The Bellman–Ford Algorithm

UNLV: Analysis of Algorithms Lawrence L. Larmore

The Single Source Minimum Path Problem

We are given a weighted directed graph $G = (V, E, W)$ with a designated *source vertex* s. That is, if $e = (u, v) \in E$, then $W(e) = W(u, v)$ is the *weight* of the edge e. A solution to the problem consists of arrays $\{V[v]\}_{v \in V}$ and $\{back[v]\}_{v \in V}$, such that $V[v]$ is the minimum weight of any path from s to v, while $back[v]$ is the next-to-the-last vertex of one of those minimum paths. There is no solution if G has a negative cycle.

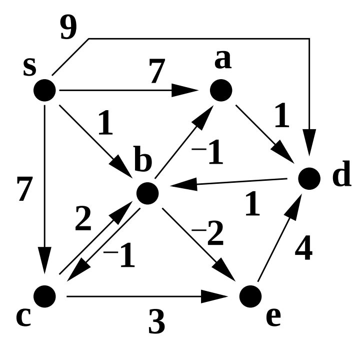
```
for all v in V
   back[v] := *
   V[v] := infinity
endfor
V[s] := 0
altered := true
while (altered)
   altered := false
   for all e = (u,v) in E
      if (V[u] + W(e) < V[v])
         V[v] := V[u] + W(e)
         back[v] := u
         altered := true
      endif
   endfor
endwhile
```

The running time of the Bellman-Ford algorithm is $O(nm)$. If ℓ is the length of the longest minimum weight path found, The above code runs in only $O(\ell m)$ time. If G has a negative cycle, the above code will never halt.

The code below contains protection against this. If the while loop executes n times and some value of V is altered at the n^{th} iteration, there must be a negative cycle.

```
for all v in V
   back[v] := *
   V[v] := infinity
endfor
V[s] := 0
altered := true
numiterations := 0
while (altered and (numiterations < n))
   altered := false
   numiterations := numiterations + 1
   for all e = (u,v) in E
      if (V[u] + W(e) < V[v])
         V[v] := V[u] + W(e)
         back[v] := u
         altered := true
      endif
   endfor
endwhile
if (altered)
   Write('There is a negative cycle.')
endif
```

Output of the Bellman Ford Algorithm



	s	a	b	c	d	e
V	0	7	1	7	9	∞
back	s	s	s	s	$\perp$	$\perp$

	s	a	b	c	d	e
V	0	0	1	0	8	-1
back	$\perp$	b	s	b	a	b

The first array shows the variables of the Bellman-Ford algorithm after one iteration, showing values of only paths of length 1 are considered, while the second array shows the values if only paths of length at most 2 are considered. After one execution of the main loop of the algorithm, the values of V will be no greater than those shown, but possibly smaller, depending on the order of the visitation of the edges.

	s	a	b	c	d	e
V	0	0	1	0	1	-1
back	$\perp$	b	s	s	a	b

The last array shows the variables of the Bellman-Ford algorithm if only paths of length at most 4 are considered. These will be the values after three iterations of the main loop of the algorithm, since no smallest path has length greater than 3.