Data Structures and Algorithms

Prof. Ganesh Ramakrishnan,
Prof. Ajit Diwan,
Prof. D.B. Phatak

Department of Computer Science and Engineering
IIT Bombay

Session: Shortest Path Algorithm
(Bellman-Ford Algorithm)

Bellman-Ford Algorithm

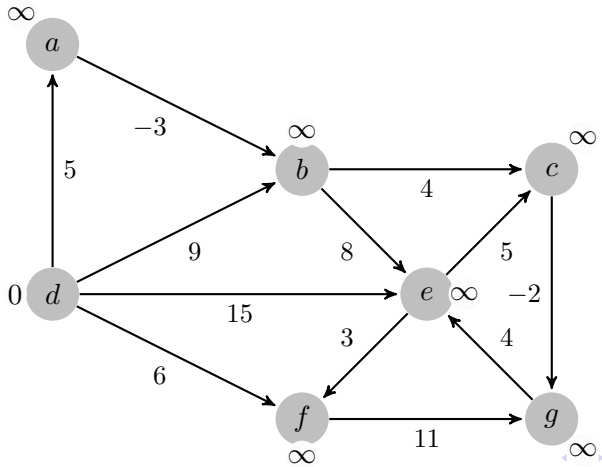
1. Computes shortest paths from a source vertex to all other vertices in a weighted directed graph.
2. Dijkstra's Algorithm does not work for negative edges.
3. Solves the problem of negative edge weights but is slower than Dijkstra's Algorithm.
4. Running time: O(VE)

Bellman-Ford Shortest Path Algorithm

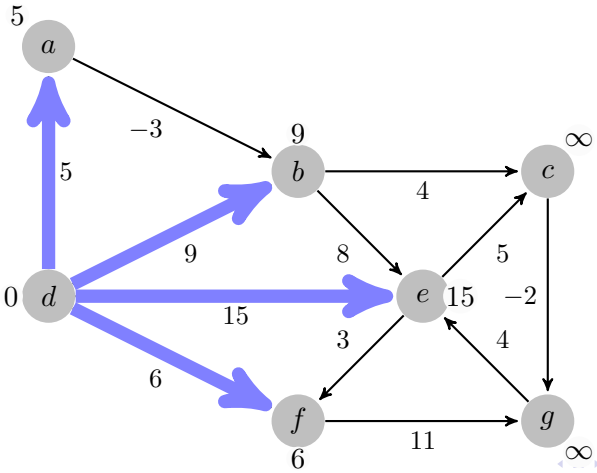
Algorithm ComputeBellman-FordSPs(G, s, w)
Output: Returns false for negative-weight cycle, else produces shortest paths with predecessor.

```
for v ∈ G.getVertices() do
    if v = s then
        d[v] = 0
        predecessor[v] = NULL
    else
        d[v] = ∞
    end if
end for
for i ∈ {1, ..., G.getVertices() − 1} do
    for e ∈ G.edges() do
        if d[u] + w < d[v] then
            d[v] = d[u] + w
            predecessor[v] = u
        end if
    end for
end for
for e ∈ G.edges() do
    if d[u] + w < d[v] then
        return FALSE
    end if
end for
return d[], predecessor[]
```

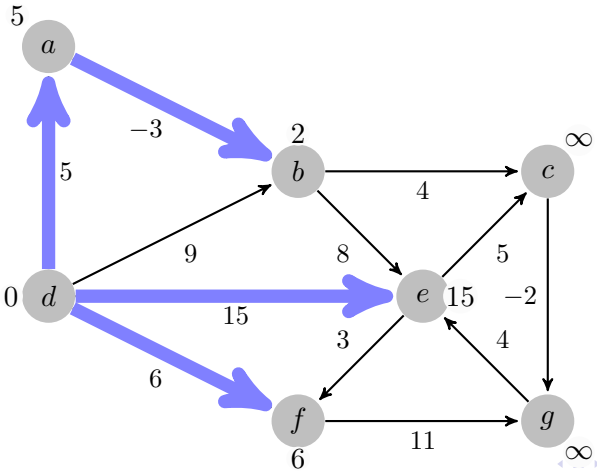
Bellman-Ford



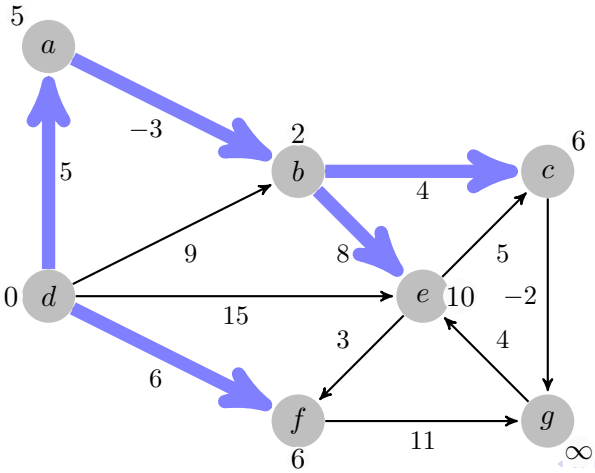
Bellman-Ford



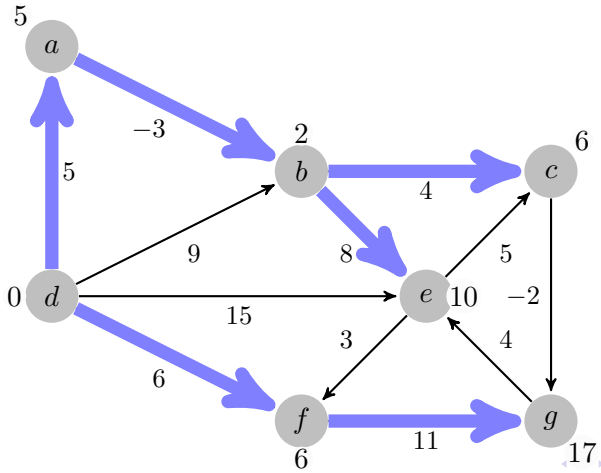
Bellman-Ford



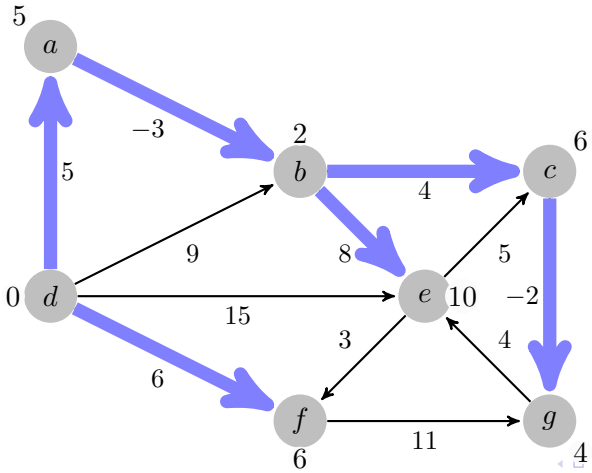
Bellman-Ford



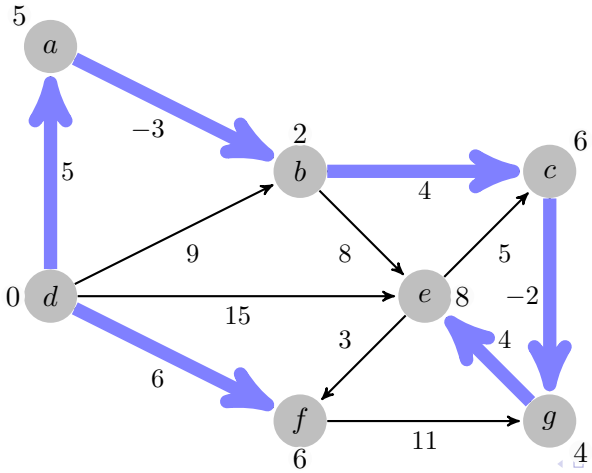
Bellman-Ford



Bellman-Ford



Bellman-Ford



Analysis of Bellman-Ford Shortest Path Algorithm

Algorithm ComputeBellman-FordSPs(G, s, w)
Output: Returns false for negative-weight cycle, else produces shortest paths with predecessor.

for $v \in G.getVertices()$ **do**
  if $v = s$ **then**
    $d[v] = 0$
    $predecessor[v] = NULL$
  else
    $d[v] = \infty$
  end if
end for $\implies$ c_1 $\times |V|$ times
for $i \in \{1, ..., G.getVertices() - 1\}$ **do**
  for $e \in G.edges()$ **do**
    if $d[u] + w < d[v]$ **then**
      $d[v] = d[u] + w$
      $predecessor[v] = u$
    end if $\implies$ c_2 $\times |1|$ times
  end for $\implies$ c_3 $\times |E|$ times
end for $\implies$ c_4 $\times |V|$ times
for $e \in G.edges()$ **do**
  if $d[u] + w < d[v]$ **then**
    return $FALSE$
  end if
end for $\implies$ c_5 $\times |E|$ times
return $d[]$, $predecessor[]$

$$T(n) = c_1|V| + c_2c_3c_4|V||E| + c_5|E| = O(|V||E|)$$

Thank you