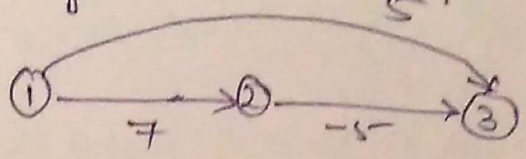
Dynamic Programming: Single Source shortest Path for General weight

→ Dijkstra Algorithm for single source shortest path Problem of directed graph $G(V,E)$ may or may not give the correct result when some or All edges of $G(V,E)$ are negative length.



In this example $|V| = n = 3$, source vertex is 1:

S	distance 1	2	3
{1}	0	7	(5)
{1,3}	0	(7)	5
{1,3,2}	0	7	5

there are direct path from source to Vertex 2 and Vertex 3.
⇒ then we select next Vertex having Minimum distance from source Vertex 1 i.e. 3.

Again in Step 2, we select a minimum distance Vertex which not belongs in S. i.e. 2

→ Dijkstra Algorithm terminated after this Step

we consider one more Step in which Vertex 2 of distance 7

$\Rightarrow dist[3] > dist[2] + c(2,3)$

$5 > 7 - 5 = 2$

$\Rightarrow dist[3] = dist[2] + c[2,3] = 2$

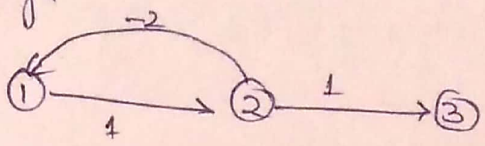
⇒ we find a vertex 3 whose distance is 2 which is minimum than the distance of path find from Dijkstra Algorithm.

⇒ Dijkstra Algo may fails for $G(V,E)$ which having Some edges or All edges of Negative weight.

⇒ we use Dynamic Programming approach to find single source shortest path Problem of General weighted graph $G(V,E)$.

→ When Negative edge length are permitted, we require that the graph have no cycles of Negative length.

→ This is necessary to assure that shortest path consisting Remain Number of edges.



In this example the length of shortest path from Vertex 1 to 3 is -∞

1, 2, 1 2, 1, 2, - - - 1, 2, 1

When there are no cycles of Negative length, there is a shortest path between any two Vertices of n vertex graph has at most n-1 edges on it.

In dijkstra Algorithm, we compute only the length dist[u].

dist[u] = length of shortest path from source vertex v to u.

Let $dist^l[u]$ = length of shortest path from source vertex v to Vertex u under the constraint that shortest path contains at most l edges.

$\Rightarrow dist^1[u] = cost[v, u] \quad 1 \le u \le n$

if there are no cycle of Negative length, we can limit our search for shortest path to path with at most n-1 edges.

$\Rightarrow dist^{n-1}[u]$ = length of constructed shortest path from v to

⇒ We compute $dist^{n-1}[u]$ by using dynamic programming approach. Consider two observation

1) if shortest path from v to u with at most k, k>1, edges has no more than k-1 edges, then

$$dist^k[u] = dist^{k-1}[u]$$

ii) if the shortest path from v to u with at most K, K>1, edges has exactly K edges, then it is made up of shortest path from v to some vertex j followed by the edge (j, u).

⇒ The path from v to j has K-1 edges, and it length is dist^{K-1}[j]

All the vertices i such that the edge <i, u> in the graph are candidates for j.

$$\Rightarrow dist^{K}[u] = \min\left\{ dist^{K-1}[u],\ \min_{i}\left\{ dist^{K-1}[i] + cost[i,u] \right\} \right\}$$

⇒ This recurrence relation can be used to compute dist^{K} from dist^{K-1} for K = 2, 3, ..., n-1.

```
Algorithm BellmanFord(v, cost, dist, n)
// Single source, all destination shortest paths with Negative edge costs
// v is the source vertex and n is the number of vertices in G(V, E)
// cost[1..n, 1..n] is the cost Adjacency matrix
// dist[j] is the length of shortest path from source vertex v to
   vertex j
{
    for i ← 1 to n do
        dist[i] = cost[v, i];

    for K ← 2 to n-1 do
        for each u such that u ≠ v and u has at least one
        incoming edge do
            for each <i, u> in the graph do
                if dist[u] > dist[i] + cost[i, u] then
                    dist[u] ← dist[i] + cost[i, u]
    }
```

time Complexity of Bellman Ford Algorithm

= $O(n^3)$ (when Adjacency matrix is used)

= $O(ne)$ when Adjacency list are used.

example:

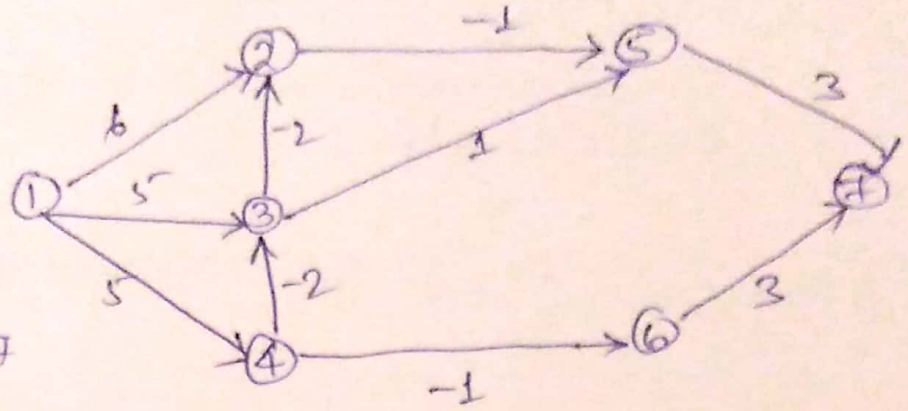


Figure: Directed graph G(V, E)

Source Vertex 1

no of Vertex = 7

$dist^k[1 \ldots 7]$

K	1	2	3	4	5	6	7
1	0	6	5	5	∞	∞	∞
2	0	3	3	5	5	4	∞
3	0	1	3	5	2	4	7
4	0	1	3	5	0	4	5
5	0	1	3	5	0	4	3
6	0	1	3	5	0	4	3

In Step 1, $dist^1[k]$, for All Vertex other than Source Vertex 1.

$\Rightarrow dist^1[1]$ = Path length from Vertex 1 to 1 = 0

$dist^1[2]$ = ① $\xrightarrow{6}$ ② = 6 = cost[1,2]

$dist^1[3]$ = cost[1,3] = ① $\xrightarrow{5}$ ③ = 5

$dist^1[4]$ = cost[1,4] = ① $\xrightarrow{5}$ ④ = 5

$dist^1[5] = dist^1[6] = dist^1[7] = \infty$ because there is no direct path from Source Vertex 1 to vertex 5, 6, 7

Step 2 :- $dist^2[2] = \min[dist^1[2], \min\{dist^1[1] + c(1,2), dist^1[3] + c[3,2]\}]$

$= \min[6, \min\{\infty + 6, 5 + (-2)\}] = \min[6, \min(6,3)]$

$= \min[6, 3] = 3$.

(because there is only two incoming edge of vertex 2 i.e $1 \to 2$, $3 \to 2$

$dist^2[3] = \min\left[dist^1[3], \min_i [dist^1[i] + cost[i,3]]\right]$

$= \min\left[5, \min\{dist^1[1] + c[1,3], dist^1[4] + c[4,3]\}\right]$

$= \min[5, \min\{0 + 5, 5 + (-2)\}]$

$= \min[5, 3] = 3$

(b

$dist^2[4] = \min\left[dist^1[4], \min\{dist^1[1] + c[1,4]\}\right]$

$= \min[5, \min\{5\}]$

$= 5$

$dist^2[5] = \min\left[dist^1[5], \min\{dist^1[2] + c[2,5], dist^1[3] + c[3,5]\}\right]$

$= \min[\infty, \min[6 + (-1), 5 + 1]]$

$= \min[\infty, 5] = 5$

$dist^2[6] = \min\{dist^1[6], \min[dist^1[4] + c[4,6]]\}$

$= \min\{\infty, \min[5 + (-1)]\} = 4$

$dist^2[7] = \min\{dist^1[7], \min\{dist^1[5] + c[5,7], dist^1[6] + c[6,7]\}\}$

$= \min[\infty, \min[\infty + 3, \infty + 3]]$

$= \infty$.

Step 3

$dist^3[2] = \min\left[dist^2[2], \min\{dist^2[i] + c(i,2)\}\right]$

$= \min\{dist^2[2], \min\{dist^2[1] + c[1,2], dist^2[3] + c[3,2]\}\}$

$= \min\{3, \min\{0+6, 3+(-2)\}\}$

$= 1$

$dist^3[3] = \min\{dist^2[3], \min\{dist^2[1] + c[1,3], dist^2[4] + c[4,3]\}\}$

$= \min\{3, \min\{0+5, 5+(2)\}\}$

$= 3$

$dist^3[4] = \min\{dist^2[4], \min\{dist^2[4] + c[1,4]\}\}$

$= 5$

$dist^3[5] = \min[dist^2[5], \min\{dist^2[2] + c[2,5], dist^2[3] + c[3,5]\}$

$= \min\{5, \min\{3+(-1), 3+1\}$

$= 2$

$dist^3[6] = \min\{dist^2[6], \min\{dist^2[4] + c[4,6]\}\}$

$= \min\{4, \min\{5-1\}\} = 4$

$dist^3[7] = \min\{dist^2[7], \min\{dist^2[5] + c[5,7], dist^2[6] + c[6,7]\}$

$= \min\{\infty, \min\{5+3, 4+3\}\} = 7$

and similarly we obtain $d^4[1 \cdots 7]$, $d^5[1 \cdots 7]$ & $d^6[1 \cdots 7]$ and the values in $d^6[1 \cdots 7]$ are the required Single Source shortest paths.