CSE 101
Design and Analysis of Algorithms

Syllabus: www.cse.ucsd.edu/classes/sp05/cse101)

Reading: Kleinberg, chapter 1
Discussion session (Day 2.5) will cover
Kleinberg, Chapters 2 & 3
Homework 0: due Day 4
Design and Analysis of Algorithms

Design: Methods for creating efficient algorithms
- Learn how to apply them to new problems

Analysis: The theoretical study of computer-program performance and resource usage

Why study?
- Algorithms help understand scalability.
- Performance helps understand what is feasible and impossible
- Algorithmic mathematics provides a language for talking about program behavior
- Speed is fun!
1.1 A First Problem: Stable Matching
Matching Residents to Hospitals

**Goal.** Given a set of preferences among hospitals and medical school students, design a self-reinforcing admissions process.

**Unstable pair:** applicant \( x \) and hospital \( y \) are unstable if:
- \( x \) prefers \( y \) to its assigned hospital.
- \( y \) prefers \( x \) to one of its admitted students.

**Stable assignment.** Assignment with no unstable pairs.
- Natural and desirable condition.
- Individual self-interest will prevent any applicant/hospital deal from being made.
Stable Matching Problem

**Goal.** Given n men and n women, find a "suitable" matching.
- Participants rate members of opposite sex.
- Each man lists women in order of preference from best to worst.
- Each woman lists men in order of preference from best to worst.

**Men's Preference Profile**

<table>
<thead>
<tr>
<th>favorite</th>
<th>least favorite</th>
</tr>
</thead>
<tbody>
<tr>
<td>1st</td>
<td>2nd</td>
</tr>
<tr>
<td>Xavier</td>
<td>Amy</td>
</tr>
<tr>
<td>Yancey</td>
<td>Bertha</td>
</tr>
<tr>
<td>Zeus</td>
<td>Amy</td>
</tr>
</tbody>
</table>

**Women's Preference Profile**

<table>
<thead>
<tr>
<th>favorite</th>
<th>least favorite</th>
</tr>
</thead>
<tbody>
<tr>
<td>1st</td>
<td>2nd</td>
</tr>
<tr>
<td>Amy</td>
<td>Yancey</td>
</tr>
<tr>
<td>Bertha</td>
<td>Xavier</td>
</tr>
<tr>
<td>Clare</td>
<td>Xavier</td>
</tr>
</tbody>
</table>
Stable Matching Problem

Perfect matching: everyone is matched monogamously.
- Each man gets exactly one woman.
- Each woman gets exactly one man.

Stability: no incentive for some pair of participants to undermine assignment by joint action.
- In matching $M$, an unmatched pair $m$-$w$ is unstable if man $m$ and woman $w$ prefer each other to current partners.
- Unstable pair $m$-$w$ could each improve by eloping.

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.
Stable Matching Problem

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

<table>
<thead>
<tr>
<th>Men's Preference Profile</th>
<th>Women's Preference Profile</th>
</tr>
</thead>
<tbody>
<tr>
<td>favorite  ↓</td>
<td>favorite  ↓</td>
</tr>
<tr>
<td>least favorite ↓</td>
<td>least favorite ↓</td>
</tr>
<tr>
<td>1st</td>
<td>2nd</td>
</tr>
<tr>
<td>Xavier</td>
<td>Amy</td>
</tr>
<tr>
<td>Yancey</td>
<td>Bertha</td>
</tr>
<tr>
<td>Zeus</td>
<td>Amy</td>
</tr>
</tbody>
</table>
Q. Is assignment X-C, Y-B, Z-A stable?
A. No. Bertha and Xavier will hook up.
**Stable Matching Problem**

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

<table>
<thead>
<tr>
<th></th>
<th>1st</th>
<th>2nd</th>
<th>3rd</th>
</tr>
</thead>
<tbody>
<tr>
<td>Xavier</td>
<td>Amy</td>
<td>Bertha</td>
<td>Clare</td>
</tr>
<tr>
<td>Yancey</td>
<td>Bertha</td>
<td>Amy</td>
<td>Clare</td>
</tr>
<tr>
<td>Zeus</td>
<td>Amy</td>
<td>Bertha</td>
<td>Clare</td>
</tr>
</tbody>
</table>

*Men’s Preference List*

<table>
<thead>
<tr>
<th></th>
<th>1st</th>
<th>2nd</th>
<th>3rd</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amy</td>
<td>Yancey</td>
<td>Xavier</td>
<td>Zeus</td>
</tr>
<tr>
<td>Bertha</td>
<td>Xavier</td>
<td>Yancey</td>
<td>Zeus</td>
</tr>
<tr>
<td>Clare</td>
<td>Xavier</td>
<td>Yancey</td>
<td>Zeus</td>
</tr>
</tbody>
</table>

*Women’s Preference List*
Stable Roommate Problem

Q. Do stable matchings always exist?
A. Not obvious a priori.

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

<table>
<thead>
<tr>
<th>Name</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>Doofus</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

Observation. Stable matchings do not always exist for stable roommate problem.
Propose-And-Reject Algorithm

Propose-and-reject algorithm. (Gale-Shapley, 1962) Intuitive method that guarantees to find a stable matching.

Initialize each person to be free.
while (some man is free and hasn't proposed to every woman) {
    Choose such a man m
    w = 1st woman on m's list to whom m 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
}
Proof of Correctness: Termination

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."

Claim. Algorithm terminates after at most $n^2$ iterations of while loop.
Pf. Each time through the while loop a man proposes to a new woman. There are only $n^2$ possible proposals. ■

<table>
<thead>
<tr>
<th>1st</th>
<th>2nd</th>
<th>3rd</th>
<th>4th</th>
<th>5th</th>
</tr>
</thead>
<tbody>
<tr>
<td>Victor</td>
<td>A</td>
<td>B</td>
<td>C</td>
<td>D</td>
</tr>
<tr>
<td>Wyatt</td>
<td>B</td>
<td>C</td>
<td>D</td>
<td>A</td>
</tr>
<tr>
<td>Xavier</td>
<td>C</td>
<td>D</td>
<td>A</td>
<td>B</td>
</tr>
<tr>
<td>Yancey</td>
<td>D</td>
<td>A</td>
<td>B</td>
<td>C</td>
</tr>
<tr>
<td>Zeus</td>
<td>A</td>
<td>B</td>
<td>C</td>
<td>D</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>1st</th>
<th>2nd</th>
<th>3rd</th>
<th>4th</th>
<th>5th</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amy</td>
<td>W</td>
<td>X</td>
<td>Y</td>
<td>Z</td>
</tr>
<tr>
<td>Bertha</td>
<td>X</td>
<td>Y</td>
<td>Z</td>
<td>V</td>
</tr>
<tr>
<td>Clare</td>
<td>Y</td>
<td>Z</td>
<td>V</td>
<td>W</td>
</tr>
<tr>
<td>Diane</td>
<td>Z</td>
<td>V</td>
<td>W</td>
<td>X</td>
</tr>
<tr>
<td>Erika</td>
<td>V</td>
<td>W</td>
<td>X</td>
<td>Y</td>
</tr>
</tbody>
</table>

$n(n-1) + 1$ proposals required
Proof of Correctness: Perfection

**Claim.** All men and women get matched.

**Pf.** (by contradiction)
- Suppose, for sake of contradiction, that Zeus is not matched upon termination of algorithm.
- Then some woman, say Amy, is not matched upon termination.
- By Observation 2, Amy was never proposed to.
- But, Zeus proposes to everyone, since he ends up unmatched. $\blacksquare$
Proof of Correctness: Stability

Claim. No unstable pairs.
Pf. (by contradiction)

- Suppose A-Z is an unstable pair: each prefers each other to partner in Gale-Shapley matching S*.

- Case 1: Z never proposed to A. ⇒ Z prefers his GS partner to A. ⇒ A-Z is stable.

- Case 2: Z proposed to A. ⇒ A rejected Z (right away or later) ⇒ A prefers her GS partner to Z. ⇒ A-Z is stable.

- In either case A-Z is stable, a contradiction. □
Summary

Stable matching problem. Given n men and n women, and their preferences, find a stable matching if one exists.

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?
Efficient Implementation

Efficient implementation. We describe $O(n^2)$ time implementation.

Representing men and women.
- Assume men are named 1, ..., n.
- Assume women are named 1', ..., n'.

Engagements.
- Maintain a list of free men, e.g., in a queue.
- Maintain two arrays $\text{wife}[m]$, and $\text{husband}[w]$.
  - set entry to 0 if unmatched
  - if $m$ matched to $w$ then $\text{wife}[m]=w$ and $\text{husband}[w]=m$

Men proposing.
- For each man, maintain a list of women, ordered by preference.
- Maintain an array $\text{count}[m]$ that counts the number of proposals made by man $m$. 
Efficient Implementation

Women rejecting/accepting.

- Does woman \( w \) prefer man \( m \) to man \( m' \)?
- For each woman, create inverse of preference list of men.
- Constant time access for each query after \( O(n) \) preprocessing.

<table>
<thead>
<tr>
<th>Amy</th>
<th>1(^{st})</th>
<th>2(^{nd})</th>
<th>3(^{rd})</th>
<th>4(^{th})</th>
<th>5(^{th})</th>
<th>6(^{th})</th>
<th>7(^{th})</th>
<th>8(^{th})</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pref</td>
<td>8</td>
<td>3</td>
<td>7</td>
<td>1</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Amy</th>
<th>1(^{st})</th>
<th>2(^{nd})</th>
<th>3(^{rd})</th>
<th>4(^{th})</th>
<th>5(^{th})</th>
<th>6(^{th})</th>
<th>7(^{th})</th>
<th>8(^{th})</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inverse</td>
<td>4(^{th})</td>
<td>8(^{th})</td>
<td>2(^{nd})</td>
<td>5(^{th})</td>
<td>6(^{th})</td>
<td>7(^{th})</td>
<td>3(^{rd})</td>
<td>1(^{st})</td>
</tr>
</tbody>
</table>

for \( i = 1 \) to \( n \)

invert[pref[i]] = i

Amy prefers man 3 to 6

2 7
Understanding the Solution

Q. For a given problem instance, there may be several stable matchings. Do all executions of Gale-Shapley yield the same stable matching? If so, which one?

An instance with two stable matchings.
- A-X, B-Y, C-Z.
- A-Y, B-X, C-Z.
Understanding the Solution

**Q.** For a given problem instance, there may be several stable matchings. Do all executions of Gale-Shapley yield the same stable matching? If so, which one?

**Def.** Man $m$ is a **valid partner** of woman $w$ if there exists some stable matching in which they are matched.

**Man-optimal assignment.** Each man receives best valid partner.

**Claim.** All executions of GS yield **man-optimal** assignment, which is a stable matching!

- No reason a priori to believe that man-optimal assignment is perfect, let alone stable.
- Simultaneously best for each and every man.
Man Optimality

Claim. GS matching $S^*$ is man-optimal.

Pf. (by contradiction)

- Suppose some man is paired with someone other than best partner. Men propose in decreasing order of preference $\Rightarrow$ some man is rejected by valid partner.
- Let $Y$ be first such man, and let $A$ be first valid woman that rejects him.
- Let $S$ be a stable matching where $A$ and $Y$ are matched.
- When $Y$ is rejected, $A$ forms (or reaffirms) engagement with a man, say $Z$, whom she prefers to $Y$.
- Let $B$ be $Z$'s partner in $S$.
- $Z$ not rejected by any valid partner at the point when $Y$ is rejected by $A$. Thus, $Z$ prefers $A$ to $B$.
- But $A$ prefers $Z$ to $Y$.
- Thus $A-Z$ is unstable in $S$. □
Stable Matching Summary

Stable matching problem. Given preference profiles of n men and n women, find a stable matching.

\[
\uparrow
\]

no man and woman prefer to be with each other than assigned partner

Gale-Shapley algorithm. Finds a stable matching in \(O(n^2)\) time.

Man-optimality. In version of GS where men propose, each man receives best valid partner.

\[
\uparrow
\]

w is a valid partner of m if there exist some stable matching where m and w are paired

Q. Does man-optimality come at the expense of the women?
Woman Pessimality

**Woman-pessimal assignment.** Each woman receives worst valid partner.

**Claim.** GS finds woman-pessimal stable matching $S^*$. 

**Pf.**
- Suppose $A-Z$ matched in $S^*$, but $Z$ is not worst valid partner for $A$.
- There exists stable matching $S$ in which $A$ is paired with a man, say $Y$, whom she likes less than $Z$.
- Let $B$ be $Z$'s partner in $S$.
- $Z$ prefers $A$ to $B$. — man-optimality
- Thus, $A-Z$ is an unstable in $S$. □
Extensions: Matching Residents to Hospitals

Variant 1. Some participants declare others as unacceptable.

Variant 2. Unequal number of men and women.

Variant 3. Limited polygamy.

Ex: Men ≈ hospitals, Women ≈ med school residents.

Def. Matching S unstable if there is a hospital h and resident r such that:
- h and r are acceptable to each other; and
- either r is unmatched, or r prefers h to her assigned hospital; and
- either h does not have all its places filled, or h prefers r to at least one of its assigned residents.
Application: Matching Residents to Hospitals

NRMP. (National Resident Matching Program)

- Original use just after WWII. — predates computer usage
- Ides of March, 23,000+ residents.

Rural hospital dilemma.

- Certain hospitals (mainly in rural areas) were unpopular and declared unacceptable by many residents.
- Rural hospitals were under-subscribed in NRMP matching.
- How can we find stable matching that benefits "rural hospitals"?

Rural Hospital Theorem. Rural hospitals get exactly same residents in every stable matching!
Lessons Learned

Powerful ideas learned in course.
- Isolate underlying structure of problem.
- Create useful and efficient algorithms.

Potentially deep social ramifications.
- Why is it that men propose?
1.2 Five Representative Problems
**Interval Scheduling**

**Input.** Set of jobs with start times and finish times.

**Goal.** Find maximum *cardinality* subset of mutually compatible jobs.

![Diagram showing intervals and jobs]

- Jobs don't overlap
Weighted Interval Scheduling

**Input.** Set of jobs with start times, finish times, and weights.

**Goal.** Find maximum *weight* subset of mutually compatible jobs.
Bipartite Matching

**Input.** Bipartite graph.

**Goal.** Find maximum *cardinality* matching.
Independent Set

**Input.** Graph.

**Goal.** Find maximum *cardinality* independent set.

subset of nodes such that no two joined by an edge
Competitive Facility Location

**Input.** Graph with weight on each node.

**Game.** Two competing players alternate in selecting nodes. Not allowed to select a node if any of its neighbors have been selected.

**Goal.** Select a *maximum weight* subset of nodes.

![Graph Diagram]

Second player can guarantee 20, but not 25.
Five Representative Problems

Variations on a theme: independent set.

Interval scheduling: \( n \log n \) greedy algorithm.
Weighted interval scheduling: \( n \log n \) dynamic programming algorithm.
Bipartite matching: \( n^k \) max-flow based algorithm.
Independent set: NP-complete.
Competitive facility location: PSPACE-complete.