Stable Matching Problem

input:

n hospitals : $H = \{h_1, h_2, \ldots, h_n\}$

n students : $S = \{s_1, s_2, \ldots, s_n\}$

$|H| = |S| = n$

preference lists for each hospital and each student

each is n long
2n lists total
$\Rightarrow n^2$ input size

output:

perfect matching that is self-reinforcing (stable)

A set $M \subseteq H \times S = \{(h, s) : h \in H \text{ and } s \in S\}$

subset — cartesian product — set builder notation / set abstraction

is a perfect matching if:

matching

- each $h \in H$ is in at most one pair of M (element of)
- each $s \in S$ is in at most one pair of M

perfect [- $|M| = |H| = |S| = n$.

example: $M = \{(A,X), (B,Z), (C,Y)\}$

A: X Y Z → X: B A C

B: Y ~~X~~ Z   Y: A B C

C: X Y Z   Z: A B C

A perfect matching M is <u>stable</u> if it contains no unstable pairs.

Given M, $h \in H$ and $s \in S$ form an unstable pair if:

- $(h,s) \notin M$
- h prefers s to their current match
- s prefers h to their current match

Is M stable? If not, how many unstable pairs are there?

(B,X)

B is matched to Z but B prefers X

X is matched to A but X prefers B

Is there a stable matching?

A:	X	Y	Z
B:	Y	X	Z
C:	X	Y	Z

X:	B	A	C
Y:	A	B	C
Z:	A	B	C

$M = \{ (A,Y), (B,X), (C,Z) \}$ ✓

How do we know M is a stable matching?

there are no unstable pairs.

how to check? go through every pair $(h,s) \notin M$ and check

$\underbrace{(|H \times S| - |M|)}_{\text{\# total matches - \# in M}}$ • total time to do check

n^2

Can you give me a naive alg. to find a stable matching (if one exists)?

Naive Stable Matching

for every perfect matching M: $n!$

check every pair not in M

if none unstable, return M $\Big] n^2$

~~return "No stable matching"~~

Gale-Shaple Stable Matching:

let M be an empty matching

While there is a hospital h that is not yet matched and has not proposed to every student:

    choose such a hospital h

    let s be the highest-ranked student on h's pref. list to whom h has not yet proposed

    If s is not matched:

        add (h, s) to M

    Else:

        let h' be s's current match in M

        $(h', s) \in M$

        If s prefers h to h':

            remove (h', s) from M and add (h, s) to M