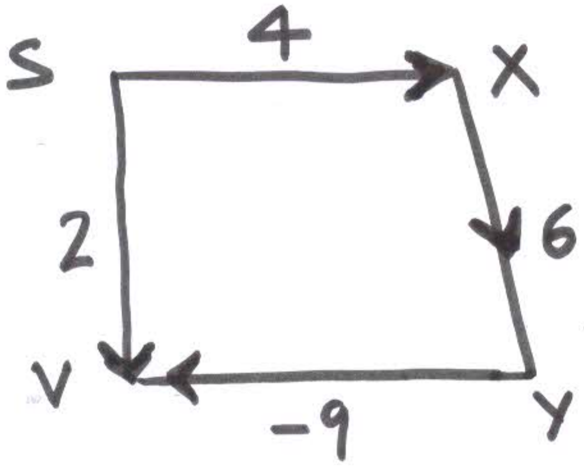
Dijkstra's Algo fails if there are -ve wts.

Ex.



The algo selects vertex v immediately after s.

But the shortest path from s to v is s — x — y — v

Bellman Ford Allows edges to have -ve wts.

Given $\vec{G} = (V, E, \omega)$ s.t

$|V| = n$, $|E| = m$, $\omega : E \rightarrow \mathbb{R}$

Find the shortest path from s to every $v \in V$.

BF$(\vec{G}, s)$

$D[s] \leftarrow 0$; $D[v] \leftarrow \infty$. if $s \neq v$

For ($i = 1$ to $n-1$)

Relax all edges simultaneously.

If $\exists (u,v) \in E$ s.t $D[v]$ can be reduced furter on relaxation of (u,v), then $\vec{G}$ contains a -ve wt cycle.

Proof of Correctness

Let $d_i(s,v)$ be the shortest distance from s to v that contains at most i edges.

<u>Claim</u>: After the i th iteration

$$D[v] = d_i(s,v) \quad \text{for each } v \in V$$

<u>Proof</u>: $d_i(s,v) = d_{i-1}(s,v)$ or

$$d_{i-1}(s,u) + \omega(u,v)$$

We can prove by induction on i

shortest distance from to every $v \in V$ is $d_{n-1}(s,v)$

$\therefore$ after $(n-1)$ iterations each $v \in V$ will have correct label. $D[v]$

Say after the nth iteration, for some v, $D[v]$ reduces further. Then we have:

$$d_n(s, v) < d_{n-1}(s, v)$$

The path that goes from s to v using n edges ~~have~~ has to make a loop, and this loop has to have a -ve weight.

<u>Complexity</u> $= O(n \cdot m)$

$= O(n^3)$

Given $\vec{G} = (V, E)$

$|V| = n \quad |E| = m \quad \omega : E \to \mathbb{R}$

We want to Compute shortest distance between every pair of vertices.

- If $\vec{G}$ has no $-ve$ wt edge, then run Dijkstra's Algo from each vertex. Cost : $O(n^3 \cdot \log n)$
- If $\vec{G}$ has no $-ve$ wt cycle then run Bellman-Ford Algo from each vertex. Cost : $O(n^4)$

- Floyd-Warshall Algo
- Based on Dynamic Programming

- Number the vertices as

  $(v_1, v_2, v_3, \ldots, v_n)$

- Let $D^k_{i,j}$ be the distance from v_i to v_j using only vertices from the set

  $\{v_1, v_2, v_3, \ldots, v_k\}$

$$D^{0}_{i,j} = \begin{cases} 0 & \text{If } i = j \\ \omega(v_i, v_j) & \text{If } (v_i, v_j) \in E \\ \infty & \text{Otherwise} \end{cases}$$

$$D^{k}_{i,j} = \text{MIN}\left[D^{k-1}_{i,j},\ D^{k-1}_{i,k} + D^{k-1}_{k,j} \right]$$

<u>Complexity</u>: $O(n^3)$

<u>Intution</u>: We are filling a $(n \times n \times n)$ matrix, and filling each cell takes constant time.