Sep 18

Notation change: Assume $M = [n] \stackrel{def}{=} \{1, \ldots, n\}$

$m \in [n]$ → refers to the m^{th} man $\quad W = [n]$

$\{m_1, \ldots, m_n\} \longrightarrow \{1, \ldots, n\}$

→ Array indices start from 1

Q0) How is the input represented?

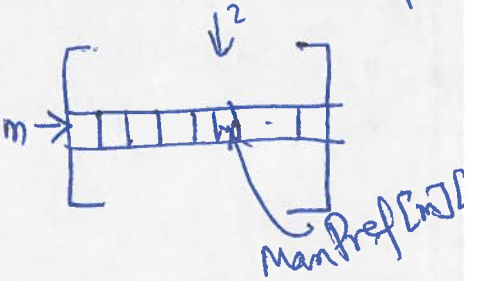
A0] 2D-arrays ManPref WomanPref

ManPref[m][i] = ID of the i^{th} most preferred woman for

WomanPref[w][i] = _____ man fr

Initialization: n/a

Query: Value of a specific entry: $\leftarrow O(1)$



Update: n/a

Q1) How do we find a free woman w?

A1]: Maintain a linked list free women

Init: Add in all n women to free : $O(n)$

Query: Pick say first entry in free : $O(1)$
(+ delete the entry)

Update: $O(1)$
- Case 1: m was free → do nothing
- Case 1: (m, w') were engaged
  - Case 2-1 : (m, w') remain engaged: Add w to linked list
  - Case 2-2 : (m, w) are engaged : Add w' to free

Q2) How do we figure out who w's best unproposed man is?

A2: Maintain array Next s.t. (← such that)
Next[w] = rank of man m that w should propose to next.

Initialization: Next[w] = 1 ∀ w ∈ [n] : O(n)

Query: who should w propose to? Lookup ManPref[w][Next[w]] } O(1)

Update: Next[w]++ : O(1)

Q3) How do we find out who m is engaged to?

A3 Array Current of length n

Current[m] = { -1 if m is free
w (m,w) are engaged.

Initialization: Current[m] = -1 ∀ m ∈ [n] : O(n)

Query: Read Current[m] : O(1)

Update: If (m,w) get engaged : Current[m] = w } O(1)
(otherwise) → 0/w no change

Q4) Is w' > w in Lm?

A4: Scan ManPref[m] & check if w' comes before w.
Issue: O(n) time ⇒ overall O(n³) time.

Q: Do better!

"stich in time saves nine"

Idea: Build a data structure in O(n²) time ⇒ Query/Update (A) ≤ O(1)

[A4]: Create a 2-D array Rank

Rank[m][w] → rank of w in ManPref[m]

Initialization: $O(n^2)$

```
for m = 1..n
    for j = 1..n
        Rank[m][ManPref[m][j]] = j
```

Query: Rank[m][w'] $\overset{?}{<}$ Rank[m][w] : $O(1)$

Update: n/a

Init (1-4) $= O(n) + O(n) + O(n) + O(n^2) = O(n^2)$

Query/Update (1-4) $= O(1) + O(1) + O(1) + O(1) = O(1)$

⇒ Overall: Init (1-4) + n^2 · Query/Update (1-4)

$= O(n^2) + n^2 \cdot O(1) = O(n^2)$.