§1 Stable Matching

Motivation: Matching KAIST students with labs <u>automatically</u> (algorithm!) to find <u>stable</u> solution.

Inputs: a) each student's order of preferred labs
b) each lab's order of preferred students

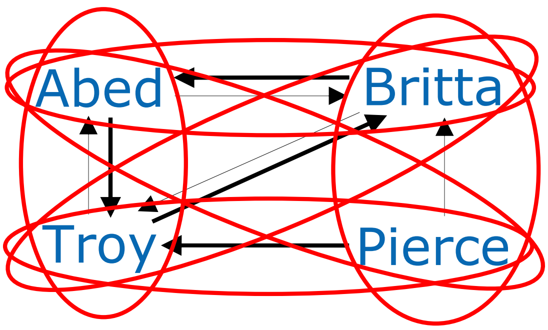
Output: 1-1 pairing w/out *unstable* tuples

Def: Tuple (S,P') is *unstable* if S prefers P' over assigned P and P' prefers S over assigned S'

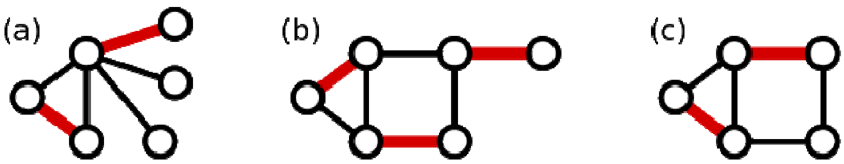
Stable Matching

CS500 M. Ziegler

Does it always exist? No!



Reminder: *A perfect* matching in a graph $G=(V,E)$ of $|V|=2n$ vertices is a subset M of n edges without common vertices.



<u>Specification:</u>

Input: n 'men' and n 'women', each with a ranking of preference among the opposite 'gender'.

Output: stable perfect matching

Def: Tuple (w,m') is *unstable* if w prefers m' over assigned m and m' prefers w over assigned w'

Stable Matching Algorithm

Machist

Gale-Shapley (1962)

M := {}

WHILE some m is unmatched

  Let m propose to w := first on m's list
    that m has not yet proposed to.

  IF w is unmatched, add (m,w) to M

  ELIF w prefers m to current partner m'
    replace (m',w) in M with (m,w)

  ELSE w rejects proposal from m.

ENDWHILE // output: M

Specification:

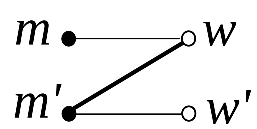
Input: n 'men' and n 'women', each with a ranking of preference among the opposite 'gender'.

Output: 'matching' w/out *unstable* tuples

Def: Tuple (w,m') is *unstable* if w prefers m' over assigned m and m' prefers w over assigned w'

Proof of Correctness

Observation A: Once a woman is matched, she never becomes unmatched but only "trades up".



Observation B: Any man proposes to women in decreasing order of preference.

M := {}

WHILE some m is unmatched

  Let m propose to w := first on m's list
    that m has not yet proposed to.

  IF w is unmatched, add (m,w) to M

  ELIF w prefers m to current partner m'
    replace (m',w) in M with (m,w)

  ELSE w rejects proposal from m.

ENDWHILE // output: M

Claim 1: At most n^2 proposals made.

Claim 2: Then all are matched.

Claim 3: Matching w/o unstable pairs.

Def: Tuple (w,m') is *unstable* if w prefers m' over assigned m and m' prefers w over assigned w'

Efficiency: implement in $O(n^2)$

Represent men by numbers $1\ldots n$; same for women.

Input: n-element arrays with order of preference for each $m,w=1\ldots n$

Output: matching, represented by two n-element arrays $\text{wife}[m]=w$ and $\text{husband}[w]=m$; $=0$ if unmatched.

WHILE some m is unmatched
  Let m propose to w := first on m's list that m has not yet proposed to.
  IF w is unmatched, add (m,w) to M
  ELIF w prefers m to current partner m' replace (m',w) in M with (m,w)
  ELSE w rejects proposal from m.
ENDWHILE // output: M

For each man m, $\text{nextProposal}[m]$

For each woman, <u>inverted</u> order of preference.

Is this running time optimal?

Understanding the Solution

Represent men by numbers $1\ldots n$; same for women.

Input: n-element arrays with order of preference for each $m,w=1\ldots n$

Example [<u>two</u> stable matchings]

	1st	2nd	3rd
Abed	Annie	Britta	Frankie
Ben	Britta	Annie	Frankie
Craig	Annie	Britta	Frankie

	1st	2nd	3rd
Annie	Ben	Abed	Craig
Britta	Abed	Ben	Craig
Frankie	Abed	Ben	Craig

{ (Abed,Annie) , (Ben,Britta) , (Craig,Frankie) }

{ (Abed,Britta) , (Ben,Annie) , (Craig,Frankie) }

why?

<u>Macho</u> Gale-Shapley produces *that* stable matching where every m gets assigned his *most* preferred choice among all w matched to him in *any* stable matching; whereas w gets assigned her *least* preferred choice.