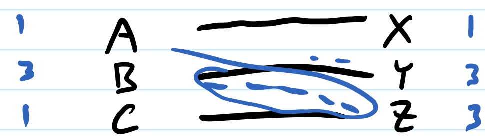
Stable Matching

Problem (informal): Given n machinists and n welders, find a good way to match them.

what does this mean?

Machinist	Preferences	Welder	Preferences
A	X, Y, Z	X	A, B, C
B	X, Z, Y	Y	A, C, B
C	Z, X, Y	Z	A, B, C



(B, Z) B prefers Z to Y
Z prefers B to C

instability

Matching: subset of $M \times W$ s.t. each $m \in M$ appears in ≤ 1 pair
each $w \in W$ appears in ≤ 1 pair

Perfect matching: a matching that includes all elts of M, W (so $\leq$ turns into $=$)

Instability: pair (m, w) that aren't matched w/each other who would prefer being together to who they are paired with

find w' s.t. $(m, w') \in$ Matching
m' $(m', w) \in$ Matching
m prefers w to w'
w prefers m to m'

Stable Matching: perfect matching with no instabilities

Gayle-Shapely

```
FreeM <- M
FreeW <- W
Invitations <- {}
Tentative <- {}

While there is an m in FreeM s.t. there is a w s.t. (m,w) not in Invitations
    choose such an m
    let w be m's highest ranked s.t. (m,w) not in Invitations

    add (m,w) to Invitations

    if w in FreeW then
        remove w from FreeW
        remove m from FreeM
        add (m,w) to Tentative
    else
        find m' s.t. (m', w) in Tentative
        if w prefers m to m'
            remove m from FreeM
            add m' to FreeM
            remove (m', w) from Tentative
            add(m, w) to Tentative
return Tentative
```

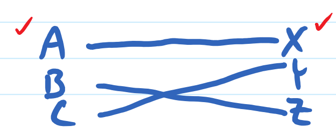
A	B	C
X	Y	Z
X	Z	Y
Y	X	Z
⋮	⋮	⋮

n! permutations

Invitations	FreeM	FreeW	Tentative
matches we've tried to make	not in current Tentative		pairs we've made (might be discarded)
	A, B, C	X, Y, Z	
(A, X)	B, C	Y, Z	(A, X)
(C, Z)	B	Y	~~(C, Z)~~
(B, X)			
(B, Z)	C		(B, Z)
(C, X)			
(C, Y)	∅		(C, Y)

Machinist	Preferences
A	X, Y, Z
B	X, Z, Y
C	Z, X, Y

Welder	Preferences
X	A, B, C
Y	A, C, B
Z	A, B, C



(B, X) NO
(B, Y) an instability? NO
(C, X) NO
(C, Z) NO

Does this always terminate?

Does this return a stable matching?

What is the running time?

Gale-Shapley Invariant

a) $\forall m,\ m \notin FreeM \longleftrightarrow \exists w$ s.t. $(m,w) \in Tent$
$\forall w,\ w \notin FreeW \longleftrightarrow \exists m$ s.t. $(m,w) \in Tent$ — FreeM, FreeW keep track of unmatched M, W

b) $\forall w,\ w \in FreeW \longleftrightarrow \sim\exists m$ s.t. $(m,w) \in Invites$ — free W = W with no invitations

c) Tent is a matching and stable (when viewed using M, W reduced to those elts in Tent)

d) $|Invites| = k$

e) $\forall w,\ j < k,\ MatchW_j(w) \neq NIL \rightarrow MatchW_{j+1}(w), \ldots, MatchW_k(w)\ != NIL$

the value after j iterations of the loop

once w is matched for 1st time, w is never free again

f) $\forall w,\ MatchW(w) =$ most preferred m s.t. $(m,w) \in Invites$ (or NIL if no such m)
$\hookrightarrow$ the m such that $(m,w) \in Tentative$ (or NIL if no such m)

g)

h) things we will discover are missing

i)

Initialization / Basis

a) For all m, w $\quad F \longleftrightarrow F$

b) For all w $\quad T \longleftrightarrow T$

c) $Tent = \emptyset$

d) $k = 0,\ |Invites| = 0$

e) f) } vacuous

Bookkeeping for Proofs

```
FreeM <- M
FreeW <- W
Invitations <- {}
Tentative <- {}
k ← 0
While there is an m in FreeM s.t. there is a w s.t. (m,w) not in Invitations
    choose such an m
    let w be m's highest ranked s.t. (m,w) not in Invitations

    add (m,w) to Invitations

    if w in FreeW then
        remove w from FreeW
        remove m from FreeM
        add (m,w) to Tentative
    else
        find m' s.t. (m', w) in Tentative
        if w prefers m to m'
            remove m from FreeM
            add m' to FreeM
            remove (m', w) from Tentative
            add(m, w) to Tentative
return Tentative
```

$k \leftarrow k+1$

$FreeW_k \leftarrow FreeW$

$FreeM_k \leftarrow FreeM$

$Tentative_k \leftarrow Tentative$

⋮

Thm: If Alg A' = Alg A with variables not read from or in output removed then output of A' = output of A for all inputs

Proof: INV: At each step, values of remaining variables are same in A, A'

Maintenance (easy parts)

```
FreeM <- M
FreeW <- W
Invitations <- {}
Tentative <- {}
k <- 0

while there is an m in FreeM s.t. there is a w s.t. (m,w) not in Invitations
    choose such an m
    let w be m's highest ranked s.t. (m,w) not in Invitations

    add (m,w) to Invitations

    if w in FreeW then
        remove w from FreeW
        remove m from FreeM
        add (m,w) to Tentative
    else
        find m' s.t. (m', w) in Tentative
        if w prefers m to m'
            remove m from FreeM
            add m' to FreeM
            remove (m', w) from Tentative
            add(m, w) to Tentative
    k <- k+1
return Tentative
```

d) $|\text{Invites}| = k$

$|\text{Invites}_{old}| = k_{old}$

$k_{new} = k_{old} + 1$

$\text{Invites}_{new} = \text{Invites}_{old} \cup \{(m,w)\}$

$(m,w) \notin \text{Invites}$ by choice of w

$|\text{Invites}_{new}| = |\text{Invites}_{old}| + |\{(m,w)\}|$
$= k+1$

Suppose INV is T before loop and $\exists m \in \text{FreeM}, w$ s.t. $(m,w) \notin \text{Invites}$

a) $\forall m,\ m \notin \text{FreeM} \longleftrightarrow \exists w$ s.t. $(m,w) \in \text{Tent}$
$\forall w,\ w \notin \text{FreeW} \longleftrightarrow \exists m$ s.t. $(m,w) \in \text{Tent}$

Only M changed are m, m'

in cases 1,2 m removed from FreeM, (m,w) added to Tent
in 2 m' added to FreeM, (m',w) removed from Tent
and no other $(m',w') \in \text{Tent}$
in case 3 no changes

Only W changed is w

in case 1, w removed from FreeW, (m,w) added to Tent $F \leftrightarrow F$
in cases 2, $w \notin \text{FreeW}$
(m,w) added to Tent $T \leftrightarrow T$
in case 3, no change

b) $\forall w,\ w \in \text{FreeW} \longleftrightarrow \sim\exists m$ s.t. $(m,w) \in \text{Invites}$

Only w changed is w

(m,w) added to Invites
in case 1, w removed from FreeW
in cases 2,3 $w \notin \text{FreeW}$ to start with and not changed
so $F \leftrightarrow F$ at end of loop

e) $\forall w,\ j<k,\ \text{MatchW}_j(w) \neq NIL \rightarrow \text{MatchW}_{j+1}(w), \ldots, \text{MatchW}_k(w)$!= NIL

Only w changed, and w will always have $(m,w) \in \text{Tent}$ or $(m',w) \in \text{Tent}$
so $\text{MatchW}(w) \neq NIL$