9/11/18

→ Running Time of the Algorithm:

-Whatever is inside the while loop needs to be done in constant time to get $O(n^2)$ implementation of GS.

- List of free men in queue/linked list.
- Engagements:

  Maintain 2 arrays - wife[m] and husband[w]

  If m engaged to w: wife[m] = w

  husband[w] = m.

- Men proposing:
  - pref[m][w]: for each man, the women are ordered by preference
  - count[m]: Count # of proposal made by m.
- Women rejecting/accepting

  pref[i] = m_3 m_1 m_2 m_4

  (m_2: current, m_4: proposes)

  Looks like she needs to go through the list and see if m_2 is before m_4.

  ↳ But with this we have $O(n^3)$ complexity. Extra factor of 'n' if she has to go through entire array

Solution to this:

Instead of storing preference list, store position of men in list → called INVERSE PREFERENCE LIST

(3) m_3 (1) m_1 (2) m_2 (4) m_4 → pref[i]

⇓

2 3 1 4 → inv-pref[w][m]

↳ inv-pref[w][m]

position of m^{th} man in original preference list

w prefers m to m' then, inv-pref[w][m] < inv-pref[w][m']

→ G-S algorithm is a sexist algorithm in terms of its outcome!

- Two Stable Matchings:

men

X	(A)	(B)	C
Y	(B)	(A)	C
Z	A	B	(C)

women

A	(Y)	(X)	Z
B	(X)	(Y)	Z
C	X	Y	(Z)

⇓ Note: G-S algorithm returns same result not matter which man you start with.

? Which of the two stable matchings is returned by G-S algorithm? The dotted one!

- In GS algorithm, each man ends up with "best possible" partner.

- woman w is a valid partner of man m if ∃ stable matching with m matched to w.
- man-optimal stable matching: every man is matched to his best valid partner.
  unique!

- Claim: GS returns the man-optimal stable matching. (exactly one such)
  [Refer the textbook for a proof]

→ National Residency Matching programs uses a version of GS algorithm.
  ↳ Hospital optimal algorithm before 1995 then switched to student optimal algorithm → works out better for students! (ツ)

ALGORITHM ANALYSIS

Running Time

- "All algorithms work fast for easy inputs but difference becomes most important for hard inputs"
  ↑ WORST CASE PERSPECTIVE

- How worst case running time increases w.r.t 'n'.
- Comparison with natural parameter.

- Some algorithms are efficient than others.
- So we need to compare them w.r.t some natural parameter 'n'.

For example, #men or women in GS

→ $T(n)$ = worst-case running time ≥ 0

Motivation:

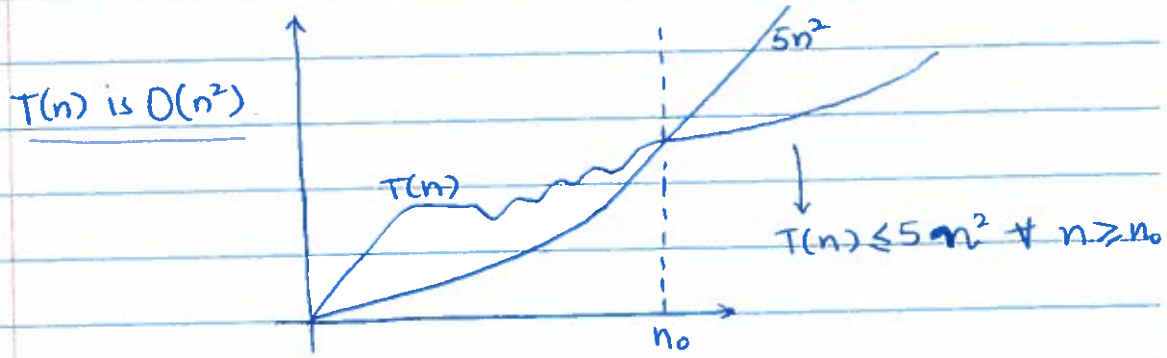
1. $3n^2$ is "order" of n^2
   → ignore constant factors.
2. Don't care about small n.

→ Upper Bound:

ignore constant factors

don't care about small n

Def: $T(n)$ is $O(g(n))$ means $\exists$ $c>0$, $n_0>0$ such that $\forall$ $n \geq n_0$, $T(n) \leq c \cdot g(n)$



$T(n) \in \underbrace{O(g(n))}_{\text{set of functions}}$

⇓

OK to write it as $T(n) = O(g(n))$

⇒ **Example 1:**

$T(n) = pn^2 + qn + r$ where $p, q, r \geq 0$

Claim: $T(n) = O(n^2)$

Proof: Suffices to find $c, n_0 > 0$ such that $\forall n \geq n_0$

$pn^2 + qn + r \leq cn^2$

Let $c = p + q + r$, (can do this as p, q, r are +ve)

$pn^2 + qn + r \leq pn^2 + qn^2 + rn^2 = cn^2$ for all $n \geq n_0$.

Thus, $T(n) = O(n^2)$

? What if $q < 0$?

$c = p - q + r \Rightarrow pn^2 + qn + r \leq pn^2 - qn^2 + rn^2 = (p - q + r)n^2$

$= cn^2$

→ $n^2 = O(n^3)$ so $T(n) = O(n^3)$ also.

→ Asymptotic Intuition: $T(n) = O(g(n))$ "$\leq$"

→ $T(n)$ = polynomial of degree k,

$T(n) = O(n^k)$

⇒ **Example 2:**

Claim: $6n^3 \neq O(n^2)$

Proof: Suppose $6n^3 = O(n^2)$

$\exists\ c, n_0 > 0$ such that $\forall n \geq n_0,\ 6n^3 \leq c \cdot n^2$

$\Updownarrow$

$\forall n \geq n_0,\ n \leq \frac{c}{6}$

But this is true only for $n \leq c/6$, not all $n \geq n_0$ ⇒⇐