CSE 421: Introduction to Algorithms

Stable Matching

Yin-Tat Lee
This Course

Talk about how to solve problems using computers, aka, algorithms.

Goal:
• Learn all basic techniques to design algorithms
• How to analyze the runtime
• Understand some problems are difficult

Grading Scheme
• Homework ~ 50%
  Weekly homework due on Wed before the class
• Midterm ~ 15-20%
• Final ~ 30-35%

Course textbook
Where to get help?

• Ask questions in the class!
• Read the textbook!
• Canvas: Online discussion forum.
• Office hours:
• Myself: M 2:30-4:00 in CSE 562

<table>
<thead>
<tr>
<th>TA</th>
<th>Office hours (from Apr 2 - June 1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Benjamin Jones</td>
<td>Mon 10:30am-11:20am</td>
</tr>
<tr>
<td>Mathew Luo</td>
<td>Tue 11:30am - 12:20pm</td>
</tr>
<tr>
<td>Angli Liu</td>
<td>Tue 2:30pm-3:20pm</td>
</tr>
<tr>
<td>Thomas Merth</td>
<td>Wed 10:30am-11:20am</td>
</tr>
<tr>
<td>Aditya Saraf</td>
<td>Thu 12:30pm-1:20pm</td>
</tr>
<tr>
<td>Guanghao Ye</td>
<td>Thu 2:30pm-3:20pm</td>
</tr>
<tr>
<td>Sai Nimmagadda</td>
<td>Fri 11:30am-12:20pm</td>
</tr>
</tbody>
</table>

CSE 007
(Starting from Apr)

Website: cs.washington.edu/421
Yin-Tat’s Research

I work in theoretical computer science.

Among others, I gave the (theoretically) fastest algorithms for
• Linear Programs
• Network flow
Two topics that we will discuss in this course.

Some problems in this course are still a hot area of research!

Poll
Stable Matching Problem

Given \( n \) men and \( n \) women, find a “stable matching”.
- We know the preference of all people.

Men’s Preference Profile

<table>
<thead>
<tr>
<th>Favorite</th>
<th>1st</th>
<th>2nd</th>
<th>3rd</th>
</tr>
</thead>
<tbody>
<tr>
<td>Xavier</td>
<td>Amy</td>
<td>Brenda</td>
<td>Claire</td>
</tr>
<tr>
<td>Yuri</td>
<td>Brenda</td>
<td>Amy</td>
<td>Claire</td>
</tr>
<tr>
<td>Zoran</td>
<td>Amy</td>
<td>Brenda</td>
<td>Claire</td>
</tr>
</tbody>
</table>

Women’s Preference Profile

<table>
<thead>
<tr>
<th>Favorite</th>
<th>1st</th>
<th>2nd</th>
<th>3rd</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amy</td>
<td>Yuri</td>
<td>Xavier</td>
<td>Zoran</td>
</tr>
<tr>
<td>Brenda</td>
<td>Xavier</td>
<td>Yuri</td>
<td>Zoran</td>
</tr>
<tr>
<td>Claire</td>
<td>Xavier</td>
<td>Yuri</td>
<td>Zoran</td>
</tr>
</tbody>
</table>
Stable Matching

Perfect matching:
• Each man gets exactly one woman.
• Each woman gets exactly one man.

Stability: no incentive to exchange
• an unmatched pair $m-w$ is unstable
• if man $m$ and woman $w$ prefer each other to current partners.
Stable Matching

Perfect matching:
- Each man gets exactly one woman.
- Each woman gets exactly one man.

Stability: no incentive to exchange
- an unmatched pair $m-w$ is unstable
- if man $m$ and woman $w$ prefer each other to current partners.

Stable matching: perfect matching with no unstable pairs.

Stable matching problem: Given the preference lists of $n$ men and $n$ women, find a stable matching if one exists.
**Example**

**Question.** Is assignment X-C, Y-B, Z-A stable?

<table>
<thead>
<tr>
<th>Favorite</th>
<th>1st</th>
<th>2nd</th>
<th>3rd</th>
</tr>
</thead>
<tbody>
<tr>
<td>Xavier</td>
<td>Amy</td>
<td>Brenda</td>
<td>Claire</td>
</tr>
<tr>
<td>Yuri</td>
<td>Brenda</td>
<td>Amy</td>
<td>Claire</td>
</tr>
<tr>
<td>Zoran</td>
<td>Amy</td>
<td>Brenda</td>
<td>Claire</td>
</tr>
</tbody>
</table>

*Men's Preference Profile*

<table>
<thead>
<tr>
<th>Favorite</th>
<th>1st</th>
<th>2nd</th>
<th>3rd</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amy</td>
<td>Yuri</td>
<td>Xavier</td>
<td>Zoran</td>
</tr>
<tr>
<td>Brenda</td>
<td>Xavier</td>
<td>Yuri</td>
<td>Zoran</td>
</tr>
<tr>
<td>Claire</td>
<td>Xavier</td>
<td>Yuri</td>
<td>Zoran</td>
</tr>
</tbody>
</table>

*Women's Preference Profile*
Example

**Question.** Is assignment X-C, Y-B, Z-A stable?

**Answer.** No. Brenda and Xavier will hook up.

<table>
<thead>
<tr>
<th>Men's Preference Profile</th>
<th>Women's Preference Profile</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>favorite</strong> ↓</td>
<td><strong>least favorite</strong> ↓</td>
</tr>
<tr>
<td>1st</td>
<td>2nd</td>
</tr>
<tr>
<td>Xavier</td>
<td>Amy</td>
</tr>
<tr>
<td>Yuri</td>
<td>Brenda</td>
</tr>
<tr>
<td>Zoran</td>
<td>Amy</td>
</tr>
</tbody>
</table>
Question: Is assignment X-A, Y-B, Z-C stable?
Answer: Yes.

<table>
<thead>
<tr>
<th></th>
<th>Men's Preference Profile</th>
<th>Women's Preference Profile</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>favorite</strong></td>
<td>1st</td>
<td>1st</td>
</tr>
<tr>
<td>2nd</td>
<td>2nd</td>
<td>2nd</td>
</tr>
<tr>
<td>3rd</td>
<td>3rd</td>
<td>3rd</td>
</tr>
<tr>
<td><strong>Men</strong></td>
<td><strong>Women</strong></td>
<td><strong>Men</strong></td>
</tr>
<tr>
<td>Xavier</td>
<td>Amy</td>
<td>Yuri</td>
</tr>
<tr>
<td>Yuri</td>
<td>Brenda</td>
<td>Xavier</td>
</tr>
<tr>
<td>Zoran</td>
<td>Amy</td>
<td>Yuri</td>
</tr>
<tr>
<td></td>
<td>Brenda</td>
<td>Zoran</td>
</tr>
<tr>
<td></td>
<td>Claire</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Claire</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Existence of Stable Matchings

Question. Do stable matchings always exist?
Answer. Yes, but not obvious.

Stable roommate problem: (or “same-sex” stable matching)

2n people; each person ranks others from 1 to 2n-1.
Assign roommate pairs so that no unstable pairs.

<table>
<thead>
<tr>
<th></th>
<th>1st</th>
<th>2nd</th>
<th>3rd</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adam</td>
<td>B</td>
<td>C</td>
<td>D</td>
</tr>
<tr>
<td>Bob</td>
<td>C</td>
<td>A</td>
<td>D</td>
</tr>
<tr>
<td>Chris</td>
<td>A</td>
<td>B</td>
<td>D</td>
</tr>
<tr>
<td>David</td>
<td>A</td>
<td>B</td>
<td>C</td>
</tr>
</tbody>
</table>

A-B, C-D ⇒ B-C unstable
A-C, B-D ⇒ A-B unstable
A-D, B-C ⇒ A-C unstable

So, Stable matchings do not always exist for stable roommate problem.
Propose-And-Reject Algorithm \cite{Gale-Shapley’62}

Initialize each person to be free.
while (some man is free and hasn't proposed to every woman) {
    Choose such a man \( m \)
    \( w = \text{1st woman on } m \text{'}s list to whom } m \text{ has not yet proposed} \)
    if (\( w \) is free)
        assign \( m \) and \( w \) to be engaged
    else if (\( w \) prefers \( m \) to her fiancé \( m' \))
        assign \( m \) and \( w \) to be engaged, and \( m' \) to be free
    else
        \( w \) rejects \( m \)
}
Main Properties of the algorithm

**Observation 1:** Men propose to women in decreasing order of preference.

**Observation 2:** Once a woman is matched, she never becomes unmatched; she only "trades up."
What do we need to prove?

• The algorithm ends.
  How many iterations it takes?

• The output is correct.
  It find a perfect matching that is stable.
Proof of Correctness: Termination

Each step, a man proposed to a new woman.

There are $n \times n = n^2$ possible man-to-woman proposals.

Therefore, it takes at most $n^2$ iterations.

One strategy to bound # iterations is to find a measure of progress.

$n(n-1) + 1$ proposals required
Claim. All men and women get matched.

Proof. (by contradiction)

Suppose, for sake of contradiction, that Zoran is not matched upon termination of algorithm.

Then some woman, say Amy, is not matched upon termination.

(Observation 2: once women matched, they never becoming unmatched.) Amy was never proposed to.

But, Zoran proposes to everyone, since he ends up unmatched.
Proof of Correctness: Stability

Claim. No unstable pairs.

Proof. (by contradiction)

Suppose \textbf{A-Z} is an unstable pair: each prefers each other to the partner in Gale-Shapley matching.

Case 1: Z never proposed to A.
\[ \Rightarrow \text{Z prefers his GS partner to A.} \]
\[ \Rightarrow \text{A-Z is stable.} \]

Case 2: Z proposed to A.
\[ \Rightarrow \text{A rejected Z (right away or later)} \]
\[ \Rightarrow \text{A prefers her GS partner to Z.} \]
\[ \Rightarrow \text{A-Z is stable.} \]

In either case \textbf{A-Z} is stable, a contradiction.
Summary

• **Stable matching problem:** Given $n$ men and $n$ women, and their preferences, find a stable matching.

• **Gale-Shapley algorithm:** Guarantees to find a stable matching for any problem instance.

• **Q:** How to implement GS algorithm efficiently?

• **Q:** If there are multiple stable matchings, which one does GS find?

• **Q:** How many stable matchings are there? 18
Why this problem is important?

In 1962, Gale and Shapley published the paper “College Admissions and the Stability of Marriage” to “The American Mathematical Monthly”
Why this problem is important?

Alvin Roth modified the Gale-Shapley algorithm and apply it to

• National Residency Match Program (NRMP), a system that assigns new doctors to hospitals around the country. (90s)

• Public high school assignment process (00s)

• Helping transplant patients find a match (2004) (Saved >1,000 people every year!)

Reference: https://medium.com/@UofCalifornia/how-a-matchmaking-algorithm-saved-lives-2a65ac448698
Why this problem is important?

Shapley and Roth got the Nobel Prize (Economic) in 2012. (David Gale passed away in 2008.)

Some of the problems in this course may seem obscure or even pointless when it was discovered.

But their abstraction allows for variety of applications.