Revisiting attempt 1 :

$d(i,j,\ell)$ = shortest path b/w i & j with at most ℓ edges.

$$d(i,j,\ell) = \begin{cases} \min_{u:(u,j)\in E} \{ d(i,u,\ell-1) + w_{u,j} \\ d(i,j,\ell-1) \end{cases}$$

→ 3D array. ⇒ $O(|V|^3)$ entries $\quad |V| = n$

$\underline{n^3}$.

For each entry (i,j,ℓ), we do $(\underbrace{d_j}_{\text{Degree of } j} + 1)$ lookups

Degree of j → Worst case: $O(|V|)$ i.e., $\leq n$

→ Worst case : $\underline{O(|V|^4)}$

$O(\underbrace{|V|^3}_{\text{No. of ele.}} \cdot \underbrace{|V|}_{\text{For each entry, } O(|V|) \text{ lookups.}})$

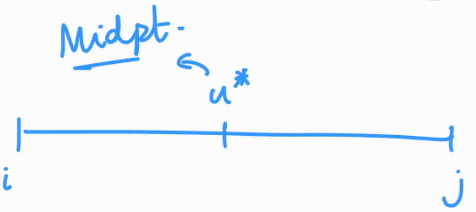
Optimise further?

Why are we choosing $\ell - 1$?

Can also see:

$$d(i,j,\ell) = \min_{u\in V} \{ d(i,u,\tfrac{\ell}{2}) + d(u,j,\tfrac{\ell}{2}) \}.$$

Base case: $d(i,u,1)$ & $d(u,j,1)$ $\forall u$.



Shortest amongst all $u \rightsquigarrow j$ paths with at most ℓ edges

Observation : If π is a shortest path from i to j and u is on it then $\pi|_{i \rightsquigarrow u}$ is also the shortest path from i to u.

(Proof using exchange argument).

Suppose not. $\exists$ path σ from i to u s.t. $d(\sigma_{i \leadsto u}) < d(\pi_{i \leadsto u})$

$\Rightarrow$ This contradicts that π is the shortest path from i to j as we can construct a path with $\sigma_{i \leadsto u} \cdot \pi_{u \leadsto j}$ which has shorter distance.

Let $n = 2^m$

So now $d(\underbrace{i, j}_{n \text{ options}}, \underbrace{\frac{n}{2^k}}_{\log n})$ $\Rightarrow$ 3D array has $n^2 \log n$

And for each entry, we are looking at $2n$ entries.

$\Rightarrow$ Overall complexity decreases to $O(n^3 \log n)$

Pseudocode

BOTTOM-UP APPROACH

```
INIT ← <Do it by yourself>.
for k in [1, log n]:
    for i in V:
        for j in V:
            minValue = ∞
            for u in V:
                val = d(i, u, 2^(k-1)) + d(u, j, 2^(k-1))
                if val < minValue:
                    minValue ← val
        d(i, j, 2^k) ← minValue
```

We shld already have subroutine i.e., $\frac{l}{2}$ before l.

The recursive way is TOP-DOWN APPROACH.

Shortest (i, j, ℓ):
for all u:
shortest $(i, u, \frac{\ell}{2})$
shortest $(u, j, \frac{\ell}{2})$

→ Will have exponential tree.
Computes already computed values again & again.

Suboptimal.

Next approach:

$d(i, j, k)$ Using vertices $\{1, 2, \ldots, k\}$.

To get shortest from i to j, we need $d(i, j, n)$.

(See prev class)

In top down approach, we first start with $k = n$.

Lookups for each entry : 3.

$O(n^3)$: Space.

3 diff. attempts

Space $< n^3$. Lookups $\leq n$

Space $= n^2 \log n$ lookups $\leq n$

Space $\leq n^3$ lookups : 3

* To get the path, maintain another array from which we obtain the min. distance.

- Longest increasing subsequence

n distinct elements.

Sequence: $a_1 \quad a_2 \quad \dots \quad a_n$

Increasing subsequence: $\underbrace{i_1 < i_2 \dots < i_k}_{\text{Indices in 1 to } n.}$ s.t $a_{i_1} < a_{i_2} \dots < a_{i_k}$

Need to find the longest inc. subseq.

Eg: 1 3 2 4 0 5 -2

Inc. subseq.: 1 2 4 5
1 3 4 5
0 5

→ Subseq. of these will also be inc. subseq.

LIS([1, n]) — LIS can contain a_n or not contain a_n.

$$\text{LIS}([1,n]) = \max\{\text{LIS}([1,n-1]),\ \widehat{\text{LIS}}([1,n-1], a_n) + 1\}$$

$\widehat{\text{LIS}}([1,n-1], a_n)$ ← LIS s.t all elements are smaller than a_n

LIS([1, n-1]): LI subseq. doesn't contain a_n

$\widehat{\text{LIS}}([1,n-1], a_n) + 1$: LIS contains a_n. So each of $(n-1)$ ele. must be smaller than a_n.

Pseudocode:

$\widehat{\text{LIS}}([1,i], x)$: → Longest inc. subseq. whose values are smaller than x

```
LIS^([1,i], x):
    if i = 0
        return 0
    m = LIS^([1,i-1], x)
    if a_i < x:
        m <- max{m, 1 + LIS^([1,i-1], a_i)}
    return m
```

→ Returning max of $\{\widehat{\text{LIS}}([1,i-1], a_i)+1,\ \widehat{\text{LIS}}([1,i-1], x)\}$

```
LIS([1,n]):
    return LIS^([1,n], ∞)
```

1	2	3	4	5	6	7
1	3	2	4	0	5	-2

$n = 7$

$\hat{LIS}([1,7], \infty)$

$\downarrow$

$m = \hat{LIS}([1,6], \infty)$

$\downarrow$

$m = \hat{LIS}([1,5], \infty)$

$\downarrow$

$m = \hat{LIS}([1,4], \infty) \longrightarrow m = \hat{LIS}([1,3], 4)$

$\downarrow \uparrow 2$

$m = \hat{LIS}([1,2], 2) \underset{2}{\rightleftarrows} m = \hat{LIS}([1,3], \infty)$

$m = \hat{LIS}([1,1], 2)$ $\uparrow$ 1

$\uparrow \downarrow 0$

$m = \hat{LIS}([1,0], 2)$

$\downarrow \uparrow 2$

$m = \hat{LIS}([1,1], 3) \underset{1}{\rightleftarrows} m = \hat{LIS}([1,2], \infty) \Rightarrow m = 1$ $\quad a_2 = 3$, $3 < \infty$, $m = \max\{1, 1+1\} = 2$

$\downarrow \uparrow 1$

$m = \hat{LIS}([1,1], \infty) \Rightarrow m = 0$ $\quad a_1 = 1$, $1 < \infty$. $m = 1$

$\downarrow \uparrow 0$

$m = \hat{LIS}([1,0], \infty)$

Observation : X takes only the values of the seq. + 1 more

Space : $O(n^2)$ entries & 2 lookups for each entry.

$\Rightarrow$ Arithmetic operation : $O(n^2)$