APPLIED MECHANISM DESIGN FOR SOCIAL GOOD

JOHN P DICKERSON

Lecture #11 – 03/01/2018

CMSC828M
Tuesdays & Thursdays
9:30am – 10:45am
LET’S TALK ABOUT PROJECTS
THIS CLASS: MATCHING & NOT THE NRMP

(SEE: LECTURE #9 OF FALL 2016 BY CANDICE SCHUMANN)
OVERVIEW OF THIS LECTURE

Stable marriage problem
  • Bipartite, one vertex to one vertex

Stable roommates problem
  • Not bipartite, one vertex to one vertex

Hospitals/Residents problem
  • Bipartite, one vertex to many vertices
MATCHING WITHOUT INCENTIVES

Given a graph $G = (V, E)$, a matching is any set of pairwise non-adjacent edges

- No two edges share the same vertex
- Classical combinatorial optimization problem

**Bipartite matching:**

- Bipartite graph $G = (U, V, E)$
- Max cardinality/weight matching found easily – $O(VE)$ and better
  - E.g., through network flow, Hungarian algorithm, etc

**Matching in general graphs:**

- Also PTIME via Edmond’s algorithm – $O(V^2E)$ and better
STABLE MARRIAGE PROBLEM

Complete bipartite graph with equal sides:

- $n$ men and $n$ women (old school terminology 😞)

Each man has a strict, complete preference ordering over women, and vice versa

Want: a stable matching

Stable matching: No unmatched man and woman both prefer each other to their current spouses
## Example Preference Profiles

<table>
<thead>
<tr>
<th></th>
<th>Diane</th>
<th>Emily</th>
<th>Fergie</th>
</tr>
</thead>
<tbody>
<tr>
<td>Albert</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bradley</td>
<td>Emily</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Charles</td>
<td>Diane</td>
<td>Emily</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Diane</th>
<th>Bradley</th>
<th>Albert</th>
<th>Charles</th>
</tr>
</thead>
<tbody>
<tr>
<td>Emily</td>
<td>Albert</td>
<td>Bradley</td>
<td>Charles</td>
</tr>
<tr>
<td>Fergie</td>
<td>Albert</td>
<td>Bradley</td>
<td>Charles</td>
</tr>
</tbody>
</table>
## EXAMPLE MATCHING #1

<table>
<thead>
<tr>
<th></th>
<th>Albert</th>
<th>Diane</th>
<th>Emily</th>
<th>Fergie</th>
</tr>
</thead>
<tbody>
<tr>
<td>Albert</td>
<td>Diane</td>
<td>Emily</td>
<td>Fergie</td>
<td></td>
</tr>
<tr>
<td>Bradley</td>
<td>Emily</td>
<td>Diane</td>
<td>Fergie</td>
<td></td>
</tr>
<tr>
<td>Charles</td>
<td>Diane</td>
<td>Emily</td>
<td>Fergie</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Diane</th>
<th>Bradley</th>
<th>Albert</th>
<th>Charles</th>
</tr>
</thead>
<tbody>
<tr>
<td>Diane</td>
<td>Bradley</td>
<td>Albert</td>
<td>Charles</td>
<td></td>
</tr>
<tr>
<td>Emily</td>
<td>Albert</td>
<td>Bradley</td>
<td>Charles</td>
<td></td>
</tr>
<tr>
<td>Fergie</td>
<td>Albert</td>
<td>Bradley</td>
<td>Charles</td>
<td></td>
</tr>
</tbody>
</table>
**EXAMPLE MATCHING #1**

<table>
<thead>
<tr>
<th></th>
<th>Albert</th>
<th>Diane</th>
<th>Emily</th>
<th>Fergie</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bradley</td>
<td>Emily</td>
<td>Diane</td>
<td>Fergie</td>
<td></td>
</tr>
<tr>
<td>Charles</td>
<td>Diane</td>
<td>Emily</td>
<td>Fergie</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Diane</th>
<th>Bradley</th>
<th>Albert</th>
<th>Charles</th>
</tr>
</thead>
<tbody>
<tr>
<td>Emily</td>
<td>Albert</td>
<td>Bradley</td>
<td>Charles</td>
<td></td>
</tr>
<tr>
<td>Fergie</td>
<td>Albert</td>
<td>Bradley</td>
<td>Charles</td>
<td></td>
</tr>
</tbody>
</table>

**No.**
Albert and Emily form a **blocking pair.**
### EXAMPLE MATCHING #2

<table>
<thead>
<tr>
<th></th>
<th>Albert</th>
<th>Diane</th>
<th>Emily</th>
<th>Fergie</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bradley</td>
<td>Emily</td>
<td>Diane</td>
<td>Fergie</td>
<td></td>
</tr>
<tr>
<td>Charles</td>
<td>Diane</td>
<td>Emily</td>
<td>Fergie</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Diane</th>
<th>Bradley</th>
<th>Albert</th>
<th>Charles</th>
</tr>
</thead>
<tbody>
<tr>
<td>Emily</td>
<td>Albert</td>
<td>Bradley</td>
<td>Charles</td>
<td></td>
</tr>
<tr>
<td>Fergie</td>
<td>Albert</td>
<td>Bradley</td>
<td>Charles</td>
<td></td>
</tr>
</tbody>
</table>
### EXAMPLE MATCHING #2

<table>
<thead>
<tr>
<th></th>
<th>Diane</th>
<th>Emily</th>
<th>Fergie</th>
</tr>
</thead>
<tbody>
<tr>
<td>Albert</td>
<td>Diane</td>
<td>Emily</td>
<td>Fergie</td>
</tr>
<tr>
<td>Bradley</td>
<td>Emily</td>
<td>Diane</td>
<td>Fergie</td>
</tr>
<tr>
<td>Charles</td>
<td>Diane</td>
<td>Emily</td>
<td>Fergie</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Bradley</th>
<th>Albert</th>
<th>Charles</th>
</tr>
</thead>
<tbody>
<tr>
<td>Diane</td>
<td>Bradley</td>
<td>Albert</td>
<td>Charles</td>
</tr>
<tr>
<td>Emily</td>
<td>Albert</td>
<td>Bradley</td>
<td>Charles</td>
</tr>
<tr>
<td>Fergie</td>
<td>Albert</td>
<td>Bradley</td>
<td>Charles</td>
</tr>
</tbody>
</table>

Yes!
(Fergie and Charles are unhappy, but helpless.)
SOME QUESTIONS

Does a stable solution to the marriage problem always exist?
Can we compute such a solution efficiently?
Can we compute the best stable solution efficiently?
GALE-SHAPLEY [1962]

1. Everyone is unmatched

2. While some man $m$ is unmatched:
   - $w := m$’s most-preferred woman to whom he has not proposed yet
   - If $w$ is also unmatched:
     - $w$ and $m$ are engaged
   - Else if $w$ prefers $m$ to her current match $m’$
     - $w$ and $m$ are engaged, $m’$ is unmatched
   - Else: $w$ rejects $m$

3. Return matched pairs
Claim
GS terminates in polynomial time (at most $n^2$ iterations of the outer loop)

Proof:
• Each iteration, one man proposes to someone to whom he has never proposed before
• $n$ men, $n$ women $\Rightarrow n \times n$ possible events

(Can tighten a bit to $n(n - 1) + 1$ iterations.)
Claim
GS results in a perfect matching

Proof by contradiction:
• Suppose BWOC that \( m \) is unmatched at termination
• \( n \) men, \( n \) women \( \rightarrow \) \( w \) is unmatched, too
• Once a woman is matched, she is never unmatched; she only swaps partners. Thus, nobody proposed to \( w \)
• \( m \) proposed to everyone (by def. of GS): \( >> \)
Claim
GS results in a stable matching (i.e., there are no blocking pairs)

Proof by contradiction (1):
- Assume $m$ and $w$ form a blocking pair

Case #1: $m$ never proposed to $w$
- GS: men propose in order of preferences
- $m$ prefers current partner $w' > w$
- $\rightarrow m$ and $w$ are not blocking
Claim
GS results in a stable matching (i.e., there are no blocking pairs)

Proof by contradiction (2):
Case #2: $m$ proposed to $w$
• $w$ rejected $m$ at some point
• GS: women only reject for better partners
• $w$ prefers current partner $m' > m$
• $\rightarrow m$ and $w$ are not blocking

Case #1 and #2 exhaust space. $><$
RECAP: SOME QUESTIONS

Does a stable solution to the marriage problem always exist?

Can we compute such a solution efficiently?

Can we compute the best stable solution efficiently?

We’ll look at a specific notion of “the best” – optimality with respect to one side of the market
(WO)MAN
OPTIMALITY/PESSIMALITY

Let $S$ be the set of stable matchings.

$m$ is a valid partner of $w$ if there exists some stable matching $S$ in $S$ where they are paired.

A matching is man optimal (resp. woman optimal) if each man (resp. woman) receives their best valid partner.

- Is this a perfect matching? Stable?

A matching is man pessimal (resp. woman pessimal) if each man (resp. woman) receives their worst valid partner.
Claim
GS – with the man proposing – results in a man-optimal matching

Proof by contradiction (1):
• Men propose in order → at least one man was rejected by a valid partner
• Let $m$ and $w$ be the first such reject in $S$
• This happens because $w$ chose some $m' > m$
• Let $S'$ be a stable matching with $m$, $w$ paired
  ($S'$ exists by def. of valid)
Claim
GS – with the man proposing – results in a man-optimal matching

Proof by contradiction (2):
• Let \( w' \) be partner of \( m' \) in \( S' \)
• \( m' \) was not rejected by valid woman in \( S \) before \( m \) was rejected by \( w \) (by assump.)
  \[ \rightarrow m' \text{ prefers } w \text{ to } w' \]
• Know \( w \) prefers \( m' \) over \( m \), her partner in \( S' \)
  \[ \rightarrow m' \text{ and } w \text{ form a blocking pair in } S' \]
RECAP: SOME QUESTIONS

Does a stable solution to the marriage problem always exist?

Can we compute such a solution efficiently?

Can we compute the best stable solution efficiently?

For one side of the market. What about the other side?
Claim
GS – with the man proposing – results in a woman-pessimal matching

Proof by contradiction:
• \( m \) and \( w \) matched in \( S \), \( m \) is not worst valid
• \( \Rightarrow \) exists stable \( S' \) with \( w \) paired to \( m' < m \)
• Let \( w' \) be partner of \( m \) in \( S' \)
• \( m \) prefers to \( w \) to \( w' \) (by man-optimality)
• \( \Rightarrow m \) and \( w \) form blocking pair in \( S' \) \( \gg \)
INCENTIVE ISSUES

Can either side benefit by misreporting?

• (Slight extension for rest of talk: participants can mark possible matches as unacceptable – a form of preference list truncation)

Any algorithm that yields woman-(man-)optimal matching

→

truthful revelation by women (men) is dominant strategy [Roth 1982]
In GS with men proposing, women can benefit by misreporting preferences

<table>
<thead>
<tr>
<th>Truthful reporting</th>
<th>Strategic reporting</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Albert</strong></td>
<td>Diane</td>
</tr>
<tr>
<td><strong>Bradley</strong></td>
<td>Emily</td>
</tr>
<tr>
<td><strong>Albert</strong></td>
<td>Diane</td>
</tr>
<tr>
<td><strong>Bradley</strong></td>
<td>Emily</td>
</tr>
<tr>
<td><strong>Diane</strong></td>
<td>Bradley</td>
</tr>
<tr>
<td><strong>Emily</strong></td>
<td>Albert</td>
</tr>
<tr>
<td><strong>Diane</strong></td>
<td>Bradley</td>
</tr>
<tr>
<td><strong>Emily</strong></td>
<td>Albert</td>
</tr>
<tr>
<td><strong>Albert</strong></td>
<td>Diane</td>
</tr>
<tr>
<td><strong>Bradley</strong></td>
<td>Emily</td>
</tr>
<tr>
<td><strong>Albert</strong></td>
<td>Diane</td>
</tr>
<tr>
<td><strong>Bradley</strong></td>
<td>Emily</td>
</tr>
<tr>
<td><strong>Diane</strong></td>
<td>Bradley</td>
</tr>
<tr>
<td><strong>Emily</strong></td>
<td>Albert</td>
</tr>
<tr>
<td><strong>Diane</strong></td>
<td>Bradley</td>
</tr>
<tr>
<td><strong>Emily</strong></td>
<td>Albert</td>
</tr>
</tbody>
</table>
Claim

There is no matching mechanism that:
1. is strategy proof (for both sides); and
2. always results in a stable outcome (given revealed preferences)
EXTENSIONS TO STABLE MARRIAGE
What if we have \( n \) men and \( n' \neq n \) women?

How does this affect participants? Core size?

- Being on short side of market: good!
- W.h.p., short side get rank \( \sim \log(n) \)
- … long side gets rank \( \sim \)random

# women held constant at \( n' = 40 \)
IMBALANCE [ASHLAGI ET AL. 2013]

Not many stable matchings with even small imbalances in the market
“Rural hospital theorem” [Roth 1986]:

- The set of residents and hospitals that are unmatched is the same for all stable matchings

Assume $n$ men, $n+1$ women

- One woman $w$ unmatched in all stable matchings
  - $\rightarrow$ Drop $w$, same stable matchings

Take stable matchings with $n$ women

- Stay stable if we add in $w$ if no men prefer $w$ to their current match
  - $\rightarrow$ average rank of men’s matches is low
ONLINE ARRIVAL [KHULLER ET AL. 1993]

Random preferences, men arrive over time, once matched nobody can switch

Algorithm: match \( m \) to highest-ranked free \( w \)

- On average, \( O(n \log(n)) \) unstable pairs

No deterministic or randomized algorithm can do better than \( \Omega(n^2) \) unstable pairs!

- Not better with randomization 😞
INCOMPLETE PREFS
[MANLOVE ET AL. 2002]

Before: complete + strict preferences
  • Easy to compute, lots of nice properties

Incomplete preferences
  • May exist: stable matchings of different sizes

Everything becomes hard!
  • Finding max or min cardinality stable matching
  • Determining if $<m,w>$ are stable
  • Finding/approx. finding “egalitarian” matching
NON-BIPARTITE GRAPH ...?

Matching is defined on general graphs:

- “Set of edges, each vertex included at most once”
- (Finally, no more “men” or “women” …)

The stable roommates problem is stable marriage generalized to any graph

Each vertex ranks all n-1 other vertices

- (Variations with/without truncation)

Same notion of stability
IS THIS DIFFERENT THAN STABLE MARRIAGE?

<table>
<thead>
<tr>
<th></th>
<th>Alana</th>
<th>Brian</th>
<th>Cynthia</th>
<th>Dracula</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alana</td>
<td>Brian</td>
<td>Cynthia</td>
<td>Dracula</td>
<td></td>
</tr>
<tr>
<td>Brian</td>
<td>Cynthia</td>
<td>Alana</td>
<td>Dracula</td>
<td></td>
</tr>
<tr>
<td>Cynthia</td>
<td>Alana</td>
<td>Brian</td>
<td>Dracula</td>
<td></td>
</tr>
<tr>
<td>Dracula</td>
<td>(Anyone)</td>
<td>(Anyone)</td>
<td>(Anyone)</td>
<td></td>
</tr>
</tbody>
</table>

No stable matching exists!
Anyone paired with Dracula (i) prefers some other $v$ and (ii) is preferred by that $v$
HOPELESS?

Can we build an algorithm that:

• Finds a stable matching; or
• Reports nonexistence

... In polynomial time?

Yes! [Irving 1985]

• Builds on Gale-Shapley ideas and work by McVitie and Wilson [1971]
IRVING’S ALGORITHM: PHASE 1

Run a deferred acceptance-type algorithm

If at least one person is unmatched: nonexistence

Else: create a reduced set of preferences

• $a$ holds proposal from $b \rightarrow a$ truncates all $x$ after $b$
• Remove $a$ from $x$’s preferences
• Note: $a$ is at the top of $b$’s list

If any truncated list is empty: nonexistence

Else: this is a “stable table” – continue to Phase 2
STABLE TABLES

1. \( a \) is first on \( b \)'s list iff \( b \) is last on \( a \)'s

2. \( a \) is not on \( b \)'s list iff
   - \( b \) is not on \( a \)'s list
   - \( a \) prefers last element on list to \( b \)

3. No reduced list is empty

Note 1: stable table with all lists length 1 is a stable matching

Note 2: any stable subtable of a stable table can be obtained via rotation eliminations
IRVING’S ALGORITHM: PHASE 2

Stable table has length 1 lists: return matching

Identify a rotation:

\((a_0, b_0), (a_1, b_1), \ldots, (a_{k-1}, b_{k-1})\) such that:
- \(b_i\) is first on \(a_i\)’s reduced list
- \(b_{i+1}\) is second on \(a_i\)’s reduced list (\(i+1\) is mod \(k\))

Eliminate it:
- \(a_0\) rejects \(b_0\), proposes to \(b_1\) (who accepts), etc.

If any list becomes empty: nonexistence

If the subtable hits length 1 lists: return matching
Claim
Irving’s algorithm for the stable roommates problem terminates in polynomial time – specifically $O(n^2)$.

This requires some data structure considerations

- Naïve implementation of rotations is $\sim O(n^3)$
ONE-TO-MANY MATCHING

The hospitals/residents problem (aka college/students problem aka admissions problem):

• Strict preference rankings from each side
• One side (hospitals) can accept $q > 1$ residents

Also introduced in [Gale and Shapley 1962]

Has seen lots of traction in the real world

• E.g., the National Resident Matching Program (NRMP)
• 5/1 – will talk about school choice
NEXT CLASS:

REAL-WORLD MATCHING: ORGAN EXCHANGE