Dijkstra's Algorithm Definitions

- s = source node
- N = set of nodes so far incorporated by the algorithm
- c(i, j) = link cost from node i to node j
  - c(i, i) = 0
  - c(i, j) = ∞ if the two nodes are not directly connected
  - c(i, j) ≥ 0 if the two nodes are directly connected
- D(v) = cost of least-cost path from node s to node v currently known (at termination, D(v) will be the least-cost path from s to v)

Dijkstra's Algorithm

- Step 1 [Initialization]
  - N = {s} Set of nodes so far incorporated consists of only source node
  - $D(v) = c(s, v)$ for $n \neq s$
  - Initial path costs to neighboring nodes are simply link costs

- Step 2 [Get Next Node]
  - Find neighboring node not in N with least-cost path from s
  - Incorporate node into N

- Step 3 [Update Least-Cost Paths]
  - $D(v) = \min[D(v), D(w) + c(w, v)]$ for all $w \notin N$

- Return to Step 2. Algorithm terminates when all nodes have been added to T

Bellman-Ford Algorithm Definitions

- s = source node
- c(i, j) = link cost from node i to node j
  - c(i, i) = 0
  - c(i, j) = ∞ if the two nodes are not directly connected
  - c(i, j) ≥ 0 if the two nodes are directly connected
- h = maximum number of links in path at current stage of the algorithm
- $D_h(v)$ = cost of least-cost path from s to v under constraint of no more than h links

Bellman-Ford Algorithm

- Step 1 [Initialization]
  - $D_0(v) = \infty$, for all $v \neq s$
  - $D_h(s) = 0$, for all h
- Step 2 [Update]
  - For each successive $h \geq 0$:
    - For each $v \neq s$, compute

$$D_{h+1}(v) = \min_{w} [D_h(w) + c(w,v)]$$

; $w = v$'s neighbor