Stable Marriages

Source

Kleinberg + Tardos
↳ ALGORITHM DESIGN

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

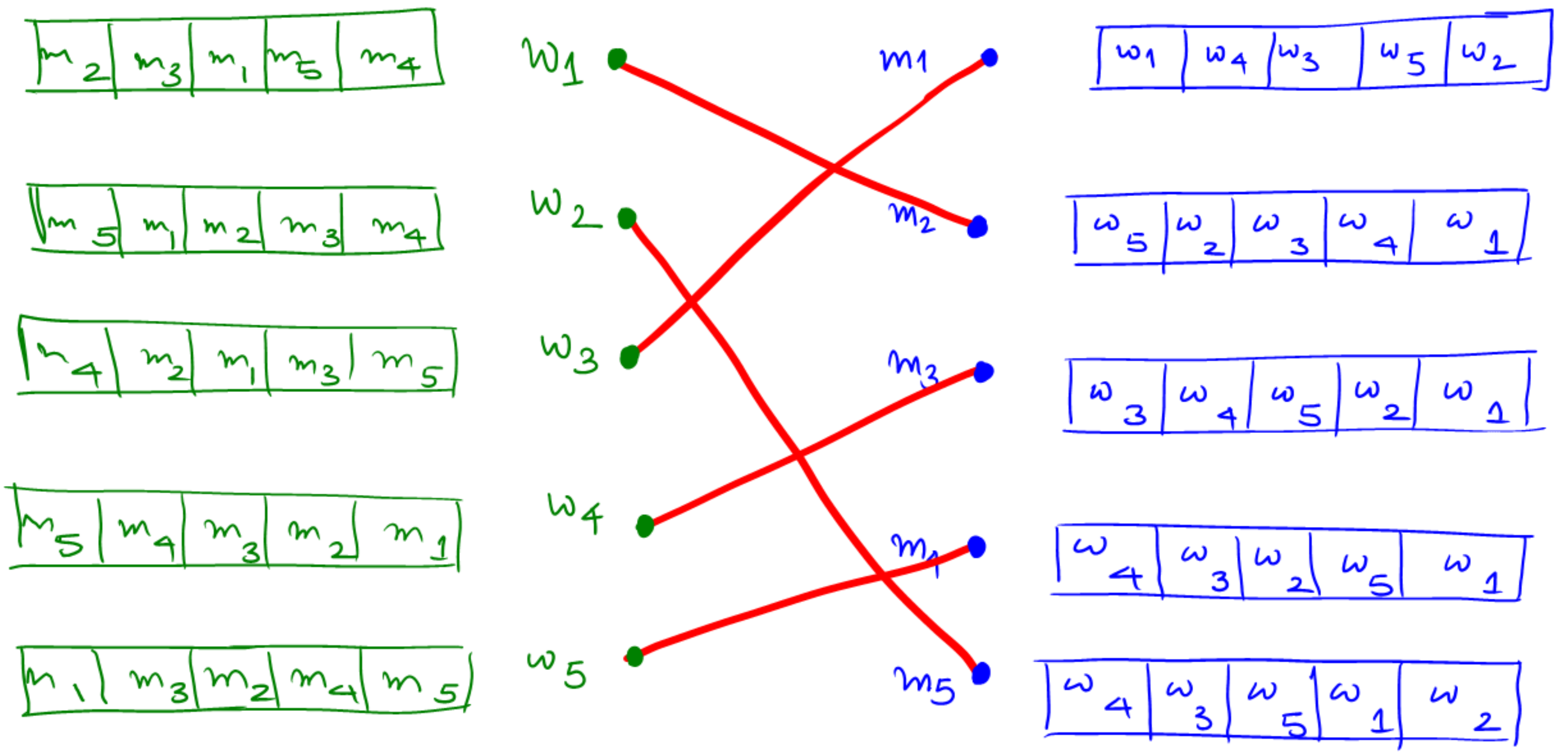
STABLE MARRIAGES

n-men: $m_1, m_2, \ldots, m_n$
n-women: $w_1, w_2, \ldots, w_n$ — INPUT
preference list for each man/woman

n-marriages: $(m_{i_1}, w_{j_1}) \ldots (m_{i_n}, w_{j_n})$
such that they are stable. — OUTPUT

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 ✓

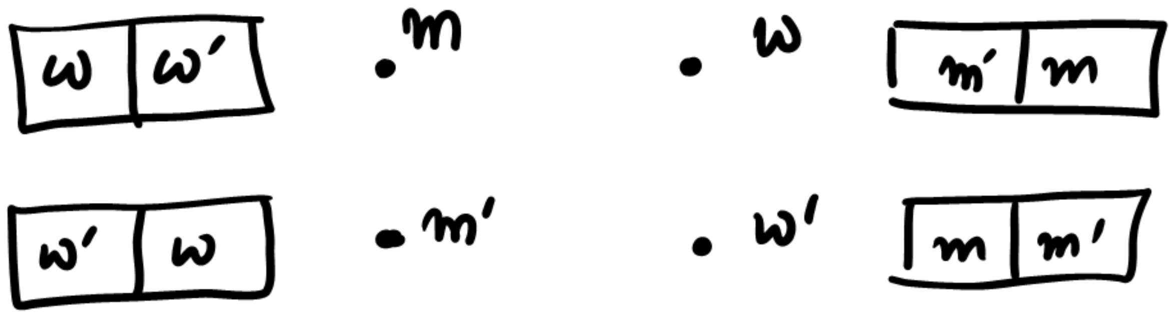
B. STABLE ? **CONSIDER PAIR (w_4, m_4)**

→ w_4 prefers m_4 over m_3

→ m_4 prefers w_4 over w_5

⇒ NOT STABLE

STABLE MARRIAGES ARE NOT UNIQUE



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 $\exists$ an unmarried man who has not proposed to all women do

1. m chooses his favourite woman w who he has not proposet 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 $\exists$ a set of n-stable marriages?

P2: Does this algorithm always terminate?

P3: Does it always produce correct result?

P4: How efficient is the algorithm?

DESIGN $\longleftrightarrow$ Properties + Correctness.

ANALYSIS $\longleftrightarrow$ Resource Complexities

ALGORITHMS

DAA

Lemma 1: Termination

Proposal Algorithm terminates after at most n^2 iterations.

<u>Proof</u>:

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.

$\Rightarrow$ At most n^2 iterations in all.

Lemma 2: When the proposal algorithm terminates every woman is married ($\Leftrightarrow$ every man is married).

Proof: (by contradiction).

1. Assume $\exists$ an unmarried woman w at termination of proposal algorithm.

2. $\Rightarrow$ $\exists$ an unmarried man m.

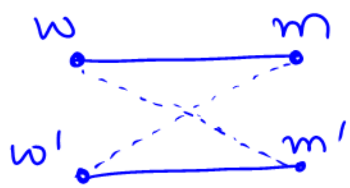
3. Once a woman gets married, she stays married, though the partners may change.

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

5. $\Rightarrow$ 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 $(m <_w m')$ — I

(b) m' prefers w over w'.

(i.e its 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 $m'' \underset{w}{>} m'$

$\Rightarrow m' \le_w m''$ II

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.]

$$\Rightarrow m'' = m_1 <_w m_2 <_w <_w m_3 <_w \cdots <_w m_k = m \quad \text{— III}$$

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

Now we get $\underset{\text{I}}{m <_w m'} \underset{\text{II}}{\le_w} m'' = m_1 \underset{\text{III}}{<_w m_2 <_w \cdots <_w} m_k = m$.

$$\Rightarrow m <_w m \quad (\text{Contradiction})$$

Two Observations

Let (m, best(m)) denote a pair
in stable matching for m
such that no woman who
m prefers more than 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, best(m)) stable matching.

This also corresponds to (w, worst(w)) stable-matching.

Nice problem to think about

Can one design an unbiased stable marriage algorithm.

—x—

Remaining Issues:

How to implement proposal algorithm?

→ Data Structures / Link Lists

→ Run Time

→ Space

/ $O(n^2)$