

Session 34: Stable Matchings

- Matching
- Maximum Matching
- Stable Matching
- Greedy Algorithm for finding the Stable Matching

Stable Matchings

Task: Pair elements from equally sized two groups considering their preferences for members of the other group so that there are no ways to improve the preferences.

This requires first some definitions ...

Men - Women

Servers - Services

Matchings

Definition: Given a finite set A , a **matching** of A is a set of (unordered) pairs of distinct elements of A where any element occurs in at most one pair (such pairs are called independent).

Definition: A **maximum matching** is a matching that contains the largest possible number of independent pairs.

Examples

Let $A = \{1, 2, 3, 4\}$

- $\{(1, 2)\}$ and $\{(1, 3), (2, 4)\}$
- $\{(2, 2)\}$ and $\{(1, 2), (2, 4)\}$
- $\{(1, 3), (2, 4)\}$

are matchings

are not matchings

is a maximum matching

Let $A = \{1, 2, 3, 4, 5\}$

- $\{(1, 3), (2, 4)\}$ and $\{(5, 3), (2, 4)\}$ are two different maximum matchings

Preferences


A **preference list** L_x defines for every element $x \in A$ the order in which the element prefers to be paired with. $x \in A$ prefers y to z if y precedes z on L_x .

Example: $A = \{\text{Lou, Glenn, Bobbie, Tyler}\}$


- $L_{\text{Lou}} = (\text{Glenn, Bobbie, Tyler})$
- $L_{\text{Glenn}} = (\text{Bobbie, Lou, Tyler})$
- $L_{\text{Bobbie}} = (\text{Lou, Glenn, Tyler})$
- $L_{\text{Tyler}} = (\text{Lou, Glenn, Bobbie})$

Lou prefers Glenn over Bobbie, and Bobbie over Tyler

Stability of Matching

Definition: A matching is **unstable** if there are two pairs $(x, y), (v, w)$ in the matching such that x prefers v to y and v prefers x to w . 

Definition: A **stable** matching is a matching that is not unstable.

Example: $\{(Glenn, Lou), (Bobbie, Tyler)\}$ is unstable 

- $L_{Glenn} = (Bobbie, Lou, Tyler)$: Glenn prefers Bobbie over Lou
- $L_{Bobbie} = (Lou, Glenn, Tyler)$: Bobbie prefers Glenn over Tyler

Therefore Glenn and Bobbie will leave their current partner and pair up.

Stability of Matching



New matching: $\{(\text{Glenn}, \text{Bobbie}), (\text{Lou}, \text{Tyler})\}$

- $L_{\text{Lou}} = (\text{Glenn}, \text{Bobbie}, \text{Tyler})$: Lou prefers Bobbie
- $L_{\text{Bobbie}} = (\text{Lou}, \text{Glenn}, \text{Tyler})$: Bobbie prefers Lou

Therefore Lou and Bobbie will leave their current partner and pair up.

(why does Lou prefers Glenn not work?)



New matching: $\{(\text{Lou}, \text{Bobbie}), (\text{Glenn}, \text{Tyler})\}$

- $L_{\text{Lou}} = (\text{Glenn}, \text{Bobbie}, \text{Tyler})$: Lou prefers Glenn
- $L_{\text{Glenn}} = (\text{Bobbie}, \text{Lou}, \text{Tyler})$: Glenn prefers Lou

New matching $\{(\text{Glenn}, \text{Lou}), (\text{Bobbie}, \text{Tyler})\}$ is initial matching

There does not exist a stable maximal matching!

(no one wants to stay with Tyler)

Marriage Problem

To guarantee, irrespective of the preference lists, the existence of a stable maximum matching it suffices to use a more stringent pairing rule.

Definition: Given a set with even cardinality, partition A into two disjoint subsets A_1 and A_2 with $A_1 \cup A_2 = A$ and $|A_1| = |A_2|$. A **matching** is a bijection from the elements of one set to the elements of the other set.

That means, that pairs can only consist of one element of A_1 and A_2 each.

Example

Assume $A_1 = \{\text{Lou, Glenn}\}$ and $A_2 = \{\text{Bobbie, Tyler}\}$

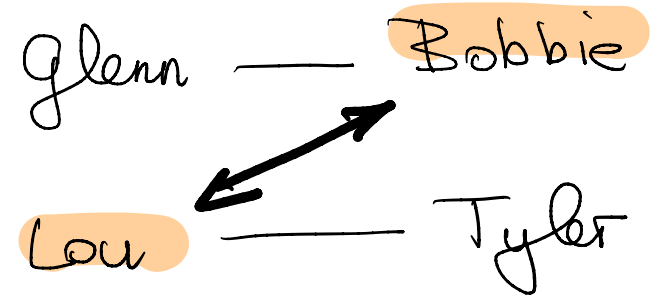
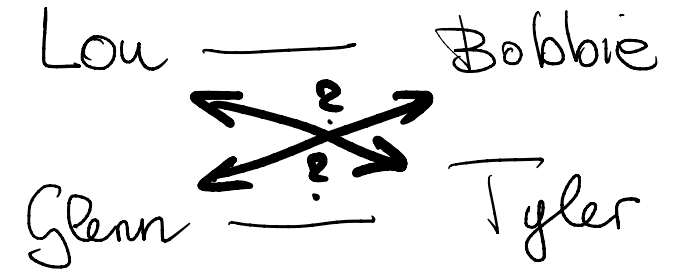
We have to adapt the preference lists

- $L_{\text{Lou}} = (\text{Bobbie, Tyler})$
- $L_{\text{Glenn}} = (\text{Bobbie, Tyler})$
- $L_{\text{Bobbie}} = (\text{Lou, Glenn})$
- $L_{\text{Tyler}} = (\text{Lou, Glenn})$

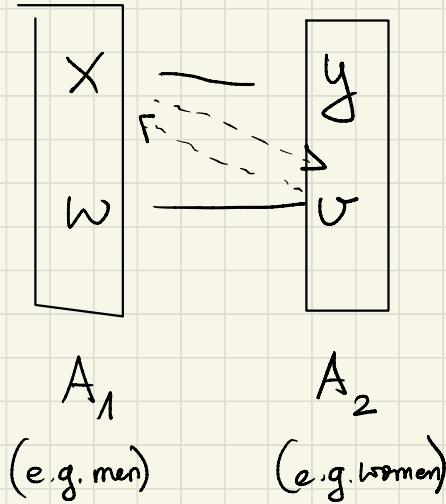
Now $\{(\text{Lou, Bobbie}), (\text{Glenn, Tyler})\}$ is stable

~~$\{(\text{Bobbie, Glenn}), (\text{Lou, Tyler})\}$~~ is unstable (Lou prefers Bobbie, Bobbie prefers Lou)

(Glenn, Bobbie)



Stability illustrated: matching (x, y) (w, v)



x prefers v over y

v prefers x over w

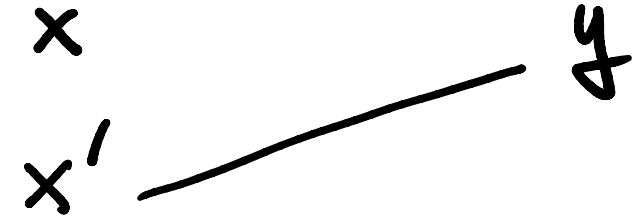
Existence of Stable Maximum Matching

A greedy algorithm to construct a stable maximum matching for the marriage problem

- It proves existence of stable matching
- It is efficient

(and it produced a Nobel price)

Gale-Shapley Algorithm



Let M be the set of pairs under construction;

Initially $M = \emptyset$

While $|M| < |A_1|$:

 Select an unpaired $x \in A_1$

 Let x propose to the first element $y \in A_2$ on L_x :

if y is unpaired **then** add the pair (x, y) to M

(greedy, (x, y) are "happy")

else (i.e., if y is paired already)

 Let $x' \in A_1$ be the element that y is paired to, (i.e., $(x', y) \in M$)

if x' precedes x on L_y **then** remove y from L_x (greedy, "y is lost for x")

else (i.e., if x precedes x' on L_y)

 Replace $(x', y) \in M$ by (x, y) and remove y from $L_{x'}$ (greedy, "y is lost for x' ", exchange partner)

The "loser" removes y from its candidate list

$$A_1 = \{x_1, x_2, x_3, x_4\} \quad A_2 = \{y_1, y_2, y_3, y_4\} \quad L_{x_i} = \{y_1, y_2, y_3, y_4\}$$

all x_i have same preferences

$$L_{y_1} = \{x_2, x_1, x_3, x_4\}$$

$$L_{y_2} = \{x_4, x_3, x_2, x_1\}$$

$$L_{y_3} = \{x_2, x_3, x_4, x_1\}$$

$$L_{y_4} = \{x_4, x_1, x_2, x_3\}$$

$$x_1 \rightarrow y_1 : y_1 \checkmark$$

$$x_2 \rightarrow y_1 : y_1 \checkmark, \quad x_1 \text{ leaves } y_1$$

$$x_1 \rightarrow y_2 : y_2 \checkmark$$

$$x_3 \rightarrow y_1 : y_1 \times, \quad x_3 \text{ dumps } y_1$$

$$x_3 \rightarrow y_2 : y_2 \checkmark, \quad x_1 \text{ dumps } y_2$$

$$x_1 \rightarrow y_3$$

⋮

$$M : (x_1, y_1)$$

$$M : (x_2, y_1)$$

$$M : (x_2, y_1)(x_1, y_2)$$

$$M : (x_2, y_1)(x_1, y_2)$$

$$M : (x_2, y_2)(x_3, y_2)$$

Summary

- Definition of Maximum Stable Matching
- Gale-Shapley Algorithm
 - Shows existence of maximum stable matching