**Bellman-Ford Algorithm**

Solves single shortest path problem in which edge weight may be negative but no negative cycle exists.

This algorithm works correctly when some of the edges of the directed graph G may have negative weight. When there are no cycles of negative weight, then we can find out the shortest path between source and destination.

It is slower than Dijkstra's Algorithm but more versatile, as it capable of handling some of the negative weight edges.

This algorithm detects the negative cycle in a graph and reports their existence.

Based on the "**Principle of Relaxation**" in which more accurate values gradually recovered an approximation to the proper distance by until eventually reaching the optimum solution.

Given a weighted directed graph G = (V, E) with source s and weight function w: E → R, the Bellman-Ford algorithm returns a Boolean value indicating whether or not there is a negative weight cycle that is attainable from the source. If there is such a cycle, the algorithm produces the shortest paths and their weights. The algorithm returns TRUE if and only if a graph contains no negative - weight cycles that are reachable from the source.

**Recurrence Relation**

**distk [u] = [min[distk-1 [u],min[ distk-1 [i]+cost [i,u]]] as i except u.**

k → k is the source vertex  
u → u is the destination vertex  
i → no of edges to be scanned concerning a vertex.

**BELLMAN -FORD (G, w, s)**

1. INITIALIZE - SINGLE - SOURCE (G, s)

2. for i ← 1 to |V[G]| - 1

3. do for each edge (u, v) ∈ E [G]

4. do RELAX (u, v, w)

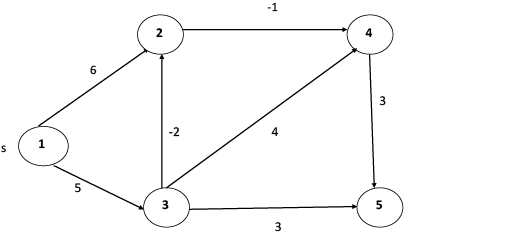
5. for each edge (u, v) ∈ E [G]

6. do if d [v] > d [u] + w (u, v)

7. then return FALSE.

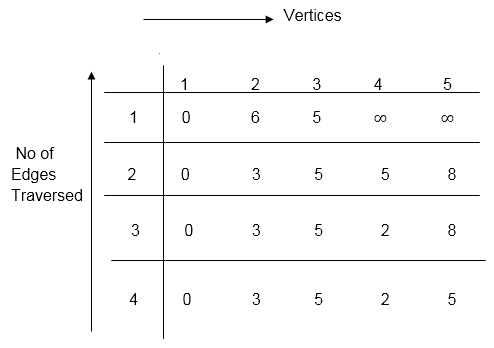
8. return TRUE.

**Example:** Here first we list all the edges and their weights.



**Solution:**

**distk [u] = [min[distk-1 [u],min[distk-1 [i]+cost [i,u]]] as i ≠ u.**



dist2 [2]=min[dist1 [2],min[dist1 [1]+cost[1,2],dist1 [3]+cost[3,2],dist1 [4]+cost[4,2],dist1 [5]+cost[5,2]]]

Min = [6, 0 + 6, 5 + (-2), ∞ + ∞ , ∞ +∞] = 3

dist2 [3]=min[dist1 [3],min[dist1 [1]+cost[1,3],dist1 [2]+cost[2,3],dist1 [4]+cost[4,3],dist1 [5]+cost[5,3]]]

Min = [5, 0 +∞, 6 +∞, ∞ + ∞ , ∞ + ∞] = 5

dist2 [4]=min[dist1 [4],min[dist1 [1]+cost[1,4],dist1 [2]+cost[2,4],dist1 [3]+cost[3,4],dist1 [5]+cost[5,4]]]

Min = [∞, 0 +∞, 6 + (-1), 5 + 4, ∞ +∞] = 5

dist2 [5]=min[dist1 [5],min[dist1 [1]+cost[1,5],dist1 [2]+cost[2,5],dist1 [3]+cost[3,5],dist1 [4]+cost[4,5]]]

Min = [∞, 0 + ∞,6 + ∞,5 + 3, ∞ + 3] = 8

dist3 [2]=min[dist2 [2],min[dist2 [1]+cost[1,2],dist2 [3]+cost[3,2],dist2 [4]+cost[4,2],dist2 [5]+cost[5,2]]]

Min = [3, 0 + 6, 5 + (-2), 5 + ∞ , 8 + ∞ ] = 3

dist3 [3]=min[dist2 [3],min[dist2 [1]+cost[1,3],dist2 [2]+cost[2,3],dist2 [4]+cost[4,3],dist2 [5]+cost[5,3]]]

Min = [5, 0 + ∞, 3 + ∞, 5 + ∞,8 + ∞ ] = 5

dist3 [4]=min[dist2 [4],min[dist2 [1]+cost[1,4],dist2 [2]+cost[2,4],dist2 [3]+cost[3,4],dist2 [5]+cost[5,4]]]

Min = [5, 0 + ∞, 3 + (-1), 5 + 4, 8 + ∞ ] = 2

dist3 [5]=min[dist2 [5],min[dist2 [1]+cost[1,5],dist2 [2]+cost[2,5],dist2 [3]+cost[3,5],dist2 [4]+cost[4,5]]]

Min = [8, 0 + ∞, 3 + ∞, 5 + 3, 5 + 3] = 8

dist4 [2]=min[dist3 [2],min[dist3 [1]+cost[1,2],dist3 [3]+cost[3,2],dist3 [4]+cost[4,2],dist3 [5]+cost[5,2]]]

Min = [3, 0 + 6, 5 + (-2), 2 + ∞, 8 + ∞ ] =3

dist4 [3]=min[dist3 [3],min[dist3 [1]+cost[1,3],dist3 [2]+cost[2,3],dist3 [4]+cost[4,3],dist3 [5]+cost[5,3]]]

Min = 5, 0 + ∞, 3 + ∞, 2 + ∞, 8 + ∞ ] =5

dist4 [4]=min[dist3 [4],min[dist3 [1]+cost[1,4],dist3 [2]+cost[2,4],dist3 [3]+cost[3,4],dist3 [5]+cost[5,4]]]

Min = [2, 0 + ∞, 3 + (-1), 5 + 4, 8 + ∞ ] = 2

dist4 [5]=min[dist3 [5],min[dist3 [1]+cost[1,5],dist3 [2]+cost[2,5],dist3 [3]+cost[3,5],dist3 [5]+cost[4,5]]]

Min = [8, 0 +∞, 3 + ∞, 8, 5] = 5

**RELEVANT READING MATERIAL AND REFERENCES:**

**Source Notes:**

1. https://www.javatpoint.com/bellman-ford-algorithm

**Lecture Video:**

1. <https://youtu.be/FtN3BYH2Zes>

**Online Notes:**

1. <http://vssut.ac.in/lecture_notes/lecture1428551222.pdf>

**Text Book Reading:**

1. Cormen, Leiserson, Rivest, Stein, “*Introduction to Algorithms*”, Prentice Hall of India, 3rd edition 2012. problem, Graph coloring.

**In addition: PPT can be also be given.**