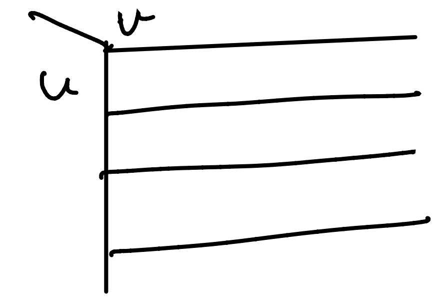
All Pairs Shortest Problem:



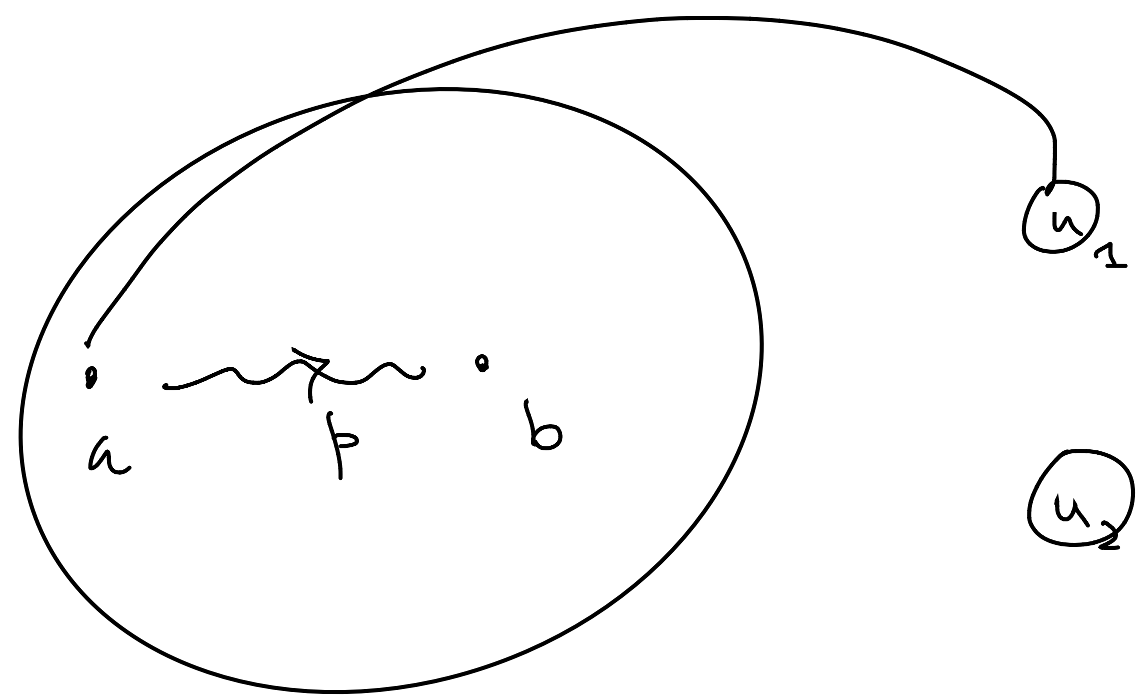
- Table of $\delta(u, v)$, $u, v \in V$

- Run single source shortest Path

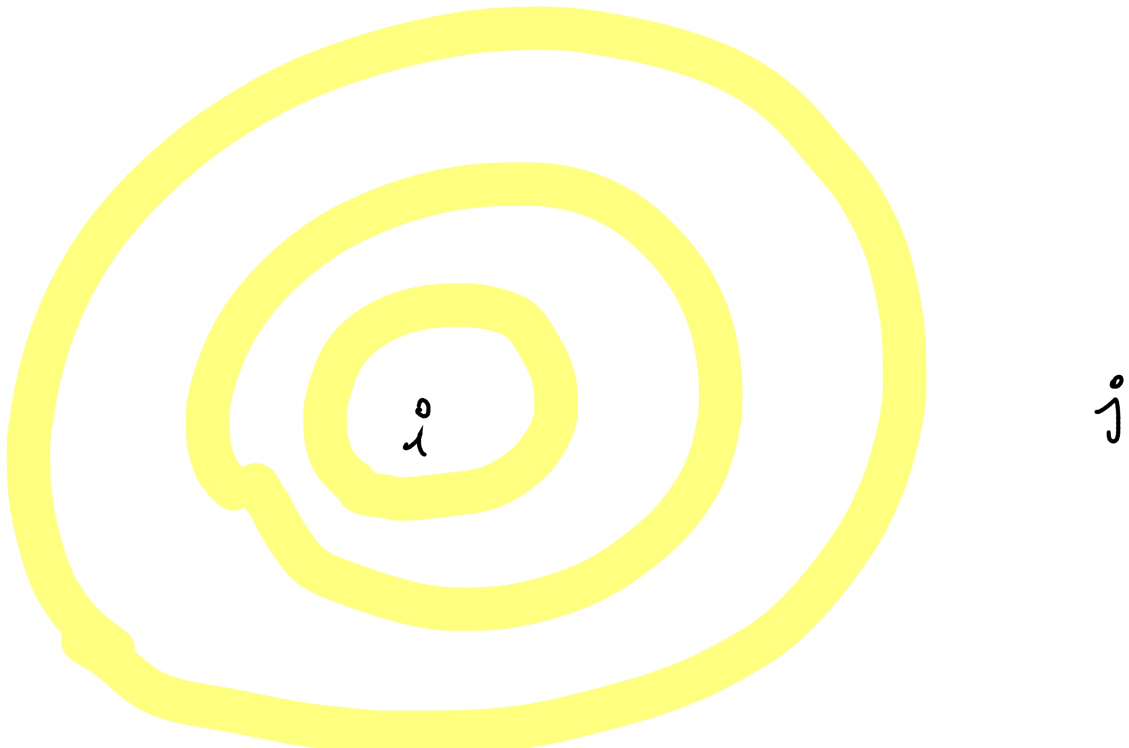
(For general case) — Bellman Ford $O(V^2 E) = O(V^4)$

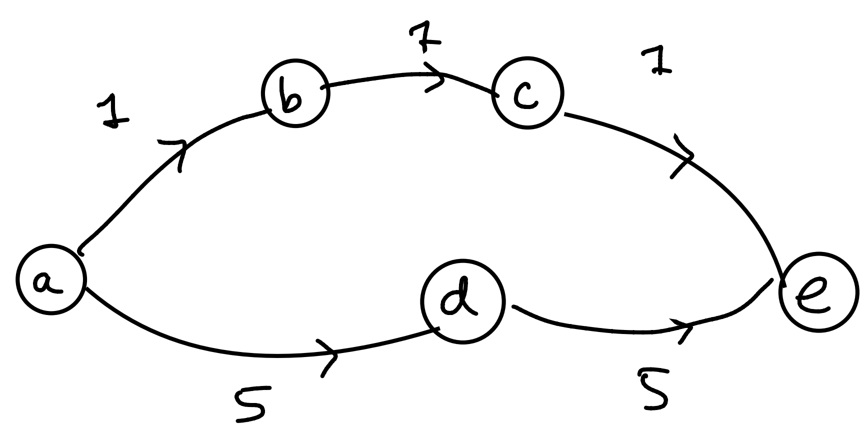
$O(V^2)$

(For non-negative edge case) — Dijkstra $O(VE \log V)$



$l_{ij}^{(m)}$: minimum weight of any path from i to j that contains utmost m edges.





$$l_{ae}^{(2)} = 10$$

Relaxation

$$l_{ij}^{(m)} = \min_{1 \le k \le n} \{ l_{ik}^{(m-1)} + w_{kj} \}$$

What is the best way to come to j

ALL-PAIR-SHORTEST-PATH (L, W) (n : number of vertices)

$L' = (l'_{ij})_{n \times n}$

for $i \leftarrow 1$ to n

do for $j \leftarrow 1$ to n loop 2

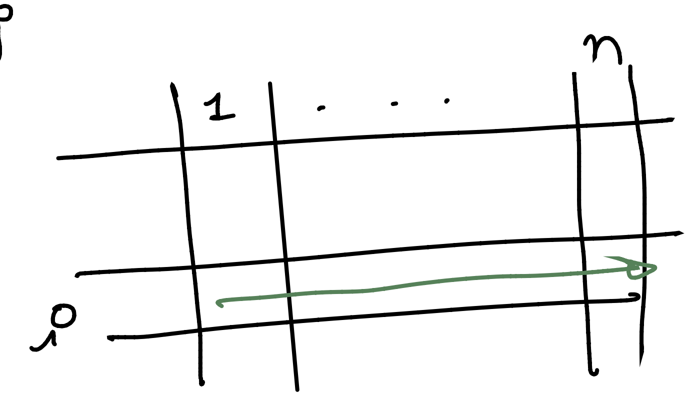
do $l'_{ij} \leftarrow \infty$

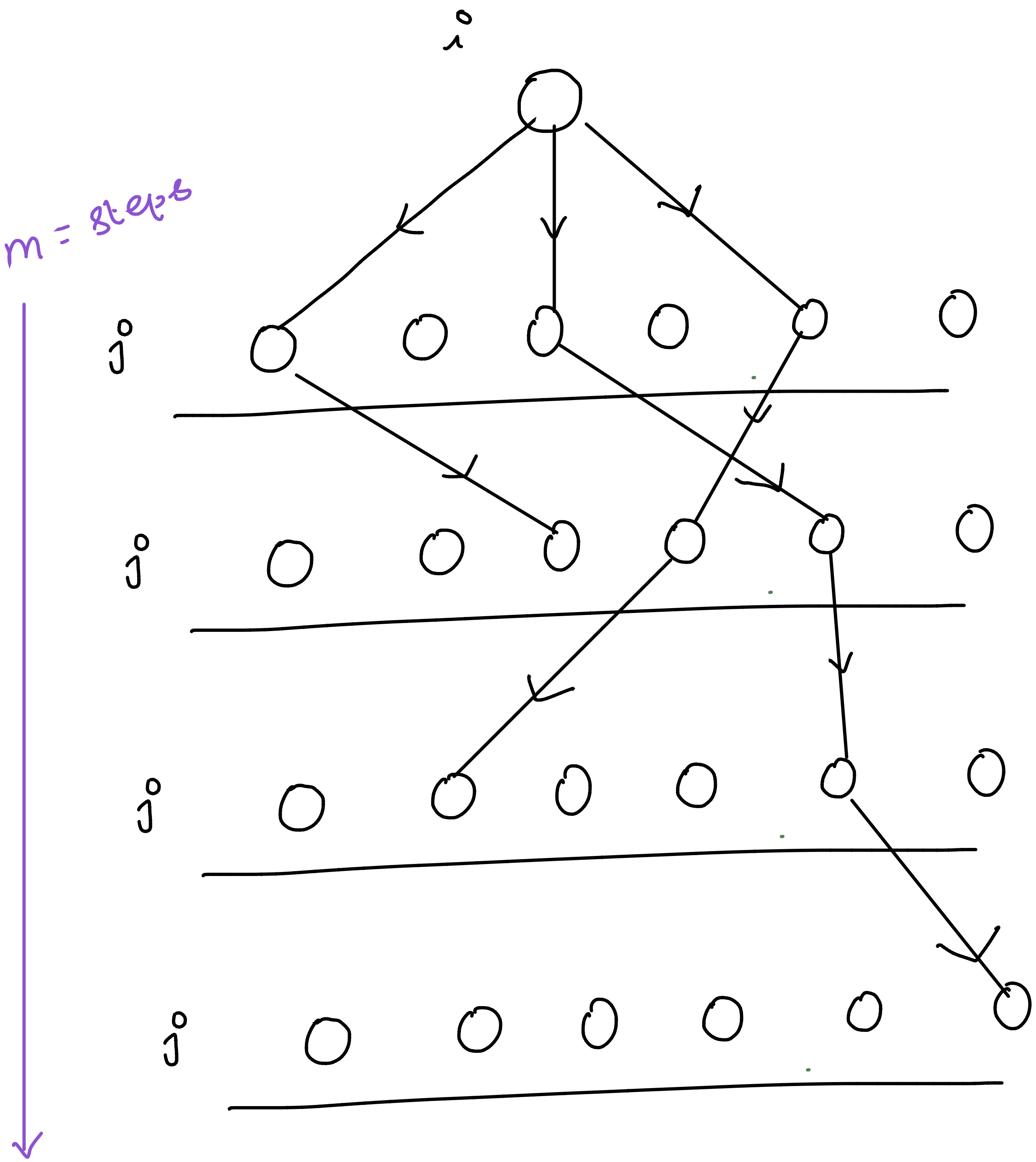
for $k \leftarrow 1$ to n loop 3

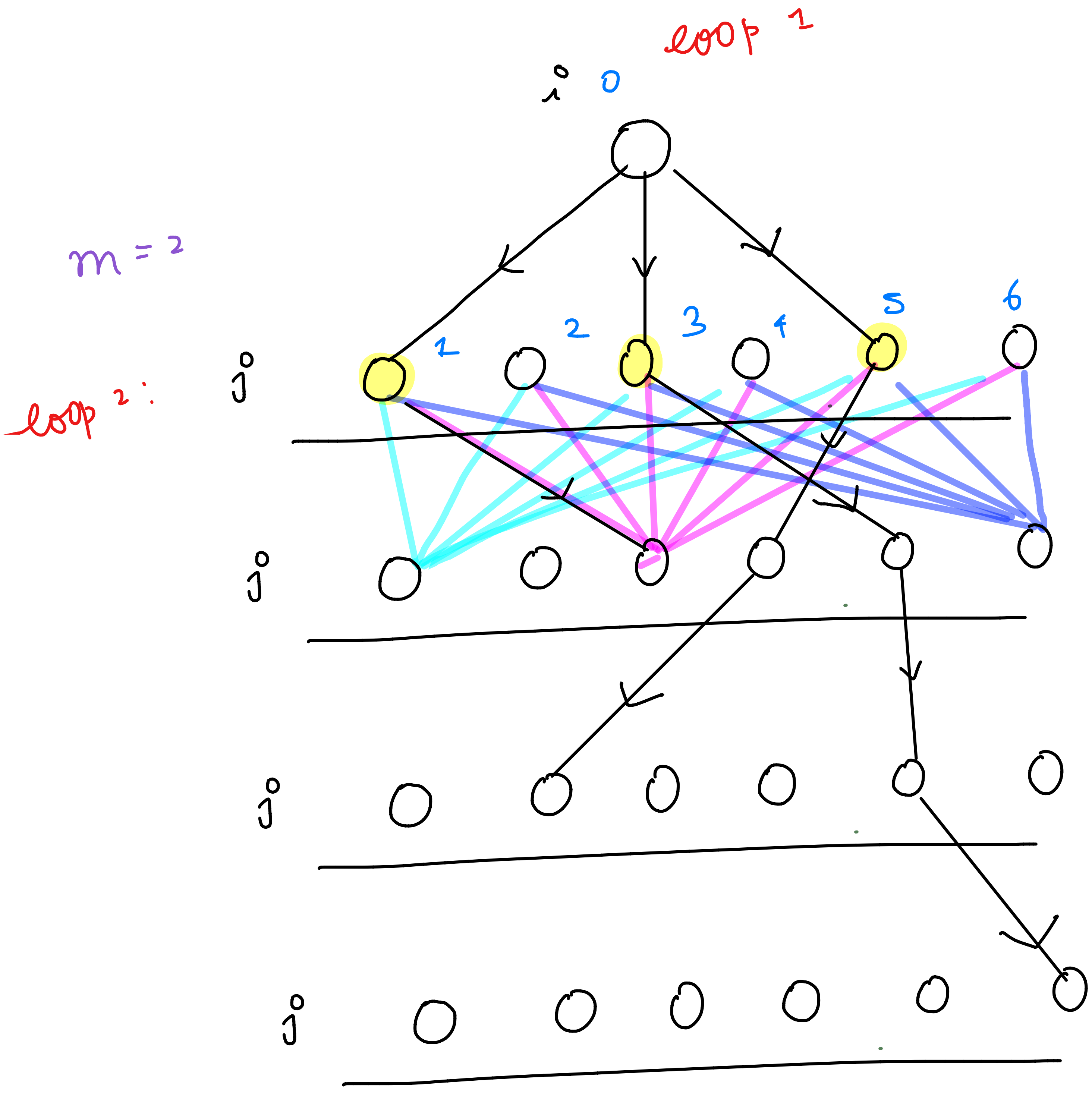
do $l'_{ij} \leftarrow \min(l'_{ij}, l_{ik} + w_{kj})$

Output L'

Imagine some Graph and fix some i and j







Backward stepping

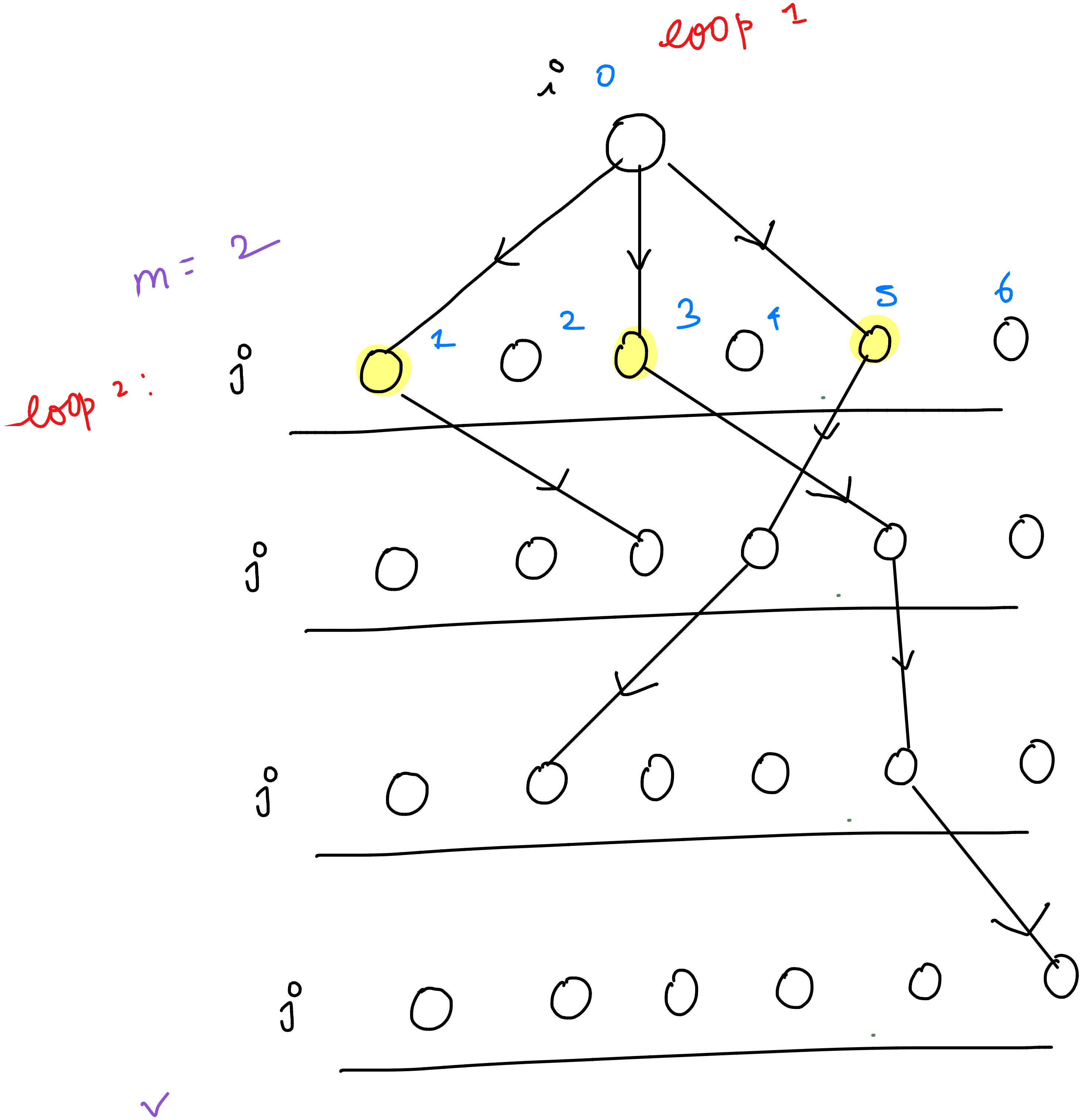
asking a destination

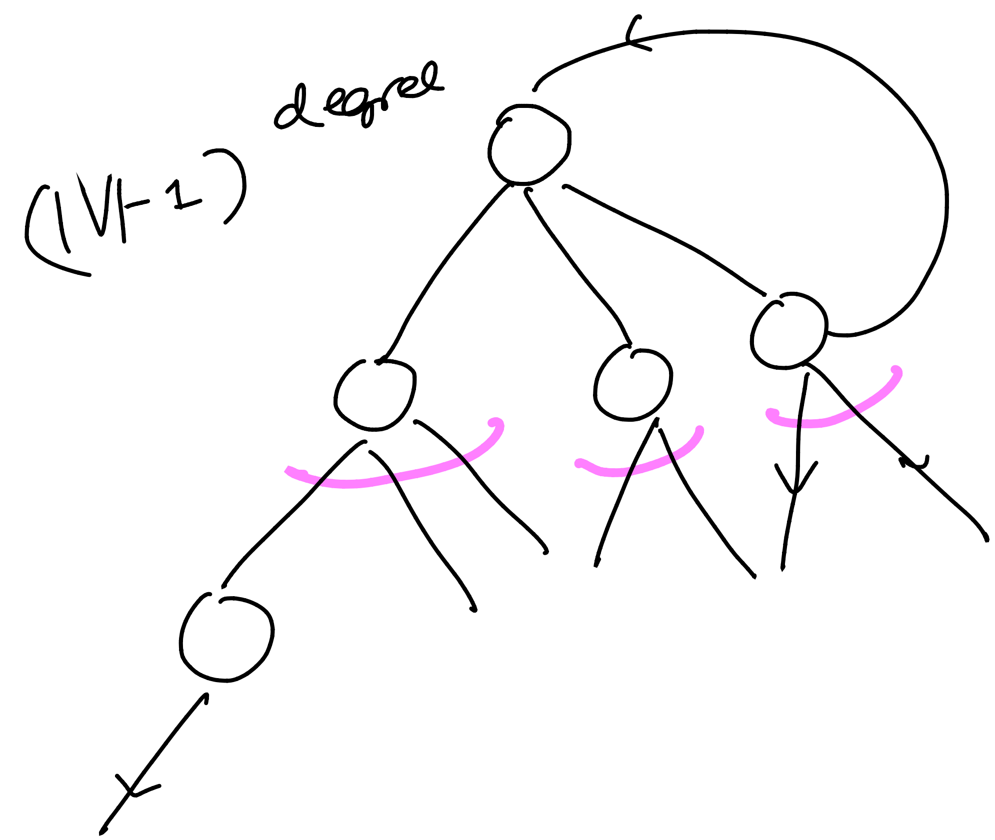
question

Forward Stepping.

Say Bellman Ford

Expanded Based on vertex and Adj[v] like Tree search.

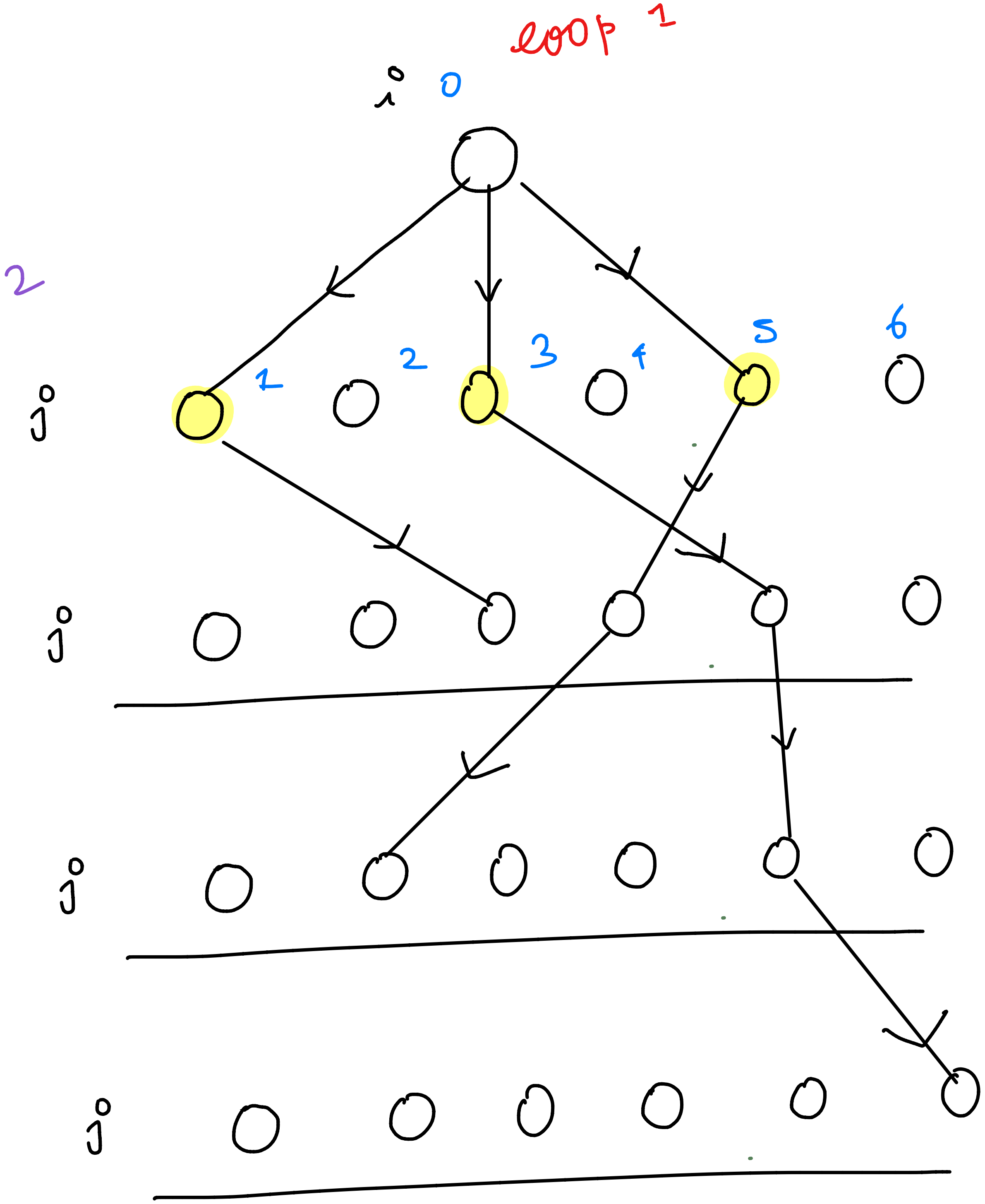




$L = W$

$m = 2$

loop 2:



Connect this to matrix multiplication

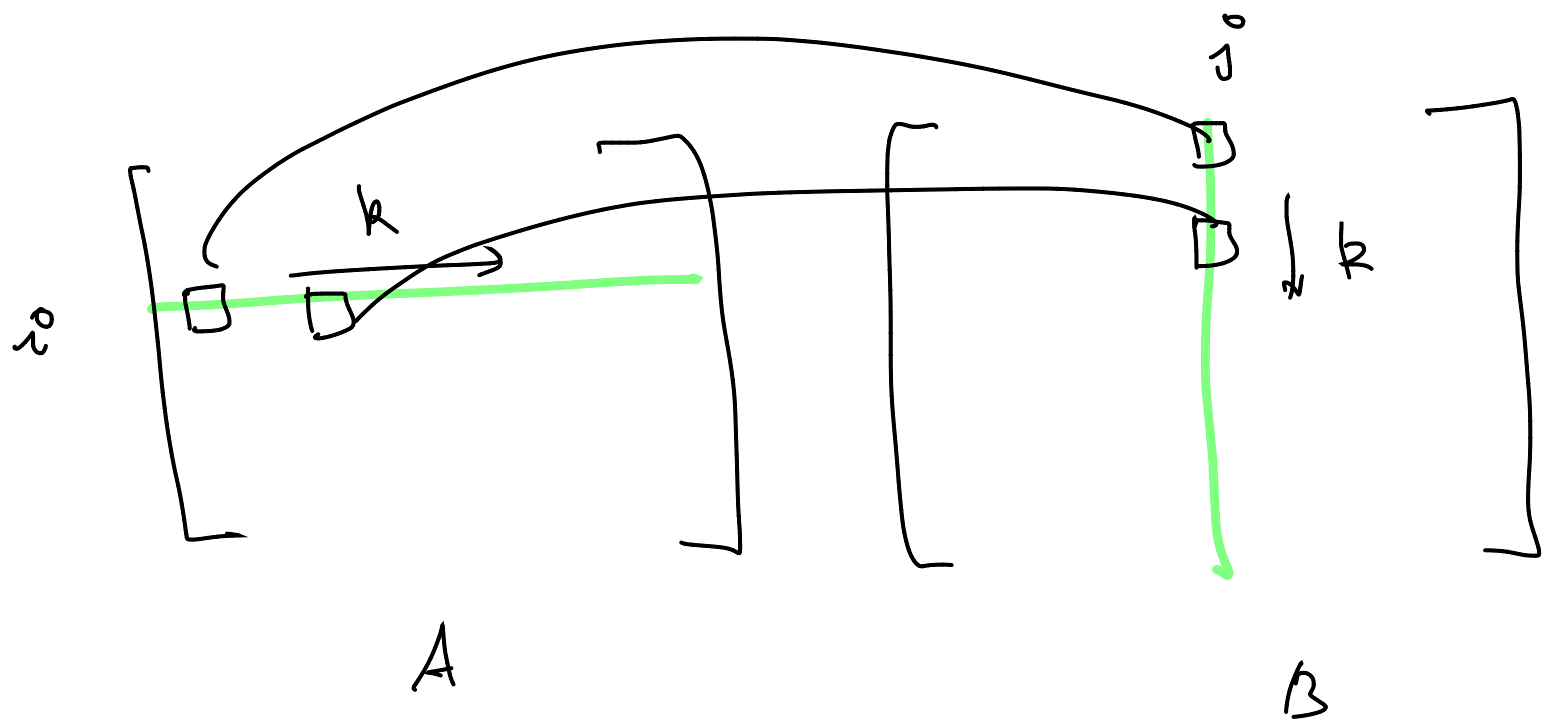
$A = (a_{ij})$, $B = (b_{ij})$, $C = (c_{ij})$

$$C = AB, \quad c_{ij} = \sum_{k=1}^{n} a_{ik} b_{kj}$$

$$= a_{i1} b_{1j} + a_{i2} b_{2j} + \dots + a_{in} b_{nj}$$

$$l_{ij}^{(m)} = \min_{1 \le k \le n} \{ l_{ik}^{(m-1)} + w_{kj} \}$$

$$\min(a_{i1} + b_{1j}, \; a_{i2} + b_{2j}, \; \dots, \; a_{in} + b_{nj})$$



ALL-PAIR-SHORTEST-PATH (L, W)

(n : number of vertices)

$C \leftarrow L' = (l'_{ij})_{n \times n}$

for $i \leftarrow 1$ to n

do for $j \leftarrow 1$ to n loop 2

do $l'_{ij} \leftarrow \infty$ $\rightarrow \infty \rightarrow 0$

for $k \leftarrow 1$ to n loop 3

do $l'_{ij} \leftarrow \min(l'_{ij}, \underbrace{l_{ik}}_{a_{ik}} + \underbrace{w_{kj}}_{b_{kj}})$

min $\rightarrow$ +

+ $\rightarrow$ ×

output L'

$\Downarrow$

ALL-PAIR-SHORTEST-PATH (L, W)

$L' = (l'_{ij})_{n \times n}$ (n : number of vertices)

for $i \leftarrow 1$ to n

do for $j \leftarrow 1$ to n loop 2

do $c_{ij} \leftarrow 0$ (via loop)

for $k \leftarrow 1$ to n loop 3

do $c_{ij} \leftarrow c_{ij} + a_{ik} b_{kj}$

output L'

SLOW VERSION → $L^{(1)} = W$, $L^{(2)} = L^{(1)} \cdot W$, $L^{(3)} = L^{(2)} \cdot W$

$L^{(m)} =$ ALL-PAIR-SHORTEST-PATH $(L^{(m-1)}, W)$

Stop at $L^{(n-1)} = L^{(n)} = L^{(n+1)} = L^{(n+2)} \ldots$

$L^{(n-1)}$ 2 4 8 9

$L^{(2)} = L^{(1)} \cdot L^{(1)}$

$L^{(4)} = L^{(2)} \cdot L^{(2)}$

FAST-ALL-PAIR-SHORTEST-PATH (W)

$L^{(1)} \leftarrow W$

$m \leftarrow 1$

loop 4 → for $m < n-1$

do $L^{(2m)} \leftarrow$ ALL-PAIR-SHORTEST-PATH-STEP $(L^{(m)}, L^{(m)})$

$m \leftarrow 2m$