# Bellman-Ford Algorithm. (Single Source Shortest path).

- → To find the shortest path from source vertex to other vertices in a weighted graph.
- → It will work for negative edges.
- → Slower than Dijkstra's algorithm. (If there are n vertices, (n-1) iterations should be performed).

## Steps:-

1) Select a source vertex and assign the distance of source vertex is 0 and the distance of all other vertices as infinity ($\infty$)

2) Calculate the shortest distance from u to v by checking

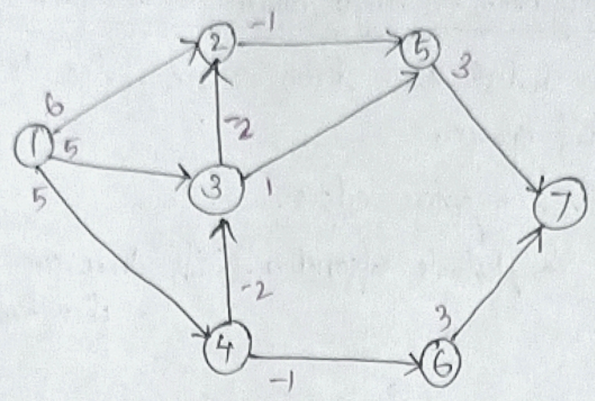
$$\text{if } (d(u) + c(u,v) < d(v)) \qquad \rightarrow \text{Relaxation.}$$
$$\text{then update } d(v) = d(u) + c(u,v)$$

3) On relaxing with all edges (n-1) times where n = no: of vertices, will we get the shortest path from source vertex to all other vertices.

## Algorithm.

BELLMAN-FORD (G, C, S)

1. Initialize_Single-Source (G, S)
2. for i=1 to |G.V| -1
3. for each edge (u,v) ∈ G.E
4. RELAX (u, v, c)
5. for each edge (u,v) ∈ G.E
6. if d(u) + c(u,v) > d(v)
   return false.
7. return True.

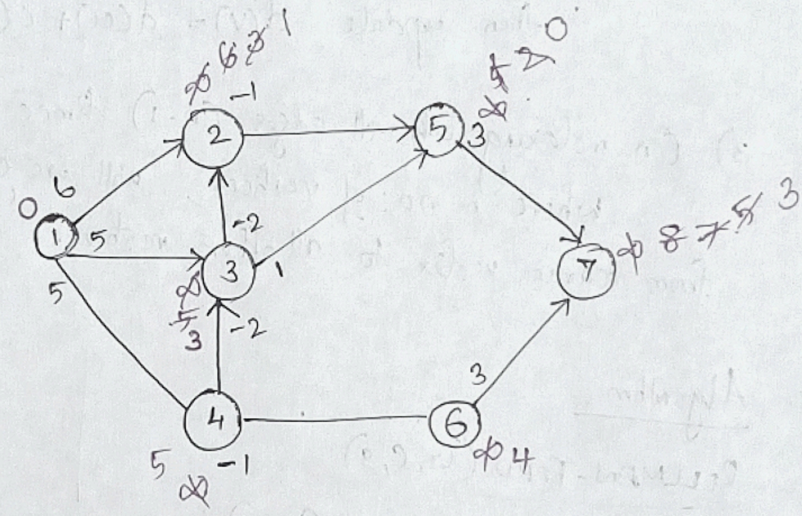



Here there are 7 vertices. ∴ 6 iterations are required.

First write all the pair of vertices.

(1,2), (1,3), (1,4), (2,5), (3,2), (3,5), (4,3), (4,6), (5,7), (6,7)

Now, relax all these edges.




Mark the distance of the starting vertex as 0 and the remaining as ∞.

Iteration 1

(1,2)

$0+6 < \infty$

$\therefore d(2) = 6$.

(1,3)

$0+5 < \infty$

$\therefore d(3) = 5$

(1,4)

$0+5 < \infty$

$\therefore d(4) = 5$

(2,5)

$6-1 < \infty$

$5 < \infty$

$\therefore d(5) = 5$.

(3,2)

$5-2 < 6$

$3 < 6$.

$d(2) = 3$.

(3,5)

$5+1 > 5$.

$6 > 5$

$\therefore$ Don't modify d(5).

(4,3)

$5-2 < 5$

$3 < 5$

$d(3) = 3$.

(5,7)

$5+3 < \infty$

$8 < \infty$

$\therefore d(7) = 8$

(6,7)

$4+3 < 8$

$7 < 8$

$\therefore d(7) = 7$

(4,6)

$5-1 < \infty$

$4 < \infty$

$\therefore d(6) = 4$

## Iteration 2.

**(1,2)**

$0+6>3$.

No change.

**(1,3)**

$0+5>3$

No change

**(1,4)**

$0+5=5$

No change.

**(2,5)**

$3-1<5$

$2<5$

$d(5)=2$

**(3,2)**

$3-2<3$

$1<3$

$\therefore d(2)=1$

**(3,5)**

$3+1>2$

No change.

**(4,3)**

$5-2=3$.

No change

**(4,6)**

$5-1=4$

No change

**(5,7)**.

$2+3<7$

$5<7$

$d(7)=5$

**(6,7)**

$4+3>5$

No change.

## Iteration 3.

**(1,2)**.

No change

**(1,3)**

No change

**(1,4)**

No change.

**(2,5)**

$1-1<2$

$0<2$

$\therefore d(2)=0$

**(3,2)**

No change

**(3,5)**

No change.

**(4,5)**

No change.

**(4,6)**

No change

**(5,7)**

$0+3<5$

$3<5$.

$d(7)=3$.

**(6,7)**

No change.

In the fourth iteration onwards it stops changing. So stop iteration.
[But the program will continue ~~at~~ till the remaining 3 more iterations are completed.]

Result:

$d(1) = 0$

$d(2) = 1$

$d(3) = 3$

$d(4) = 5$

$d(5) = 0$

$d(6) = 4$

$d(7) = 3$.

Time Complexity:

$\rightarrow O(E \cdot (|V|-1))$

$= O(E \cdot V)$

→ In case of complete graph, time complexity:

In complete graph, no: of edges $(E) = \frac{n(n-1)}{2}$

(where $n$ is the no: of vertices)

So, $O(E \cdot V)$ becomes → $\left[\frac{n(n-1)}{2} (n-1)\right]$

$= O(n^3)$

Drawback of Bellman-Ford algorithm.

This algorithm will not work with graphs having cycles. If there a negative cycle in the graph, the value continue changing even after $(n-1)$ iterations.

Eg:-