

Eg. 3. Shortest paths with -ve wts.

- Recall that $G = (V, E, w)$ is a directed weighted graph with source vertex $s \in V$.

We have to compute $\{d(s, v) \mid v \in V\}$ the set of shortest distances (& paths $P(s, v)$).

▷ Dijkstra solves the problem in $O(m + n \log n)$ time, assuming non-negative weights.

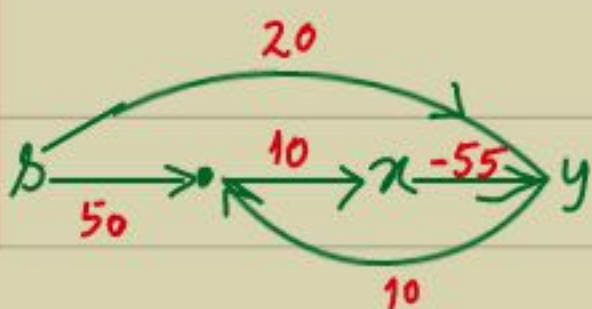
- The non-negative wts. are critical in the analysis.

What to do with negative wts.?

▷ A negative cycle means that distance can be as small as $-\infty$, (though the path is not simple)

- We will assume that there is no -ve cycle.

Bad eg.



▷ Subpath of $P(s, y)$ is not optimal!

$\therefore d(s, y) = 5$ while $d(s, x) = 40$.

Theorem: If $G = (V, E, w)$ has no negative cycle then shortest paths possess optimal substructure property.

Proof:

• Suppose the path $P(s, y)$ violates the optimal subpath property.

$\Rightarrow \exists x \in P(s, y)$ s.t. distance of s to x along $P(s, y)$ is $> \delta(s, x)$.

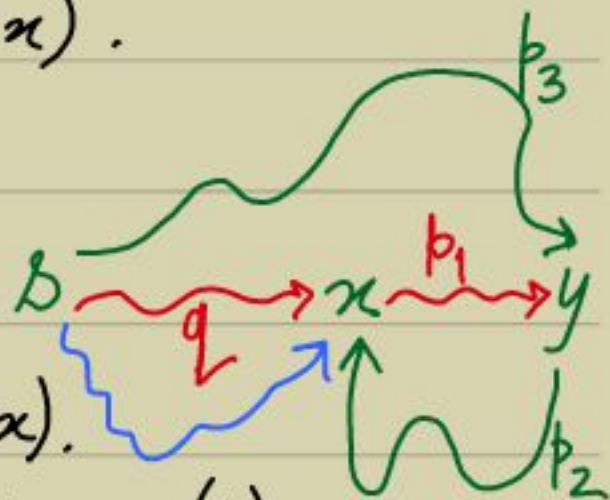
• Note that $P(s, x)$ has to contain y .

• Let the red path be

$P(s, y)$ & the green be $P(s, x)$.

• $w(q) > \delta(s, x) = w(p_3) + w(p_2)$.

• But, $w(q) + w(p_1) < w(p_3)$.



$\Rightarrow w(p_1) + w(p_2) < 0$.

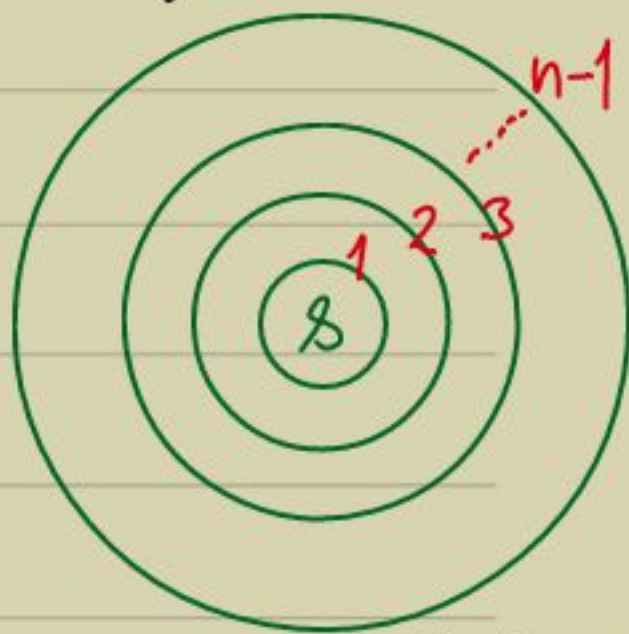
$\Rightarrow \exists$ a negative cycle. □

- This makes the problem amenable to a recursive formulation.

The idea of Bellman & Ford:

- Consider paths $s \rightsquigarrow v$ with $\leq i$ edges & define $P(v, i)$ to be a shortest one, Define $L(v, i)$ to be its length.

▷ Since we only consider simple paths, $s \rightsquigarrow v$ will have $< n$ edges.



- Let us attempt a recursive formulation:

Case 1: $P(v, i)$ has $< i$ edges:

$$\underline{L(v, i)} = L(v, i-1).$$

Case 2: $P(v, i)$ has i edges:

$$\underline{L(v, i)} = \min_{(x, v) \in E} L(x, i-1) + w(x, v).$$

$$\Rightarrow \underline{L(v, i)} = \min(L(v, i-1), \min_{(x, v) \in E} L(x, i-1) + w(x, v)).$$

Base case: $L(s, 1) = 0$ & other $L(v, 1) = \begin{cases} w(s, v), & \text{if } (s, v) \in E \\ \infty, & \text{else.} \end{cases}$

Bellman-Ford ($s, G=(V, E, w)$) {

For each $v \in V \setminus \{s\}$

if $(s, v) \in E$ $L(v, 1) \leftarrow w(s, v);$

else $L(v, 1) \leftarrow \infty;$

$L(s, 1) \leftarrow 0;$

For $i = 2$ to $n-1$

For each $v \in V$ {

$L(v, i) \leftarrow L(v, i-1);$

For $(u, v) \in E$

$L(v, i) \leftarrow \min(L(v, i),$
 $L(u, i-1) + w(u, v));$

}

}

Exercise: Modify it to find $P(v, i)$.

Theorem (Bellman, Ford 1958): Given $(s, G=(V, E, w))$

without negative cycles, we can compute
 $\{P(s, v) \mid v \in V\}$ in $O(mn)$ time.

Proof:

- Dynamic programming implementation fills an $n \times n$ matrix $P := ((P(v, i)))$.

- The matrix update shall take time $O(n \cdot \sum_v \deg v) = O(nm)$.
- Correctness is left as an exercise.
- Finally, $\{L(v, n-1) \mid v \in V\}$ is the output. \square

Exercise: It can be done in $O(n^2)$ space.

Exercise: Can you detect negative cycles?

▷ Shortest paths in the presence of negative cycles is NP-hard.

Ex. 4. All-pairs shortest paths (faster)

- Dijkstra takes: $O(m+n \lg n)$ for single-source
 $\Rightarrow O(mn+n^2 \lg n)$ for all-pairs.

- Bellman-Ford takes: $O(mn)$ for single-source
 $\Rightarrow O(mn^2)$ for all-pairs.

- Can it be solved faster?

Negative weights allowed but no negative cycle.

How to exploit substructure?

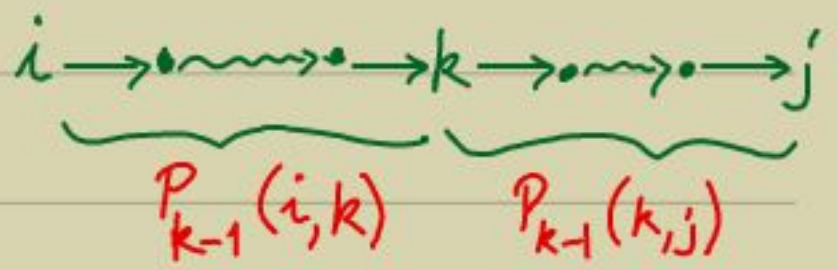
- Floyd-Warshall's idea: To find a shortest path between vertices i & j , consider the max-index vertex k in such a path.

Defn: $P_k(i, j)$ is a shortest path $i \rightarrow j$ with the max-vertex k in between.

$D_k(i, j)$ is its length.

- The recursive formulation is easy.

- Case 1: $P_k(i,j)$ indeed passes k .



$$\Rightarrow D_k(i,j) = D_{k-1}(i,k) + D_{k-1}(k,j).$$

- Case 2: $P_k(i,j)$ does not pass k .

$$\Rightarrow D_k(i,j) = D_{k-1}(i,j).$$

$$\triangleright \underline{D_k(i,j)} = \min(D_{k-1}(i,j), D_{k-1}(i,k) + D_{k-1}(k,j)),$$

for $i, j \in [n]$ & $k \in [0 \dots n]$.

Adjacency matrix essentially

$$\triangleright \text{Base case: } \underline{D_0(i,j)} = \begin{cases} w(i,j), & \text{if } (i,j) \in E \\ 0, & \text{if } i=j \\ \infty, & \text{else} \end{cases}$$

- What is the "matrix" in the dynamic programming implementation?

For each k use an $n \times n$ matrix.

- Qn: Can we manage with one matrix?

- There is an amazing reason:

Observation: 1) The k -th row/column of matrix

D_{k-1} does not change, i.e.

$$D_k(i, k) = D_{k-1}(i, k) \text{ \&}$$

$$D_k(k, j) = D_{k-1}(k, j).$$

2) For $i, j \in [n] \setminus \{k\}$, the (i, j) -th entry of matrix D_{k-1} changes only based on its k -th row/column.

- This yields a simple pseudocode:

FloydWarshall($G = (V, E, w)$) {

 For $i = 1$ to n

 For $j = 1$ to n

 if $(i, j) \in E$ $D[i, j] \leftarrow w(i, j);$

 else if $(i = j)$ $D[i, j] \leftarrow 0;$

 else $D[i, j] \leftarrow \infty;$

 For $k = 1$ to n

 For $i = 1$ to n

 For $j = 1$ to n

..... (contd.)

... (contd.)

if ($D[i,j] > D[i,k] + D[k,j]$)

$D[i,j] = D[i,k] + D[k,j];$

}

Lemma: At the end of k -th iteration, $D[i,j]$ is the length of shortest path using intermediate vertices $[k]$.

Pf: (Exercise)

□

Exercise: Retrieve the shortest paths.

Theorem (Floyd, Warshall, 1962): Given a graph $G = (V, E, w)$, the all-pairs shortest paths are computable in $O(|V|^3)$ time.

- Bellman-Ford iterates on the hops from the source s .

Floyd-Warshall iterates on the max-index of an intermediate vertex.