Stable Marriages

Source
Kleinberg + Tardos
L \rightarrow \text{Algorithm}
\rightarrow \text{Design}

by
Anil Maheshwari
School of Comp. Sci
Carleton Univ.
OTTAWA - CANADA
Anil@scs.carleton.ca
Stable Marriages

N-men: m₁, m₂, ..., mₙ
N-women: w₁, w₂, ..., wₙ

preference list for each man/woman

N-marriages: (mᵢ₁, wᵢ₁) ... (mᵢₙ, wᵢₙ)
such that they are stable.

Stable Marriage: If there $\nexists$ a pair

$(m, w)$ such that

(a) $m$ likes $w$ better than his current partner.
(b) $w$ likes $m$ better than her current partner.

Problem: Given input $\rightarrow$ produce output.
Observe:

A. Matching ✓

B. Stable?

Consider pair (w4,m4)

→ w4 prefers m4 over m3
→ m4 prefers w4 over w5

⇒ Not Stable
STABLE MARRIAGES ARE NOT UNIQUE

\[
\begin{array}{c|c}
| w & w' | \\
\hline
| m' & m \ \\
\end{array} \quad \begin{array}{c|c}
| w & m' \ \\
\hline
| m & m' \ \\
\end{array}
\]

I: PAIRS (m,w) and (m',w') are Stable

II: PAIRS (m,w') and (m',w) are Stable
Problem:

Find a set of Marriages such that

→ Every Woman \( W_i \) is married.
→ Every Man \( M_i \) is married.
→ All Marriages are STABLE.
Proposal Algorithm

While ∃ an unmarried man who has not proposed to all women do

1. M chooses his favourite woman w who he has not proposed yet.

2. M proposes to w

3. if w is not married or likes M better than her current partner M’.

4. [w divorces M’.

5. [w marries M. 😊
Wow!

P1: Does there always exist a set of n-stable marriages?

P2: Does this algorithm always terminate?

P3: Does it always produce a correct result?

P4: How efficient is the algorithm?
Lemma 1: Termination

Proposal Algorithm terminates after at most $n^2$ iterations.

Proof: 1. There are $n$ men.
2. Each man can propose to $n$ women.
3. A man never proposes to the same woman twice.
4. At most $n^2$ proposals are made.
5. In each iteration one proposal is made.

$\implies$ At most $n^2$ iterations in all.
Lemma 2: When the proposal algorithm terminates every woman is married (⇒ every man is married).

Proof: (by contradiction).

1. Assume ∃ an unmarried woman w at termination of proposal algorithm.

2. ⇒ ∃ an unmarried man m.

3. Once a woman gets married, she stays married, though the partners may change.

4. When algorithm terminates than m must have proposed to each woman, including w.

5. ⇒ w is married, either to m or somebody whom she ranks higher.
**Lemma 3**: All marriages computed by proposal algorithm are stable.

**Proof**: (by contradiction)

Let \((w,m)\) and \((w',m')\) be two marriages computed by the algorithm such that

(a) \(w\) prefers \(m'\) over \(m\) \( (m <_w m') \)
(b) \(m'\) prefers \(w\) over \(w'\).  
(i.e. it's not stable)

Consider \(m'\).

Since \(m'\) prefers \(w\) over \(w'\),
\(m'\) would have proposed to \(w\) before \(w'\).

\(\Rightarrow\) Let \(m''\) be the man to whom \(w\) is married just after \(m'\) proposed.
Case A: If \( w \) accepts \( m' \), then \( m'' = m' \).

Case B: If \( w \) rejects \( m' \), then \( m'' > m' \).

Let \( m'' = m_1, m_2, \ldots, m_k = m \) be the sequence of partners that \( w \) has since this time till the end of the algorithm.

Notice that for any woman, only reason she switches the partner is because a higher preferred one proposed to her.

\[ \Rightarrow m'' = \begin{cases} m_1 <_w m_2 <_w m_3 <_w \ldots <_w m_k = m \end{cases} \]

From assumption we have that \( w \) prefers \( m' \) over \( m \) (i.e., \( m <_w m' \)).

Now we get \( m <_w m' \leq _w m'' = m_1 <_w m_2 <_w \ldots <_w m_k = m \).

\[ \Rightarrow m <_w m \text{ (Contradiction)} \]
Two Observations

Let \( (m, \text{best}(m)) \) denote a pair in stable matching for \( m \) such that no woman who \( m \) prefers more than \( \text{best}(m) \) forms a stable matching with \( m \).

In proposal algorithm, there is a choice in terms of choosing "unmarried man".

CLAIM: Any execution of proposal algorithm produces \( (m, \text{best}(m)) \) stable matching.

This also corresponds to \( (w, \text{worst}(w)) \) stable matching.
Nice problem to think about

Can one design an unbiased stable marriage algorithm.

Remaining Issues:

- How to implement proposal algorithm?
  - Data Structures / Link Lists
  - Run Time
  - Space / $O(n^2)$