



Algorithms: Design  
and Analysis, Part II

# The Bellman-Ford Algorithm

---

The Basic Algorithm

# The Recurrence

**Notation:** Let  $L_{i,v}$  = minimum length of a  $s$ - $v$  path with  $\leq i$  edges.

- With cycles allowed
- Defined as  $+\infty$  if no  $s$ - $v$  paths with  $\leq i$  edges

**Recurrence:** For every  $v \in V$ ,  $i \in \{1, 2, \dots\}$

$$L_{i,v} = \min \left\{ \begin{array}{ll} L_{(i-1),v} & \text{Case 1} \\ \min_{(u,v) \in E} \{L_{(i-1),w} + c_{wv}\} & \text{Case 2} \end{array} \right\}$$

**Correctness:** Brute-force search from the only  $(1 + \text{in-deg}(v))$  candidates (by the optimal substructure lemma).

# If No Negative Cycles

**Now:** Suppose input graph  $G$  has no negative cycles.

$\Rightarrow$  Shortest paths do not have cycles

[removing a cycle only decreases length]

$\Rightarrow$  Have  $\leq (n - 1)$  edges

**Point:** If  $G$  has no negative cycle, only need to solve subproblems up to  $i = n - 1$ .

**Subproblems:** Compute  $L_{i,v}$  for all  $i \in \{0, 1, \dots, n - 1\}$  and all  $v \in V$ .

# The Bellman-Ford Algorithm

Let  $A$  = 2-D array (indexed by  $i$  and  $v$ )

**Base case:**  $A[0, s] = 0$ ;  $A[0, v] = +\infty$  for all  $v \neq s$ .

For  $i = 1, 2, \dots, n - 1$

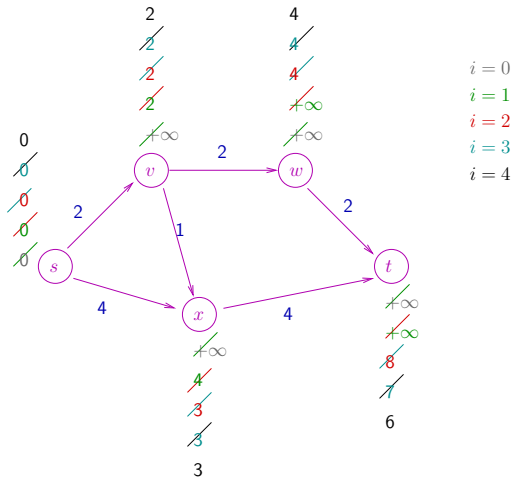
For each  $v \in V$

$$A[i, v] = \min \left\{ \begin{array}{l} A[i - 1, v] \\ \min_{(w, v) \in E} \{A[i - 1, w] + c_{wv}\} \end{array} \right\}$$

**As discussed:** If  $G$  has no negative cycle, then algorithm is correct  
[with final answers =  $A[n - 1, v]$ 's]

# Example

$$A[i, v] = \min \left\{ \begin{array}{l} A[i-1, v] \\ \min_{(w,v) \in E} \{ A[i-1, w] + c_{wv} \} \end{array} \right\}$$



# Quiz

**Question:** What is the running time of the Bellman-Ford algorithm? [Pick the strongest true statement.] [ $m = \#$  of edges,  $n = \#$  of vertices]

A)  $O(n^2) \rightarrow \#$  of subproblems, but might do  $\Theta(n)$  work for one subproblem

B)  $O(mn)$

C)  $O(n^3)$

D)  $O(m^2)$

**Reason:** Total work is  $O\left(n \sum_{v \in V} \text{in-deg}(v)\right) = O(mn)$

# iterations of outer loop (i.e. choices of  $i$ )

work done in one iteration =  $m$

# Stopping Early

**Note:** Suppose for some  $j < n - 1$ ,  $A[j, v] = A[j - 1, v]$  for all vertices  $v$ .

$\Rightarrow$  For all  $v$ , all future  $A[i, v]$ 's will be the same

$\Rightarrow$  Can safely halt (since  $A[n - 1, v]$ 's = correct shortest-path distances)