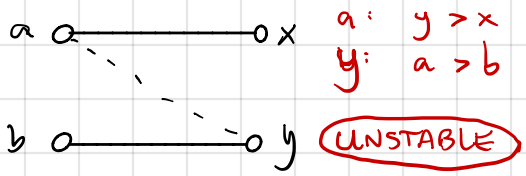
29 Jan 2018

**ADMINISTRATIVE** Office hrs start today.
See info on Piazza and website.

* Change in my office hours *

| | | | | |
|---|---|---|---|---|
| This week | Wed | 10:15 – 11:15 | & Thurs | 1:15 – 2:15 |
| Afterward | Tues | 11 – 12 | & Thurs | 1:15 – 2:15 |




## The Proposal Algorithm

Start with everyone unmatched.

→ preprocessing: create a reverse index for each applicant's list.

While there exists an unmatched employer, $x$
who has not yet proposed to every applicant:

$O(n^2)$ time to create the index.

$x$ "proposes" to the highest ranked applicant it has not yet proposed to, $y$.

if $y$ is matched and prefers its current match to $x$, make no change to the matching.

else $(x, y)$ become matched to each other.
previous partner of $y$ (if any) becomes unmatched.

endwhile

**Fact 1.** The while loop runs for at most $n^2$ iterations.

**Reason:** Once every employer has proposed to every applicant, the test at the start of the loop will not be satisfied.

**Fact 2.** Every iteration of the loop can be implemented to run in $O(1)$ time. (If we do $O(n^2)$ preprocessing steps to build reverse indices.)

**Conclusion:** Entire algorithm runs in $O(n^2)$

# Proof of Correctness. Why does the Proposal Algorithm always output a stable perfect matching?

**Property E.** For any employer, at any stage of the algo., the set of applicants they have proposed to is an initial segment of their ranking of applicants.

**Proof.** (Would be deemed unnecessary in a homework solution.)
Call the employer $x$. Prove the property by induction on $t$, the number of while loop iterations.

IND HYPOTH: After $t$ iterations $x$ has proposed to the first $i(t)$ applicants on its list, for some $i(t)$.

BASE CASE: $t = 0$, $x$ has proposed to no one, so $i(t) = 0$.

IND STEP: Assume true for $t-1$ iterations.

Case 1. The employer chosen in iteration $t$ is some $z \neq x$. Then the set of proposals $x$ has made is unchanged, $i(t) = i(t-1)$.

Case 2. The employer chosen in iteration $t$ is $x$. Then $x$ proposes to highest ranked $y$ that it didn't yet propose to. The position of this $y$ on $x$'s list is $1 + i(t-1)$, by definition of $i(t-1)$. Therefore the ind. hyp. holds true after iteration $t$, by setting $i(t) = 1 + i(t-1)$.

QED.

**Property A.** After the first loop iteration in which applicant $y$ receives a proposal, $y$ remains matched throughout the remainder of the algorithm's execution and the sequence of employers to whom $y$ is matched form an increasing sequence in $y$'s ranking of employers.

**Claim.** At the end of the algorithm, it outputs a perfect matching.

**Proof.** By contradiction. Suppose employer $x$ is unmatched at the end.

Termination condition: $x$ has proposed to every applicant.

Property A: every applicant has received a proposal $\Rightarrow$ they are all matched.

$x$ is unmatched $\Rightarrow$ $< n$ matched employers.

every applicant matched $\Rightarrow$ $n$ matched applicants.

Pigeonhole $\Rightarrow$ some employer is matched to 2 or more applicants.

This contradicts the logic of the algorithm, which maintains an invariant that every employer is matched to $\leq 1$ applicant.
(A super careful proof should establish that invariant using induction too.)

**Claim 2.** The perfect matching that the alg. outputs is stable.

**Proof.** By contradiction. Suppose $(a,x)$ and $(b,z)$ are matched at the end but $a$ prefers $z$ to $x$ and $z$ prefers $a$ to $b$.
($a,b$ are applicants. $x,z$ are employers.)

Property E: $z$ proposed to $a$ during the alg's execution.
Property A: $a$ cannot be matched to $x$ after receiving $z$'s proposal. Contradiction!

**FACT.** Define applicant $a$ and employer $x$ to be "stable partners" if there exists a stable perfect matching in which they are matched.

When you run the Proposal Algorithm every employer gets their favorite stable partner and every applicant gets their least favorite stable partner.

**Proof.** See book.