Dynamic Programming

1_ Intuition

- Overlapping Subproblems
- Optimal Substructure
- Induction

2_ "Structure"

1. We solve smallest (trivial) subproblem optimally "$k=0$" (basis step)
2. Then, by going through (all) the ($i \in [0, k]$) previous solutions and analyzing them, find how we can guarantee building the optimal solution for step $k+1$ "$k \xrightarrow{?\ \lim k?} k+1$" (Induction Hypothesis)
3. With that relation, implement finding the optimal solution for any step $k+1$, by recursively resolving and storing the answer to subprob $[0 \ldots k]$ into a matrix. (Inductive step) "$\forall k \geq 0: S(k) \rightarrow S(k+1)$" "$\forall k \geq 0: S(k)$"

Example : ⌜All pairs Shortest path problem⌟ (Floyd Algo)

Given a DWG (directed weighted Graph)
find shortest path between every vertices (v_i, v_j) in G.

$G = (V, E)$ $V = [\![1, \overset{= |V|}{n}]\!]$ $L \in \mathbb{M}_{n \times n}$, $L(i,j) = \begin{cases} 0, & i = j \\ \infty, & \text{no edge} \\ \omega(i,j), & \text{otherwise} \end{cases}$

$\omega(i,j) :=$ weight of edge (i,j).

$D \in \mathbb{M}_{n \times n}$, $D(i,j)$: shortest path from v_i to v_j

things to check:

- Opti Sub prop. holds? (i.e. if X is opti sol for k, then X holds the opti sol for all sub $0 \dots k-1$)

  ↳ Assume p is the shortest path from v_i to v_j, (i.e. p is optimal $= D(i,j)$)

  if $v_k \in p \to (v_i, \dots v_k)$ is optimal $\wedge$

  $(v_k, \dots, v_j)$ is optimal $(p = (v_i, \dots, v_k, \dots, v_p))$

  ($\exists$ shorter path from $v_i \to v_k$ i.e.

  $D(i,k) \neq (v_i, \dots, v_k) \to \exists p', v_k \notin p' \wedge \underbrace{p' = D(i,j) < p}_{\text{contradiction}}$)

- Triv resol? ("$k = 0$") Basis Step

  Here we can just initialize D to L since the B.S. will be

(B.S) all the path for all adjacent nodes (ie. if $(i) \xrightarrow{x} (j)$ then the shortest path is just "given by the edge" (length 1, $\omega = x$)).

  ↳ "skipped"

Approach:

(I.H) Assuming we filled

find a recurrence relation to compute (Strong induction)

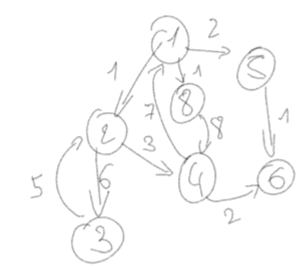
here a step k doesn't rep state of an int but a matrix D

i.e. $D_0, D_1 \dots$ between 2 steps we iterate through whole matrix.

And at step k, we stored the optimal path from i to j (for all pairs (i,j) i.e. whole mat) using only nodes from $[1 \text{ to } k]$.

i.e. at step $k+1$ we check for all paths from i, to j if its shorter to pass by $k+1$ instead. i.e. if the path from i to $k+1$, and from $k+1$ to j is shorter than the current one from i to j.

We don't have the pblm "where to put the stop to $k+1$" bc we rec store the crt best path from $i \to k+1$ and $k+1 \to j$ and we completely replace former by this one



$$\begin{array}{c} j \quad i=1, j=3 \\ \downarrow \quad 1,3,\ D(1,2) \\ \begin{pmatrix} 1 & 2 & 3 & 4 \\ & & & \\ & & & \end{pmatrix} \end{array}$$

$$\begin{pmatrix} 0 & 1 & 3 \\ \infty & 0 & 6 \\ \infty & 5 & 0 \end{pmatrix}^{[1,2,3]}$$

$k=2 \Rightarrow$ only use nodes from 1 to 2 !

$$D(1,3) - \min_{k \le 2} (D(1,3), D(1,k) + D(k,3))$$

$$= D(1,3),\ D(1,2) + D(2,3)$$

e.g. with $n=5$

$$D(1,4) = \min_{\forall k} (D(1,4),\ D(1,k) + D(k,4))$$

$k=1$: $D(\cancel{1,1}) + D(1,4) = D(1,4)$

$k=2$: $D(1,2) + D(2,4)$

$k=3$: $D(1,3) + D(3,4)$

$k=4$: $D(1,4) + \cancel{D(4,4)}$

s.p between i,j at step k is min length of all path starting from i and ending in j, using only nodes 1 to k.

Let S_{ij} the final optimal path from i to j

At each step k (for whole mat i.e. $\forall (i,j) \in \mathbb{N}^2$):

- either $v_k \notin$ optimal path S_{ij} : change nothing in $D(i,j)$
- $v_k \in S_{ij}$: $D[i,j] = D[i,k] + D[k,j]$

So instead of implementing smth to check wether $\sigma_k \in S_{ij}$
we just take $\min(D[i,j], D[i,k] + D[k,j])$
and repeat for all k. (So also for all (i,j))

The rec relation becomes:

$$\begin{cases} D_{k+1}(i,j) = \min\left(D_k[i,j], D_k[i,k+1] + D_k[k+1,j]\right) \quad \forall (i,j) \in [\![0, n-1]\!]^2 \\ D_0 = L \in \mathbb{M}_n \end{cases}$$

$\Rightarrow$ Exactement comme le COINS problem:
Comment obtenir le path qu'on veut
en combinant seulement tous les opti paths qu'on a déjà
trouvé ? $\Rightarrow$ Au début seulement avec les edges $(x, x+1)$
est-ce que faire $x \to k \to x+1$ plus court que $x \to x+1$!
puis on a des paths de $+$ en $+$ long et on peut finir $k-1$ et restart

It's not bc $D(x,y)$ contains a value that it's the s.p. it's only if $k=n$ ie. end of algo/recursion. $D(x,y)$ can contain "temporary best" value that will get corrected later

Because the s.p. between i,j is fully correct only at step $k=n$, all the paths between pairs are built (checked for optimality) gradualy ie. up to step K for each K.

Meaning the cost $D_K(i,j)$ is <u>not</u> optimal for $k<n$ but the path being constructed in P (array of s.p.) ~~is optimal~~! (up to K). (no thats why we iterate through whole mat at each step)

The "real" induction is on the paths of length K being constructed at the same "level" for each (i,j). And since the $P_K(i,j)$ isn't complete then its cost $D(i,j)$ is wrong,

At each step K we traverse the entire matrix to see if passing by K on the road from i to j is shorter (for all i,j bc whole mat)

Shortest path between 1, k is D(1, k), the s.p. (short. path) between (k, k+1) is either (k, k+1) (if it exists and is the min) or $\min_{\forall j}(d(k,j) + d(j, k+1))$, i.e. the min of the length of all path starting from k and ending to k+1. (i.e. trying all the intermediary steps possible and taking the best option after having seen all of them).