Correctness

```
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
```

seq of matches for welder 3

$m_3 = NIL, NIL, NIL, 4, 4, \underbrace{4, 6, 1}, 1, \ldots$

Match[i] = m_{ij} after j iterations

Obs 1: Let m_{ij} be the machinist welder i is tentatively matched with after j iterations (or NIL if not matched yet)

a) Then if m invites w_i during iteration j, $m_{ij} \neq NIL$ and $m_{ik} \neq NIL$ for $k > j$

b) If $k > j$ and $m_{ij} \neq m_{ik}$ then welder i prefers m_{ik} to m_{ij}

Obs 2: $m \in$ FreeM $\leftrightarrow$ there is no w s.t. $(m,w) \in$ Tentative

$w \in$ Free W $\leftrightarrow$ there is no m s.t. $(m,w) \in$ Tentative

Obs 3: Tentative $\subseteq$ Invitations and is a matching

Obs 4: $m \in$ FreeM $\rightarrow$ there is a w s.t. $(m, w) \notin$ Invitations

Suppose $m \in$ FreeM but for all w $(m,w) \in$ Invites

So by Obs 1, all w are matched

and by Obs 3, all n machinists are matched

So by Obs 2, no machinist is Free $\Rightarrow\Leftarrow$

When G-S terminates, Tentative is a perfect matching

2 ways alg can terminate: ~~1)~~ There are free machinists who have made all invitations — contradicts Obs 4

2) No free machinists

↳ then all n machinists matched and so all n welders matched (Obs 3)

so matching is perfect

$p \vee q$

$p \rightarrow c$

$\sim p$

q

$q \rightarrow$ no free welders $\rightarrow$ perfect matching

When G-S terminates, Tentative is a stable matching

Suppose not stable — there is an instability (m, w') wrt Tentative. (def stable)

$(m, w') \notin$ Tentative (def instability)

Tentative is a perfect matching, so can find w s.t. $(m,w) \in$ Tentative and m' s.t. $(m',w') \in$ Tentative

m invited w (only way to get match (m,w))

m prefers w' to w (def instability)
w' prefers m to m' (def instability)
m invited w' (m makes invites in ↓ order of pref, and eventually invited w who they don't prefer to w')

for result of that invitation

2 cases:

~~i)~~ rejected immediately – w' already matched with m'' and w' prefers m'' to m

2 subcases : m'' = m' so w' prefers m' to m

m'' ≠ m' so w' prefers m' to m'' (obs 1)

so w' prefers m' to m (transitive)

ii) accepted but revoked later

so by obs 1 again, w' prefers m' to m

$\Rightarrow \Leftarrow$

w's seq of matches

NIL, NIL, ..., m'', m'', ..., m'

m invites w'

Suppose ∃ instability

p ∨ q ∨ r

p → c

q → c

r → c

c

in both cases, w' prefers m' to m

but then (m, w') is not an instability

∴ stable

Machinist Optimality

w is a valid partner for m if $(m,w) \in S$ for some stable matching S

best(m) is m's best valid partner (m's most preferred among m's valid partners)

A	XYVWZ	V	ADCEB	(A, Y)	(A, V)
B	VXWYZ	W	ABDCE	(B, W)	(B, W)
C	VZWYX	X	DECAB	(C, Z)	(D, X)
D	WVXZY	Y	CBAED	(D, V)	(C, Y)
E	XYVWZ	Z	ABDEC	(E, X)	(E, Z)
				stable	also stable (have welders make invitations)

A's valid partners = { Y, V, ... } ← might be more stable matchings, but I know none have (A, X)
best (A) = Y
B's valid parters = { W, ... }
best (B) = W
best (C) = Z
best (D) = V
best (E) = X

G-S always returns $S^* = \{(m, best(m))\}$

Proof: Suppose not. Then there is some execution $\mathcal{E}$ returns S s.t. $(m,w) \in S$ but $w \neq best(m)$ for some m

m invited best(m) before w in $\mathcal{E}$ (invitations in ↓ order of pref)

so there is a rejection by a best valid partner in $\mathcal{E}$
so there is a rejection by a valid partner in $\mathcal{E}$ (valid ⊇ best valid)

consider 1st rejection in $\mathcal{E}$ of a valid partner : w rejects m
then w = best (m)

m must have been rejected by all prev invitees before w; prev invitees are preferred (invites in ↓ order) but none are valid (else not 1st rejection by valid)

w rejects m in favor of some m', so w prefers m' to m (code)

there is some stable matching S' s.t. $(m,w) \in S'$ (w is valid; def valid)

find w' s.t. $(m',w') \in S'$ (S' is a perfect matching)
then $w \neq w'$ ($(m,w) \in S'$, can't also have $(m',w) \in S'$)

w' is a valid partner of m' ($(m',w') \in S'$, S' is stable; def valid)

m' prefers w to w' (w rejecting m is 1st rejection of valid partner)

(m', w) is instability in S' $\Rightarrow\Leftarrow$

∴ G-S always returns S^*

Examples

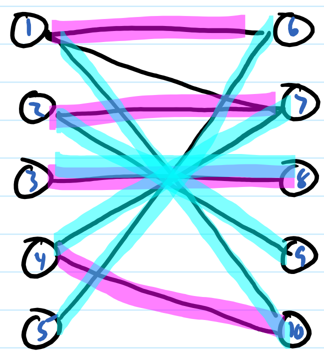
Interval Scheduling: Given n requests with start $s(i)$, finish $f(i)$
find largest set of compatible requests

↳ nonoverlapping - request i compat w/ request j
if $i \neq j$ and
$s(i) < f(i) \leq s(j) < f(j)$
$s(j) < f(j) \leq s(i) < f(i)$

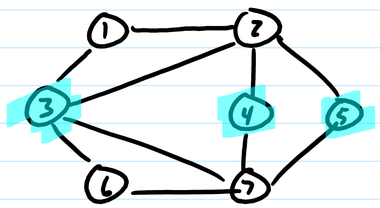
Weighted Interval Scheduling: add weight w_i to each request, find set of compatible requests to maximize total weight

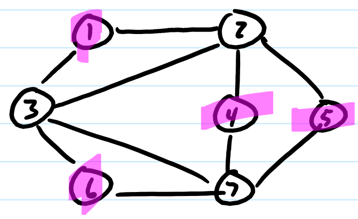
Stable Matching: Gale-Shapley

Bipartite Matching: Given bipartite G, find maximum matching
↳ can split verts into 2 parts S_1, S_2
s.t. all edges between S_1, S_2



Independent Set: Given graph G, find largest set of vertices s.t. no edge between vertices in set





Brute force:
go through all 2^n subsets
keep track of largest IS

Competitive Facility Location: Given G with weighted verts, bound B
game between P1, P2 alternating choosing vert s.t. not adjacent to already chosen,
is there a strategy for P2 to guarantee a total $\geq B$?