SINGLE SOURCE SHORTEST PATHS

NIKITA UIKEY
181210033

WHAT WE'LL TALK ABOUT

Outline of Contents

- Shortest-Path Variants
- Negative Weight Edges
- Dijkstra's Algorithm
- BELLMAN-FORD ALGORITHM

SHORTEST-PATH VARIANTS

There are some variants of the shortest path problem.

- SINGLE- DESTINATION SHORTEST - PATHS PROBLEM:

FIND THE SHORTEST PATH TO A GIVEN DESTINATION VERTEX T FROM EVERY VERTEX V.

- SINGLE - PAIR SHORTEST - PATH PROBLEM:

FIND THE SHORTEST PATH FROM U TO V FOR GIVEN VERTICES U AND V.

- ALL - PAIRS SHORTEST - PATHS PROBLEM:

FIND THE SHORTEST PATH FROM U TO V FOR EVERY PAIR OF VERTICES U AND V.

SINGLE SOURCE SHORTEST PATH

- To find the shortest path from a source vertex v to all other vertices in the graph.
- For a connected ,weighted, directed graph G(V,E), associated with each edge ⟨u,v⟩ ∈ E, there is a weight ω(u,v).
- The weight ω(p) of path p=⟨ v0 , v1 ,..., vk ⟩ is the sum of the weights of its constituent

edges:

$$\omega(p)=\sum_{i=1}^{k} \omega(v_{i-1}, v_i).$$

$$\delta(u,v)=\left\{\begin{array}{ll} \min\{\omega(p) : u\ v\}; & \text{if there is a path from } u \text{ to } v, \\ \infty & \text{, otherwise.} \end{array}\right\}$$

SHORTEST PATH= PATH OF MINIMUM WEIGHT

NEGATIVE WEIGHT EDGES

IT IS A WEIGHTED GRAPH IN WHICH THE TOTAL WEIGHT OF AN EDGE IS NEGATIVE.

- If a graph has a negative edge, then it produces a chain. After executing the chain if the output is negative then it will give - ∞ weight and condition get discarded.
- If weight is less than negative and - ∞ then we can't have the shortest path in it.

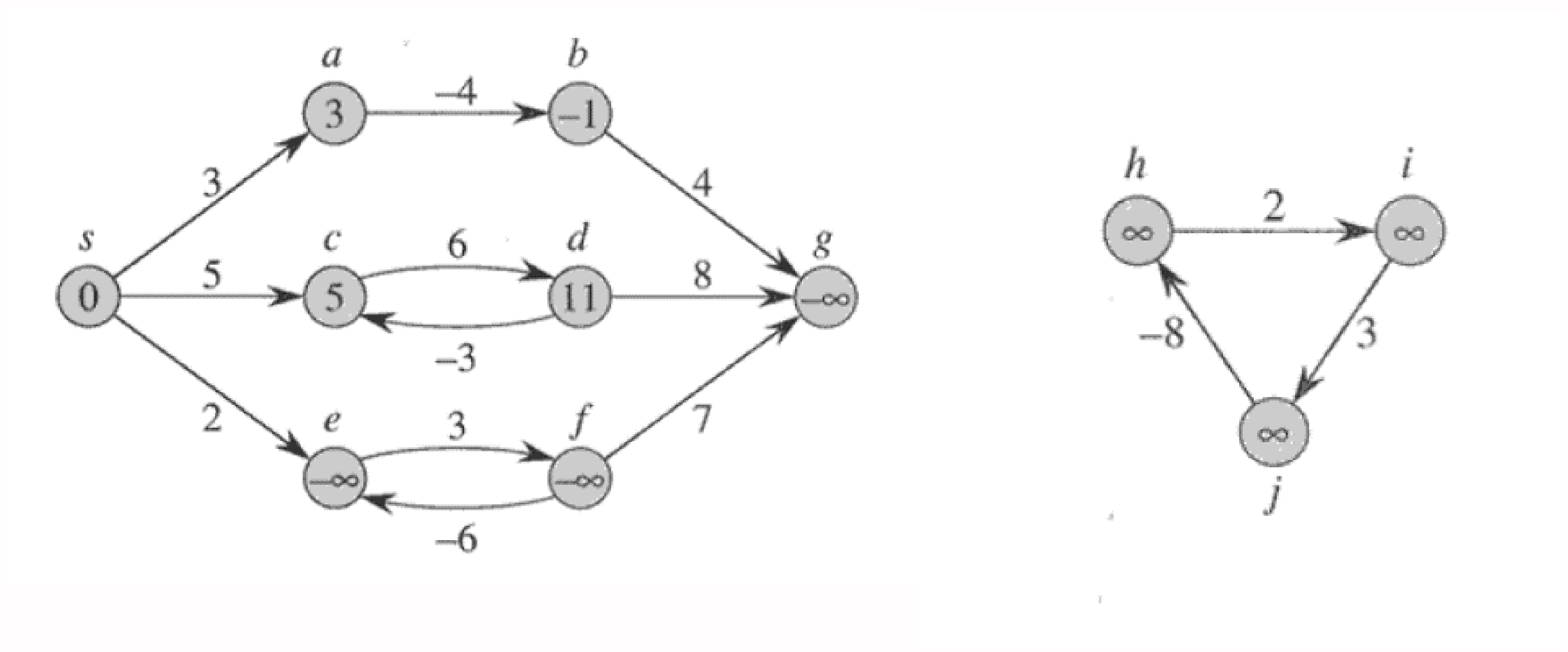
Briefly, if the output is -ve,

then both condition get discarded.

1. - ∞
2. Not less than 0.

And we cannot have the shortest Path.

Negative edge weight in a directed graph



Beginning from s Adj [s] = [a, c, e]

Weight from s to a is 3

SUPPOSE WE WANT TO CALCULATE A PATH FROM S→C. SO WE HAVE 2 PATHS /WEIGHT

s to c = 5, s→c→d→c=8

But s→c is minimum

So s→c = 5

SUPPOSE WE WANT TO CALCULATE A PATH FROM S→E. SO WE HAVE TWO PATHS AGAIN

s→e = 2,

s→e→f→e=-1 As -1 < 0 ∴ Conditio

n gets discarded.

Dijkstra's Algorithm

Dijkstra's Algorithm (G, w, s)

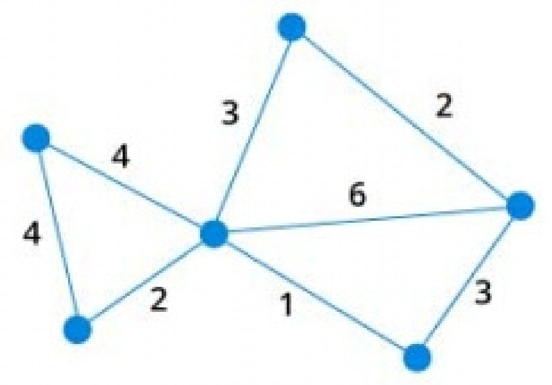
```
1. INITIALIZE - SINGLE - SOURCE (G, s)
2. S$\leftarrow\varnothing$
3. Q$\leftarrow$V [G]
4. while Q $\neq$ $\varnothing$
5. do u $\leftarrow$ EXTRACT - MIN (Q)
6. S $\leftarrow$ S U {u}
7. for each vertex v $\in$ Adj [u]
8. do RELAX (u, v, w)
```

- It is a greedy algorithm that solves the single-source shortest path problem for a directed graph G = (V, E) with nonnegative edge weights, i.e., w (u, v) $\geqslant$ 0 for each edge (u, v) $\in$ E.
- Time Complexity of Dijkstra's Algorithm is O(V2) but with min-priority queue it drops down to O(V+ E.logV).

EXAMPLE

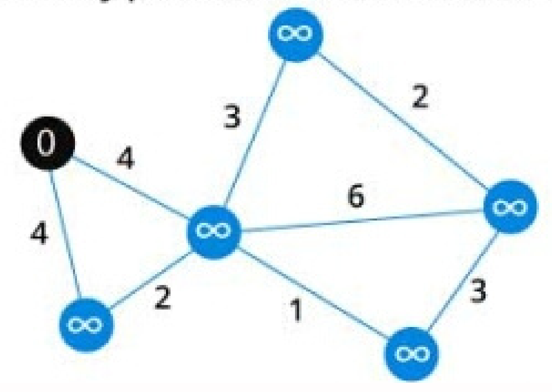
1

Start with a weighted graph



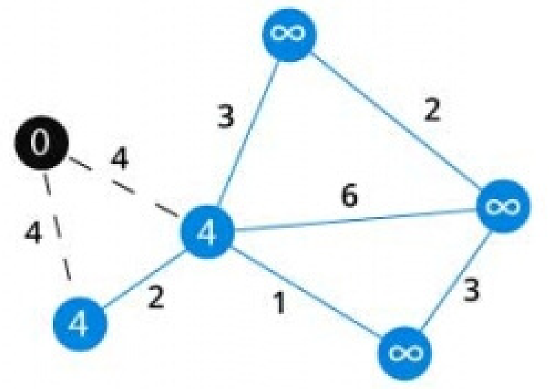
2

Choose a starting vertex and assign infinity path values to all other vertices



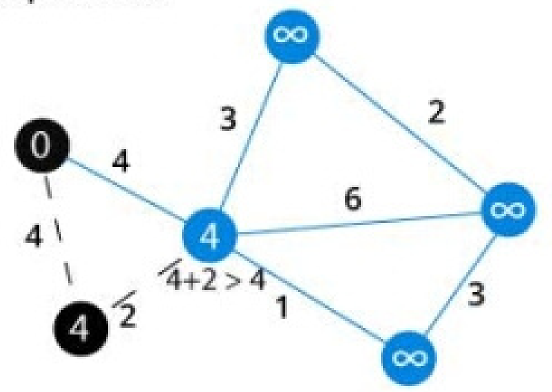
3

Go to each vertex adjacent to this vertex and update its path length



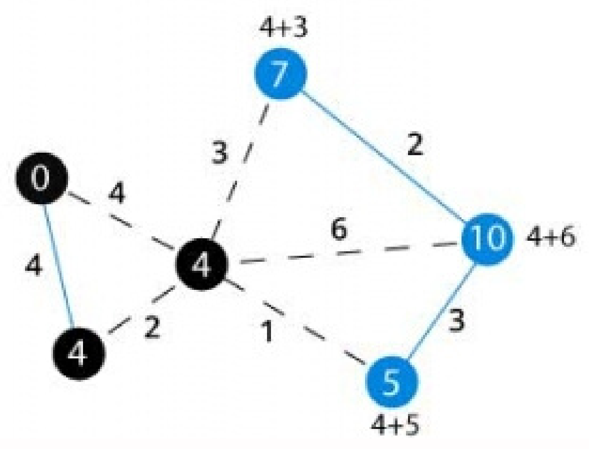
4

If the path length of adjacent vertex is lesser than new path length, don't update it.



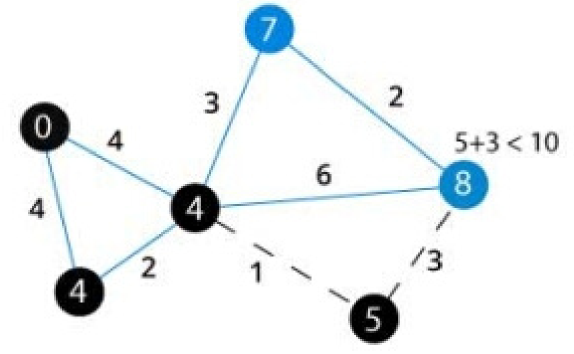
5

Avoid updating path lengths of already visited vertices



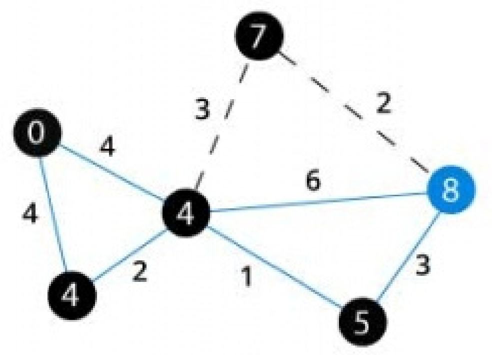
6

After each iteration, we pick the unvisited vertex with least path length. So we chose 5 before 7



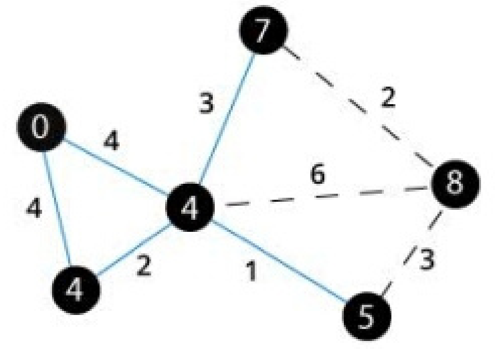
7

Notice how the rightmost vertex has its path length updated twice



8

Repeat until all the vertices have been visited



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 similar to Dijkstra's algorithm but it can work with graphs in which edges can have negative weights..

Why would one ever have edges with negative weights in real life?

- Negative weight edges might seem useless at first but they can explain a lot of phenomena like cashflow, heat released/absorbed in a chemical reaction etc.
- For instance, if there are different ways to reach from one chemical A to another chemical B, each method will have sub-reactions involving both heat dissipation and absorption.
- If we want to find the set of reactions where minimum energy is required, then we will need to be able to factor in the heat absorption as negative weights and heat dissipation as positive weights.

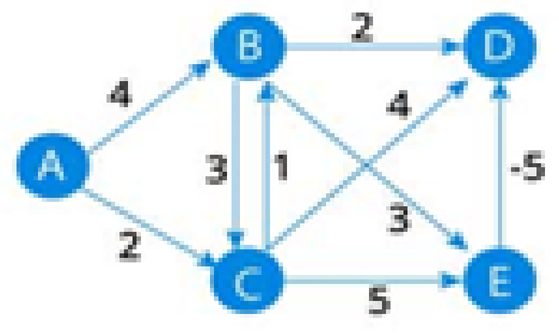
PSEUDOCODE

```
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

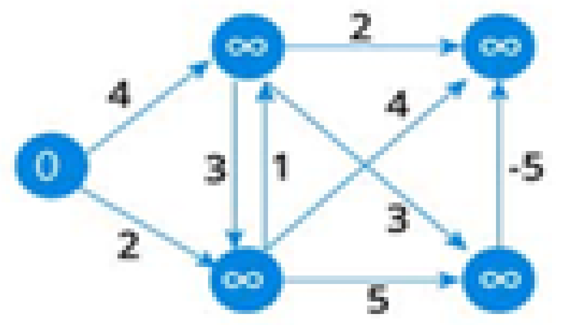
1

Start with a weighted graph



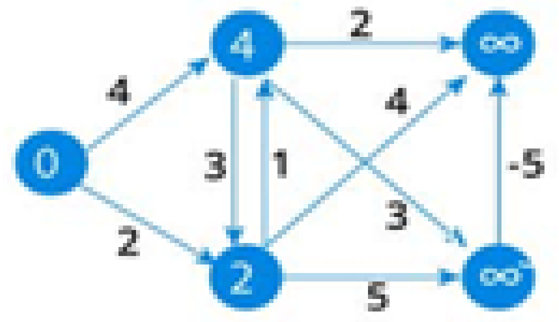
2

Choose a starting vertex and assign infinity path values to all other vertices



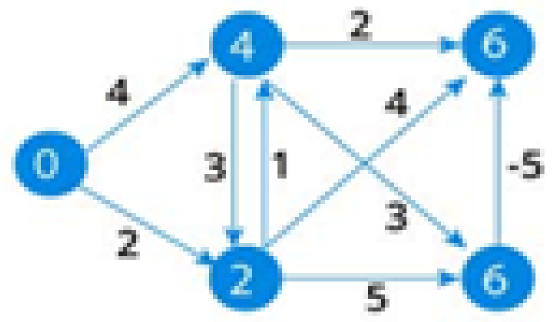
3

Visit each edge and relax the path distances if they are inaccurate



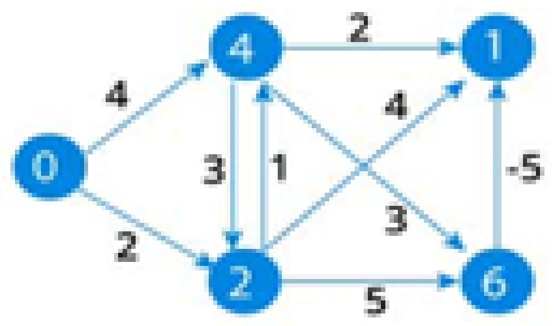
4

We need to do this V times because in the worst case, a vertex's path length might need to be readjusted V times



5

Notice how the vertex at the top right corner had its path length adjusted



6

After all the vertices have their path lengths, we check if a negative cycle is present.

A	B	C	D	E
0	∞	∞	∞	∞
0	4	2	∞	∞
0	3	2	6	6
0	3	2	1	6
0	3	2	1	6

TIME COMPLEXITY

- First nested for-loop performs |V|-1 relaxation passes; relax every edge at each pass.
- Last for-loop checks the existence of a negative-weight cycle reachable from s.
- Time Complexity of Bellman Ford algorithm is O(V·E)