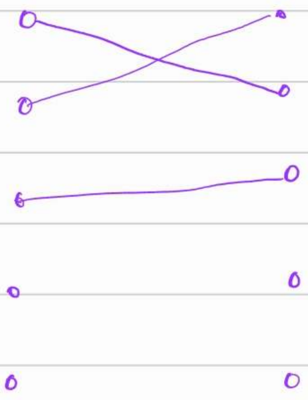
Stable Matching

23/02/2022

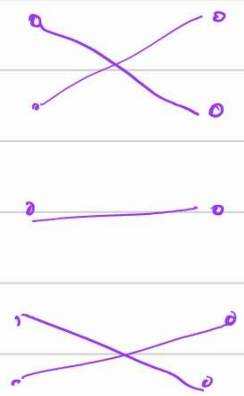


$M = \{m_1, \ldots, m_n\}$

$W = \{w_1, \ldots, w_n\}$

$(m_{i_1}, w_{i_1}), (m_{i_2}, w_{i_2})$

Defn:- A matching is a set of edges where every vertex has at most one edge incident on it.



$\leftarrow$ Perfect Matching

Defn:- where every vertex has exactly one edge incident on it.

Constraints: Each Man & Woman has a preference list.

$m_1 \rightarrow \{w_{i_1} > w_{i_2} > \cdots > w_{i_n}\}$

$w_1 \rightarrow \{m_{i_1} > m_{i_2} > \cdots > m_{i_n}\}$

Things that can go wrong in a matching

(m, w) & (m', w')

m — w

m' — w'

But according to their preference list

$$m \to \begin{matrix} w' \\ \vee \\ w \end{matrix}$$

m prefers w' to w

$$w' \to \begin{matrix} m \\ \vee \\ m' \end{matrix}$$

w' prefers m to m'.

nothing prevents m and w' abandoning their current parents and getting married with each other.

This we will call as instability in a matching.

We want a perfect matching where this instability doesn't occur.

This is called as "Stable matching".

(m, w) in a stable matching
(m', w')

then

1. m prefers w to w' or
2. w' prefers m' to m

Example 1

$\{w' > w\} \leftarrow m \qquad w \rightarrow \{m > m'\}$

$\{w > w'\} \leftarrow m' \qquad w' \rightarrow \{m' > m\}$

① $m - w$, $m' - w'$: Stable matching

② $m - w'$, $m' - w$: stable matching

Example 2

$m \qquad w$

$m' \qquad w'$

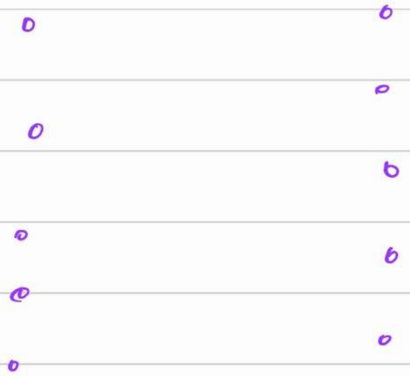
$m - \{w > w'\} \qquad w - \{m > m'\}$

$m' - \{w > w'\} \qquad w' - \{m > m'\}$

$m - w$, $m' - w'$: Stable.

$m - w'$, $m' - w$: Unstable

→ Gale-Shapely' 1960s
(Noble prize of economics)



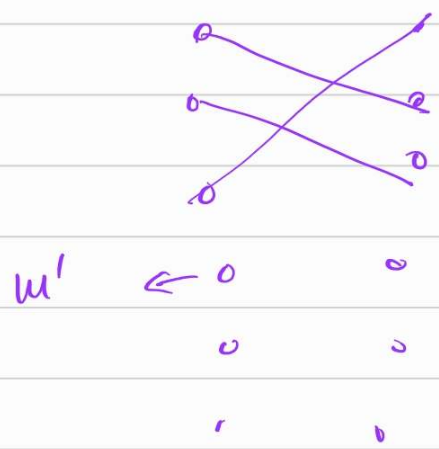
$M = \{m_1, \ldots, m_n\}$

$W = \{w_1, \ldots, w_n\}$

Devising an Algorithm:

① $m \xrightarrow{\text{proposes}} w_i$ to the highest ranked woman in his preference list.

② w accepts m's proposal (temporarily!) (Engagement!) if she is free.

③ 

m' proposes to the highest ranked woman he has not proposed yet.

$m' \xrightarrow[\text{to}]{\text{proposal}} w'$

$m'' \text{———} w'$ but in w' preference list $m' > m''$

then she accepts m' proposal.
and m'' becomes free.

Otherwise ; $m'' > m'$ then she rejects m' proposal.

Algorithm :

Intially all $m \in M$ and $w \in W$ are free

While there is a $m \in M$ who is free and hasn't proposed to every woman

Choose such a man m

let w be the highest-ranked woman in m's preference list to whom he hasn't proposed yet.

if w is free then
(m, w) become engaged. and this to matching.

Else w is currently engaged to m'
if w prefers m to m' then
(m, w) becomes engaged
m' becomes free

Else w rejects m and m remains free

End if.
End if
End while.

Output the matching set.

Observations 1: Once a woman is engaged she remains engaged throughout.
Moreover, her current partner's rank keeps increasing throughout the execution.

Observations 2: From man's perspective, his current partner's rank keeps decreasing.

Runtime: at most $O(n^2)$ proposals

Lemma:- The algorithm produces a stable matching.

Claim: It produces a perfect matching

Proof:- m proposes to every woman as long as he is free.

Proof of the lemma:- (Stability)

Proof by contradiction:

Let S be the output of the algorithm.

S is not stable by assumption.

(m, w) s.t. m prefers $w' > w$
(m', w') and w' prefers $m > m'$

In the algo, m proposes to w' before w.

Case 1:- w' and m get engaged temporarily.

w' prefers m' over m.
This contradicts our assumption.

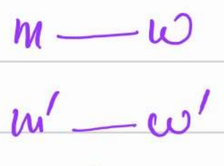
Case 2:- w' rejects m's proposal.

w' was engaged with m''.
But rejection implies w' prefers $m'' > m$.

(m', w') this implies $m' > m'' > m$.

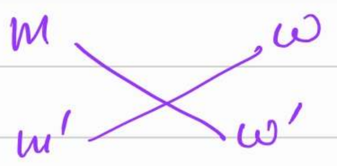
$\{w' > w\} \leftarrow m$ $\qquad$ $w \rightarrow \{m > m'\}$

$\{w > w'\} \leftarrow m'$ $\qquad$ $w' \rightarrow \{m' > m\}$



1

Stable matching



2

Stable matching.

Our algorithm produces the second stable matching in the above example.

This algorithm is better from the perspective of proposer's.