Single Source Shortest Path Problem

Given a graph G=(V,E), a weight function w: E -> R, and a source node s, find the shortest path from s to v for every v in V.

Drawbacks of Dijkstra's algorithm:

- Dijkstra's algorithm may or may not work (give correct answer) if the graph contain negative weight edge.
- Dijkstra's algorithm will not work when graph contain negative weight edge cycle. Graph G contain cycles of edges of negative total weight.

To resolve the above problem we have algorithm -**Bellman-Ford SSSP Algorithm**

- We allow negative edge weights.
- One can detect if there is a negative weight cycles in the given graph.

Bellman-Ford SSSP Algorithm

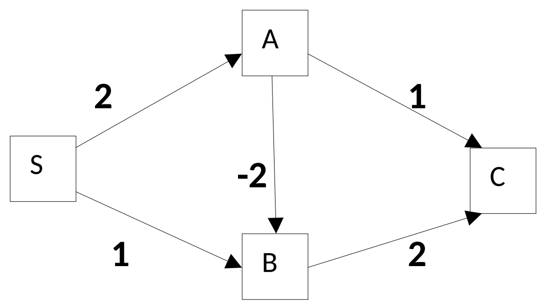
```
Input: directed or undirected graph G = (V,E,w)
   for all v in V {
         d[v] = infinity; parent[v] = nil;      }------- O(V)
   }
  d[s] = 0; parent[s] = s;
  for i := 1 to |V| - 1 {      // ensure that  information on distance from s propagates
      for each (u,v) in E  {     // relax all edges
          if (d[u] + w(u,v) < d[v]) then { d[v] := d[u] + w(u,v); parent[v] := u; }      }O(VE)
      }
  }
```

Analysis of algorithm

Running time: O(V) + O(VE)

Overall running time complexity of Bellman-Ford SSSP Algorithm is O(VE)

Example Bellman-Ford



Iterations

Node	initial	1	2	3
S(source)	0	0	0	0
A	∞	2	-1	-1
B	∞	1	1	1
C	∞	∞	3	0

For this algorithm we required V-1(in above e.g. we require 3 iteration) iterations to solve the problem and verification of result we perform one more iteration to check whether the result of last iteration (V-1th iteration) changes or not if it change then their exist a negative weight cycle and if doesn't change then our result is correct.

For verification of above example do one more iteration (do the procedure one ore time).

Iterations

Node	initial	1	2	3	4(verification)
S(source)	0	0	0	0	0
A	∞	2	-1	-1	-1
B	∞	1	1	1	1
C	∞	∞	3	0	0

In the verification iteration (4th) result is same as 3rd, so result is correct and negative weight cycle not exist.

Correctness

Fact 1: The distance estimate d[v] never underestimates the actual shortest path distance from s to v.

Fact 2: If there is a shortest path from s to v containing at most I edges, then after iteration i of the outer for loop: d[v] <= the actual shortest path distance from s to v.

Theorem: Suppose that G is a weighted graph without negative weight cycles and let s denote the source node. Then Bellman-Ford correctly calculates the shortest path distances from s.

Proof: Every shortest path has at most |V| - 1 edges. By Fact 1 and 2, the distance estimate d[v] is equal to the shortest path length after |V|-1 iterations.

Variations

- One can stop the algorithm if an iteration does not modify distance estimates. This is beneficial if shortest paths are likely to be less than |V|-1.
- One can detect negative weight cycles by checking whether distance estimates can be reduced after |V|-1 iterations.