DYNAMIC

Greedy - predefined procedure to obtain optimal soln

Dynamic - all feasible solns,

Iteration

Tabulation → bottom up approach

Memoisation → Top-down-approach

* Fibonacci. ⇒ $F(n) = F(n-1) + F(n-2)$ $n > 1$

$F(0) = 0$ $F(1) = 1$

1. Knapsack (Maximization problem) (0/1)

↳ $O(nN)$ n: items, N-capacity.

$$F(i,j) = \begin{cases} \max\{F(i-1,j), v_i + F(i-1, j-w_i)\} & j - w_i \geq 0 \\ F(i-1,j) & \text{if } j - w_i < 0. \end{cases}$$

* Optimal subset - most valuable subset.

initial condition:- $F(i,0) = 0$; $F(0,j) = 0$

Bottom-up approach - subproblems to goal

$(n+1)$ rows $(w+1)$ cols:

$(n+1)(w+1) = O(nw)$

Top-down approch

1. Memory function:

while performing fn recursively, it stores some value in memory.

2. Virtual initialization:

initialise all other values other than rth row, col c to -1.

$F(i,j)$ i: set of items j: capacity.

(2) Transitive closure:

v_i - list of intermediate vertices each numbered not higher than k, v_j

v_i — (v_k) — v_j

$r_{ik}^{k-1} = 1$, $r_{kj}^{k-1} = 1$

Hence, $r_{ij}^{(k)} = r_{ij}^{(k-1)} = 1$ or $r_{ik}^{(k-1)}$ and $r_{kj}^{(k-1)} = 1$

WARSHALL:

complexity: $O(n^3)$

$W_{ij}(k) = W_{ij}(k-1)$ ....

(or)

$W_{ij}(k) = W_{ik}(k-1)$ & $W_{kj}(k-1)$

$\Rightarrow W_k = 1$

(3) FLOYD'S (All pairs shortest)

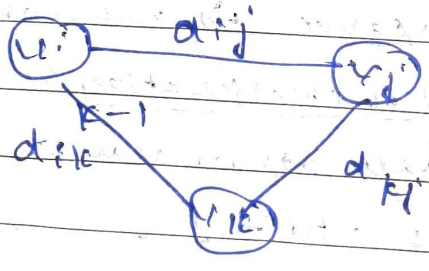
dist b/w A + A / B - B = 0

i) if intermediate vertex:

$d_{ij}^{k} = d_{ik}^{k-1} + d_{kj}^{k-1}$

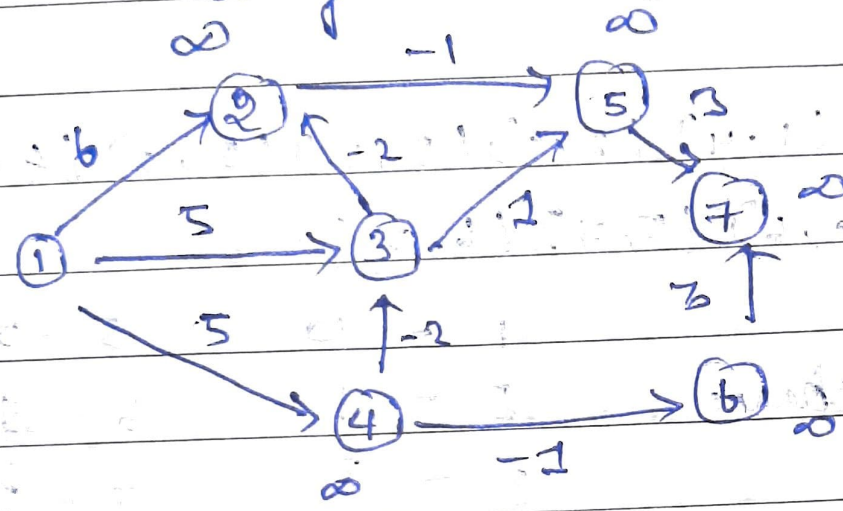
ii) No intermediate: $d_{ij}^{k} = d_{ij}^{k-1}$

To find shortest, choose min among below



* Bellman-Ford Algorithm

single source shortest path



→ works even on -ve edges correctly.

relax vertices of $n-1 = 6$ $(|V|-1)$

(1) select any edge first, make a list of edges,
(1,2) (1,3) (1,4) (2,5) (3,2) (4,3)
(4,6) (6,7) (5,7) (3,5)

(2) Relax all edges $|7|-1$ times = 6 times
mark distances

1 - 0

2 - ~~∞~~ ~~6~~ 3

3 - ∞ 5

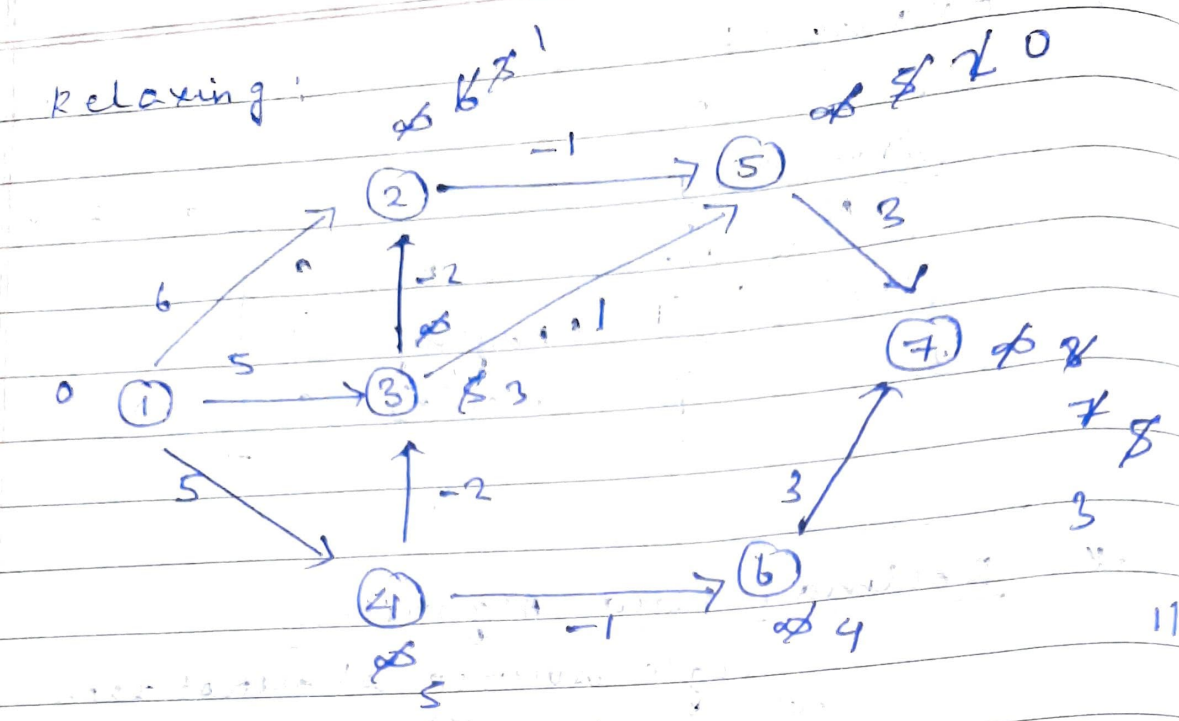
4 - ∞ 5

5 - ∞ 5

6 - ∞

7 - ∞

Relaxing:



(1,2) (1,3) (1,4) (2,5) (3,2) (3,5) (4,3)
(4,6) (5,7) (6,7)

thus, we have

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

Here, no. of relaxations = $O(|E||V|-1)$
$= O(|V||E|)$
$\approx O(n^2)$

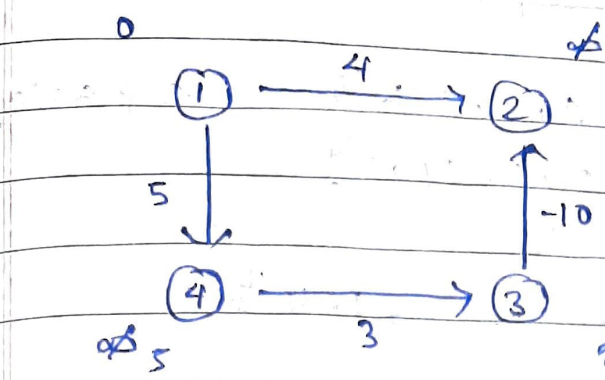
in complete graph [Edge b/w every pair of vertex]

thus we have $|E| = \frac{n(n-1)}{2}$

$$O(VE) = \frac{n(n-1)}{2}(n-1) = \boxed{O(n^3)}$$

Thus, best: $O(n^2)$
worst: $O(n^3)$

* Example:



∞ 4 - 2

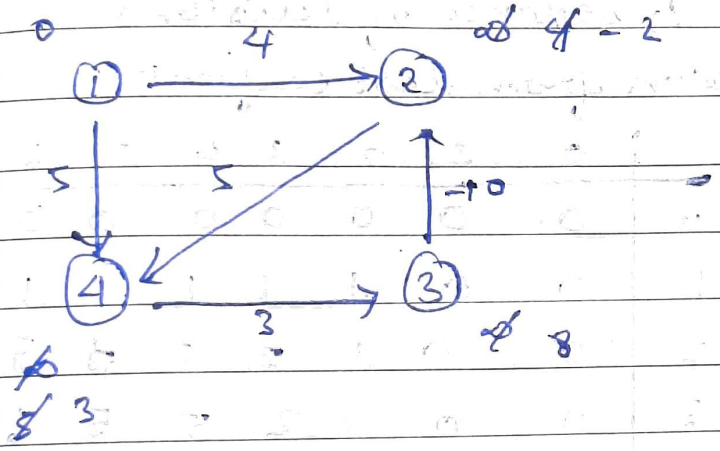
n = 4 - 1 = 3

3 relaxations

1 - 0
2 - -2
3 - 8
4 - 5

list : (1,2) (1,4) (3,2) (4,3)

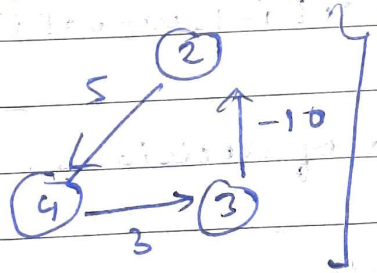
* drawback:



(3,2) (4,3) (1,4) (1,2) (2,4)

Here, even after n-1 no: of relaxations, values of edges change upon relaxation ↓

because we have ~~more than~~ 1 cycle of edges in the graph given; sum as -ve.



→ (-2) → which means its edges will reduce continuously

↓

-ve weight cycle.

Formula :

$$v[i, w] = \max \begin{cases} v[i-1, w], \\ v[i-1, w - w[i]] + p[i] \end{cases}$$