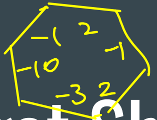
CSE525 Lec12
Graph DP

(Shortest path chapter in JE)

...

Single-source (s) to All-target Shortest-Path

Single source single dest.



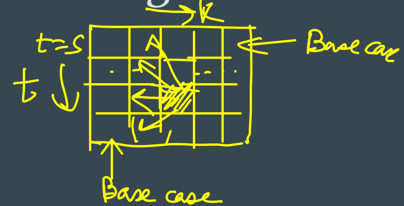
Memo : 2D array
n × m (all edges)
n × (n−1)
Fill column wise.

|V| = n |E| = m

no −ve cycle

Define function f(t,k). f(t, k) = shortest-path dist. from s to t using at most k edges.

shortest path   s ~~ sp(s,t) ~~ t



Case analysis of any such path: (cases should depend upon k) ??
on the length of sp(s,t)

- The path may contain <= k-1 edges. In that case, f(t,k) = ?? f(t, k−1)
- The path actually contains k edges. In that case, f(t,k) = ??

Base case : f(t,1) = { t=s, 0 ; t≠s, W(s,t) }

dist(s,t) = W(s,v) + dist(v,t)

s —•— P —— t   (node v)

Time and space complexity?

f(s,k) = 0

Time: $O(n^2 \times n) = O(n^3)$

sp: O(n) to comp dist

s ~~ t   # edges ≤ n−1

dist(s,t) = f(t, n−1) = f(t, n)

∴ P must be the shortest path from s to t using ≤ k−1 edges.

{ P must be shortest for v ~~ t ; P uses k−1 edges }

f(t,k) = W(s,v) + f(t, k−1)

This is Shimbel-Ford-**Bellman**’s algorithm.

No theoretically-better method is known to solve SSSP.

W(t,t) = 0

s ~~ P ~~ u — t

dist(s,t) = dist(s,u) + W(u,t)

f(t,k) = W(u,t) + f(u,k−1)

$$f(t,k) = \min_{u \in N(t) \,/\, u \in V} \{ W(u,t) + f(u,k-1) \}$$

All-Pairs Shortest-Path

$W(s,t) = \infty$ if no edge

Goal: Compute distance matrix D[u,v] = shortest path distance from u⤳v

g(s, t, k) = shortest-path dist. from s to t using at most k edges

$\forall u,v$ $Dist(u,v) = g(u,v,n-1)$

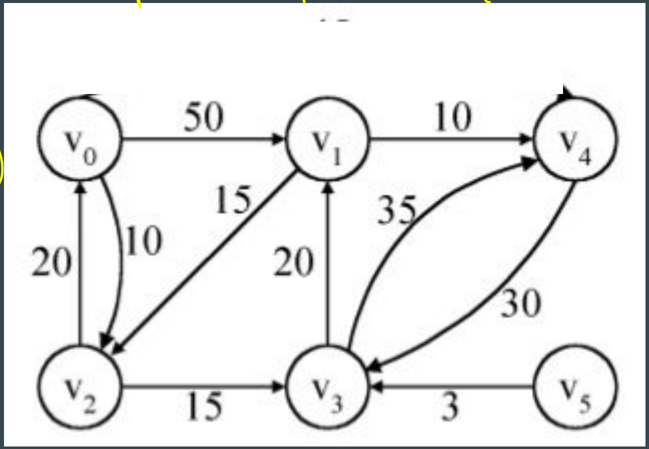
Time compl. :- $O(n^3 \times n) = O(n^4)$

Space compl. :- $O(n^2)$

- What is g(0,4,1) = ? ∞
- What is g(0,4,2) = ? 60
- What is g(0,4,3) = ? 60

Memo: 3d array $n \times n \times (n-1)$

To fill $g(\cdot,\cdot,k)$, need $g(\cdot,\cdot,k-1)$



$$g(s,t,k) = \min_{u \in V} \{ g(s,u,k-1) + W(u,t) \}$$

or $u \in N(t)$

s ⤳ u → t

$$g(s,t,1) = W(s,t)$$

Faster APSP

$g(s, t, k)$ = dist. from $s \rightsquigarrow t$ using <= k edges

Define V x V matrix $H_k[u,v] = g(u,v,k)$

Base case: $H_1 = W$

Compute [illegible], $H_2, \ldots, H_{(n-1)}$

$$H_k[u,v] = \min_z \{ H_{k-1}[u,z] + W[z,v] \}.$$

So, $H_k = H_{k-1} \# H_1 = ((H_{k-2} \# W) \# W) = W \# W \# W \# W \# W \# W \# \ldots \# W$

Define binary operation on matrices: A # B

$(A\#B)[u,v] = \min_{w=1\ldots n} \{ A(u,w) + B(w,v) \}$

Ex. Show # is associative.

((A # A) # A) # A) = A # A # A # A

= ((A # A) # (A # A))

Distance matrix D = $H_{(n-1)}$, $H_{(n)}$, $H_{(n+1)}$, $H_{(n+2)}$

b : next higher power of 2 after n-1

$b = 2^{\lceil \lg_2 n \rceil}$

How to compute H # H # … # H (how many times?)

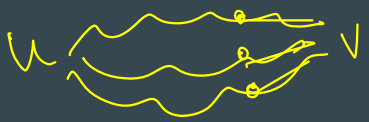
$W \rightarrow W^2 \rightarrow W^4 \rightarrow W^8 \dots W^b = H_{n-1} = D$

Time complexity :- num of # op = lg n

Complexity of (A # B) = $O(n^3)$ } $O(n^3 \lg n)$

Space complexity :- $O(n^2)$

How to obtain shortest paths



Ex:- Extend to faster APSP

Predecessor matrix P[u,v] = predecessor of v on some shortest path on u⇝v ← P[u, v, n−1]

Ex: Compute P from D

Compute shortest path from u ⇝ v using P?

v ← P[u,v] ← P[u,w] ← . . . u
(w = P[u,v])

P[u,v,k] = pred. of v on some s.p. on u⇝v using at most k edges

APSP algo (O(n^4))

$$g(s,t,k) = \begin{cases} g(s,t,k-1) \\ \min_{u \neq t} \{ \ldots \} \end{cases}$$

P[s,t,k] = P[s,t,k−1]

s ⇝ (≤ k−1) → t

P[s,t,k] = u

$$P[u,v,1] = \begin{cases} \text{Null} & \text{if } u \neq v \\ u & \text{if } u = v \end{cases}$$