Bellman-Ford Algorithm

The Bellman-Ford algorithm solves the single-source shortest-paths problem in the general case in which edge weights may be negative.

\[
\text{BELLMAN-FORD}(G, w, s)
\]

1. \text{INITIALIZE-SINGLE-SOURCE}(G, s)
2. \text{for} \ i = 1 \ \text{to} \ |G. V| - 1
3. \text{for each edge} \ (u, v) \in G. E
4. \text{RELAX}(u, v, w)
5. \text{for each edge} \ (u, v) \in G. E
6. \text{if} \ v. d > u. d + w(u, v)
7. \text{return} \ FALSE \text{ -- a negative cycle was detected}
8. \text{return} \ TRUE \text{ -- no negative cycles}

\[\Theta(V) \text{ call} \quad \# \text{ of time 5} \]
\[\Theta(V) \quad \# \text{ of time 5} \]
\[(V-1)(E+1) \quad \text{for each edge} \ (u, v) \in G. E \quad \Theta(1) \text{ call} \]
\[(V-1) \quad \text{for each edge} \ (u, v) \in G. E \quad 1 \text{ to } E+1 \]
\[1 \text{ to } E+1 \quad \text{for each edge} \ (u, v) \in G. E \quad 1 \text{ to } E \]
\[1 \text{ to } E \quad \text{for each edge} \ (u, v) \in G. E \quad 0 \text{ to } 1 \]
\[0 \text{ to } 1 \quad \text{for each edge} \ (u, v) \in G. E \quad 0 \text{ to } 1 \]
\[\Theta(VE) \quad \text{for each edge} \ (u, v) \in G. E \quad \text{for each edge} \ (u, v) \in G. E \]

The Bellman-Ford algorithm runs in time \(O(VE)\), since the initialization in line 1 takes \(\Theta(V)\) time, each of the \(|V| - 1\) passes over the edges in lines 2–4 takes \(\Theta(E)\) time, and the \text{for} loop of lines 5–7 takes \(O(E)\) time.
The **Bellman-Ford algorithm** solves the single-source shortest-paths problem in the general case in which edge weights may be negative.

**Bellman-Ford** \((G, w, s)\)

1. **Initialize-Single-Source** \((G, s)\)
2. for \(i = 1\) to \(|G.V| - 1\) do
3. for each edge \((u, v) \in G.E\)
4. \[\text{RELAX}(u, v, w)\]  
5. for each edge \((u, v) \in G.E\)
6. if \(v.d > u.d + w(u, v)\)
7. \[\text{return} \ FALSE\]  
8. \[\text{return} \ TRUE\]

The Bellman-Ford algorithm runs in time \(O(VE)\), since the initialization in line 1 takes \(\Theta(V)\) time, each of the \(|V| - 1\) passes over the edges in lines 2–4 takes \(\Theta(E)\) time, and the for loop of lines 5–7 takes \(O(E)\) time.
Examples

Definition
Subpaths

Examples
Basic Methods
Bellman-Ford 1
Bellman-Ford 2
Bellman-Ford 3
Dijkstra 1
Dijkstra 2
Dijkstra 3

CS 3343 Analysis of Algorithms

Single-Source Shortest Paths – 4
Theorem 24.4 (Correctness of the Bellman-Ford algorithm)
Let BELLMAN-FORD be run on a weighted, directed graph $G = (V, E)$ with source $s$ and weight function $w : E \rightarrow \mathbb{R}$. If $G$ contains no negative-weight cycles that are reachable from $s$, then the algorithm returns TRUE, we have $v.d = \delta(s, v)$ for all vertices $v \in V$, and the predecessor subgraph $G_\pi$ is a shortest-paths tree rooted at $s$. If $G$ does contain a negative-weight cycle reachable from $s$, then the algorithm returns FALSE.

lines 2-4 relax all paths $V-1$ or less

lines 5-7 relax only if a negative cycle
Dijkstra's algorithm solves the single-source shortest-paths problem on a weighted, directed graph $G = (V, E)$ for the case in which all edge weights are nonnegative. 

**DIJKSTRA**($G, w, s$)

1. **INITIALIZE-SINGLE-SOURCE**($G, s$)
2. $S = \emptyset$
3. $Q = G.V$ add all vertices to $Q$
4. **while** $Q \neq \emptyset$
5. $u = \text{EXTRACT-MIN}(Q)$
6. $S = S \cup \{u\}$
7. **for** each vertex $v \in G.Adj[u]$
8. \text{RELAX}(u, v, w)

**Theorem 24.6 (Correctness of Dijkstra's algorithm)**

Dijkstra's algorithm, run on a weighted, directed graph $G = (V, E)$ with non-negative weight function $w$ and source $s$, terminates with $u.d = \delta(s, u)$ for all vertices $u \in V$. 
There are 3 edges from $S$.

Know shortest path to $b$ is 8 because any other path has to go thru $a$ or $c$ and so must be longer.

Know shortest path to $a$ is 13 because any other path has to go over a subpath $\geq (3)$.