrow \emptyset$, $i_\alpha \Leftarrow 1 \quad \forall \alpha \in [n]$

While $\exists$ unmatched job α:

$a \leftarrow J_\alpha[i_\alpha]$ // favorite applicant w/o reject

If a unmatched: $M \Leftarrow M \cup \{(a,\alpha)\}$

Elif a prefers α to β (current match):

$M \Leftarrow M \setminus \{(a,\beta)\} \cup \{(a,\alpha)\}$

i_β++ // reneged

Else: i_α++ // rejected

Return M

After $O(n^2)$ preprocessing (index lookups, etc.)

Can implement each iter in $O(1)$ time:

Maintain M as Array indexed by a

Runtime: Potential method.

Define function Φ that captures algo progress.

Our potential:

$$\Phi = |M| + \sum_{\alpha} i_{\alpha}$$

Algo ends when $|M| = n$, so

$$\Phi \geq n + n^2 \Rightarrow \text{termination}$$

Every iter: • pointer i_α grows

OR

• $|M|$ grows

Φ grows!

Total: $O(n^2)$ linear time!

Correctness: Perfect matching

If $|M| \neq n$, the loop continues!

Stable matching

- Let $\{(a, \alpha), (b, \beta)\} \in M$
- Suppose (a, β) unstable
- If a had offer from β then would not be w/ α

(a, α) $\qquad\qquad$ $(a, \geq \beta)$

our world $\qquad\qquad$ if β offered to a

- But β likes $a > b$, so must have offered first.

Hence, no unstable pairs.

Structural fact: Outcome always same, regardless of tiebreaking.

$$\left.\begin{array}{l}\text{Say } a \text{ feasible for } d \\ \quad d \text{ feasible for } a\end{array}\right\} \text{ if } (a, d) \in M \text{ stable}$$

(for some choice of stable M)

Key Claims

1: Every job d gets best feasible a

2: Every applicant a gets worst feasible d

Uniqueness of M follows immediately.

(No ties in preference lists)

Proof of claim 1: Consider first time in G-S where best feasible a for α rejects for some other job β

Because a feasible, $\exists$ stable M' pairing

(a, α) and (b, β)

- a prefers β to α
- β prefers a to b ← feasible for β... Can't have rejected yet when β makes a offer!

Unstable! $\Rightarrow\Leftarrow$

Proof of claim 2: Suppose G-S pairs (a, α) but some other stable M' pairs (a, β) ← worse feasible job for a

(b, α)

- a prefers α to β
- α prefers a to b (Proof: Claim 1 says so.)

Unstable! $\Rightarrow\Leftarrow$