1.1 A First Problem: Stable Matching

Stable Matching Problem

Given $n$ men and $n$ women.
- 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.

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

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.
Is this Matching Stable?

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>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>

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>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>

Xavier ← Unstable! → Clare
Yancey ← Stable → Bertha
Zeus ← → Amy

Brute-force Search

Enumerate all perfect matchings; check for stableness.

# of perfect matchings: n!

Solves the problem, but inefficient.

Propose-And-Reject Algorithm


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
}
Analysis of the Algorithm

Why the algorithm is correct?

Is the algorithm efficient?

Next, we’ll carry out a series of analysis.

If you find it difficult, that’s OK.

- You are just not used to this kind of reasoning.

Proof of Correctness: Termination

We need to prove the program terminates. Remember an algorithm must be finite.

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.

\[
\begin{array}{ccccccccc}
\text{man} & A & B & C & D & E & W & X & Y & Z & V \\
\text{observation} & 1 & 2 & 3 & 4 & 5 & 1 & 2 & 3 & 4 & 5 \\
\text{man} & A & B & C & D & E & W & X & Y & Z & V \\
\text{woman} & A & B & C & D & E & W & X & Y & Z & V \\
\text{man} & A & B & C & D & E & W & X & Y & Z & V \\
\text{woman} & A & B & C & D & E & W & X & Y & Z & V \\
\text{man} & A & B & C & D & E & W & X & Y & Z & V \\
\text{woman} & A & B & C & D & E & W & X & Y & Z & V \\
\end{array}
\]

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.

\[ n(n-1) + 1 \] 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.
  ⇒ 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.

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 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.
Stable Matching: Not Unique

**Q.** For a given problem instance, there may be several stable matchings.

With **TWO** stable matchings:

<table>
<thead>
<tr>
<th>First</th>
<th>Second</th>
</tr>
</thead>
<tbody>
<tr>
<td>Xavier</td>
<td>Amy</td>
</tr>
<tr>
<td>Yancey</td>
<td>Bertha</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>First</th>
<th>Second</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amy</td>
<td>Yancey</td>
</tr>
<tr>
<td>Bertha</td>
<td>Xavier</td>
</tr>
</tbody>
</table>

The output of the GS algorithm.

Understanding the Solution

**Q.** Do all executions of Gale-Shapley yield the same stable matching? If so, which one?

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

**Def.** Woman $w$ is the best valid partner of man $m$ if $m$ prefers $w$ to any other valid partners.

**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.
  - In $S^*$, 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.

\[
\text{no man and woman prefer to be with each other than assigned partner.}
\]

**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.

- $w$ 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

**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. — non-optimality
- Thus, A-Z is an unstable in $S$. □

---

<table>
<thead>
<tr>
<th>S*</th>
<th>S</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zeus -- Amy</td>
<td>Yancey -- Amy</td>
</tr>
<tr>
<td></td>
<td>Zeus -- Bertha</td>
</tr>
</tbody>
</table>

---

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.

---

Lessons Learned

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

**Potentially deep social ramifications.** [legal disclaimer]
- Historically, men propose to women.
- CS majors get the best partners!
- Theory can be socially enriching and fun
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.

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.

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: $n^2$ max-flow based algorithm.
- Independent set: NP-complete.
- Competitive facility location: PSPACE-complete.