1.1 A First Problem: Stable Matching

**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.

**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.

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

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

**A.** Yes.

<table>
<thead>
<tr>
<th>Men's Preference Profile</th>
<th>Women's Preference Profile</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>1st</strong></td>
<td><strong>2nd</strong></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>

### Stable Matching Problem

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

**A.** No. Bertha and Xavier will hook up.

<table>
<thead>
<tr>
<th>Men's Preference List</th>
<th>Women's Preference List</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>1st</strong></td>
<td><strong>2nd</strong></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>

### 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>Men's Preference List</th>
<th>Women's Preference List</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>1st</strong></td>
<td><strong>2nd</strong></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>

#### 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: 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.

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>Arron</td>
<td>C</td>
<td>D</td>
<td>A</td>
<td>B</td>
</tr>
<tr>
<td>Tommy</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>E</td>
</tr>
</tbody>
</table>

\( n(n-1) \times 1 \) proposals required

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.
    \( \Rightarrow \) Z prefers his GS partner to A.
    \( \Rightarrow \) A-Z is stable.

  • Case 2: Z proposed to A.
    \( \Rightarrow \) A rejected Z (right away or later)
    \( \Rightarrow \) A prefers her GS partner to Z. \( \leftarrow \) women only trade up
    \( \Rightarrow \) A-Z is stable.

  • In either case A-Z is stable, a contradiction.
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>Pref</th>
<th>1st</th>
<th>2nd</th>
<th>3rd</th>
<th>4th</th>
<th>5th</th>
<th>6th</th>
<th>7th</th>
<th>8th</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></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>Inverse</th>
<th>4th</th>
<th>8th</th>
<th>2nd</th>
<th>5th</th>
<th>6th</th>
<th>7th</th>
<th>3rd</th>
<th>1st</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
</tr>
</tbody>
</table>

```
for i = 1 to n
    inverse[pref[i]] = i
```

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 wife[m], and husband[w].
  - set entry to 0 if unmatched
  - if m matched to w then wife[m]=w and husband[w]=m

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

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 ⇒ 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.

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.

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.
- 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. [legal disclaimer]
Interval Scheduling

**Input.** Set of jobs with start times and finish times.
**Goal.** Find maximum cardinality subset of mutually compatible jobs.

---

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.

---

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:** \( n^2 \) max-flow based algorithm.

**Independent set:** NP-complete.

**Competitive facility location:** PSPACE-complete.