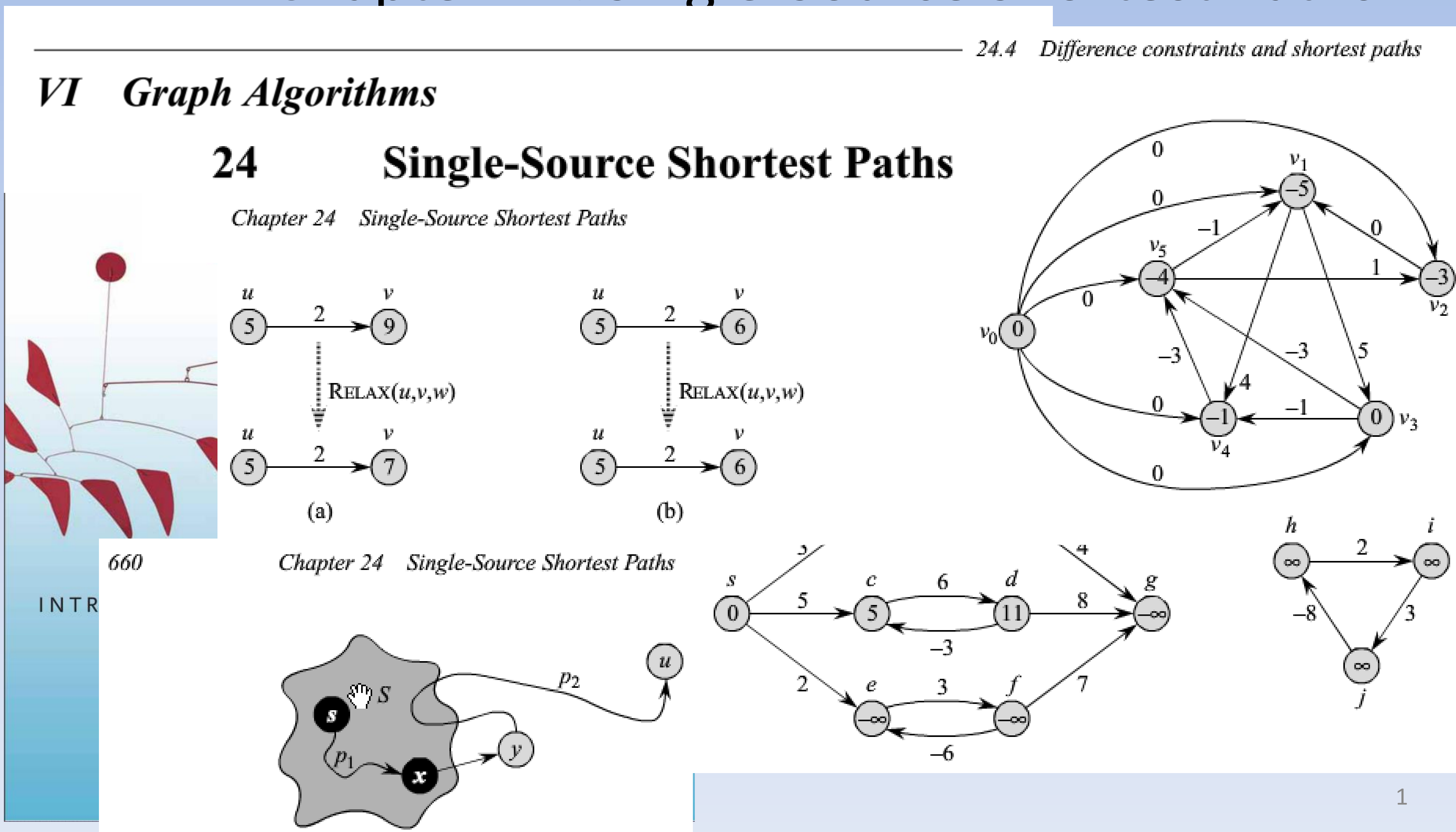
Design and Analysis of Algorithms

Section VI : Graph Algorithms

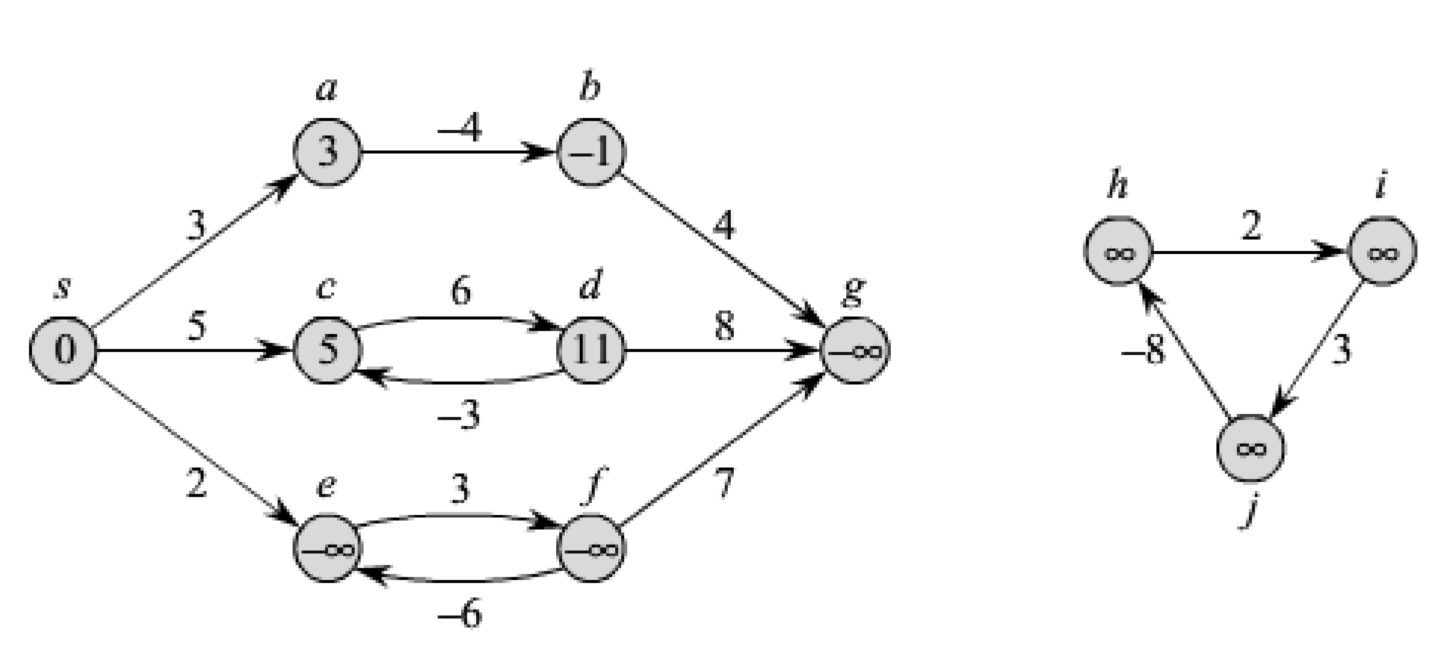
Chapter 24: Single-Source Shortest Paths



Shortest-Paths Problem

- Given:
  - Weighted, Directed Graph $G=(V, E)$
  - Weight Function $w: E \rightarrow \mathbb{R}$
    - Edges -> Real-Valued Weights
- Weight of path $P=<v_0, v_1, \ldots, v_k>$
  - $w(p) = \sum w(v_{i-1}, v_i)$
- Shortest-Path Weight $\delta(u,v)$ is the minimum weight path $w(p)$ that goes from u to v, otherwise ∞
- The shortest path from u to v is any path p with a weight of $\delta(u,v)$

Negative Weight Edges

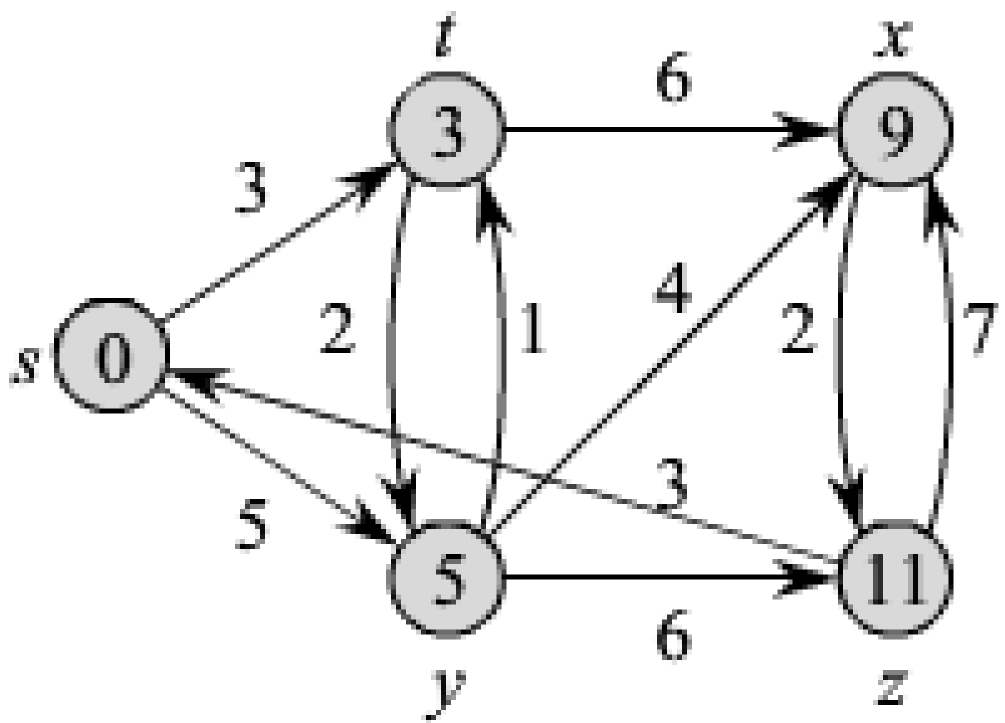


- No Negative Weight Cycles
- Shortest Paths can never contain a cycle.

Shortest Paths Representation

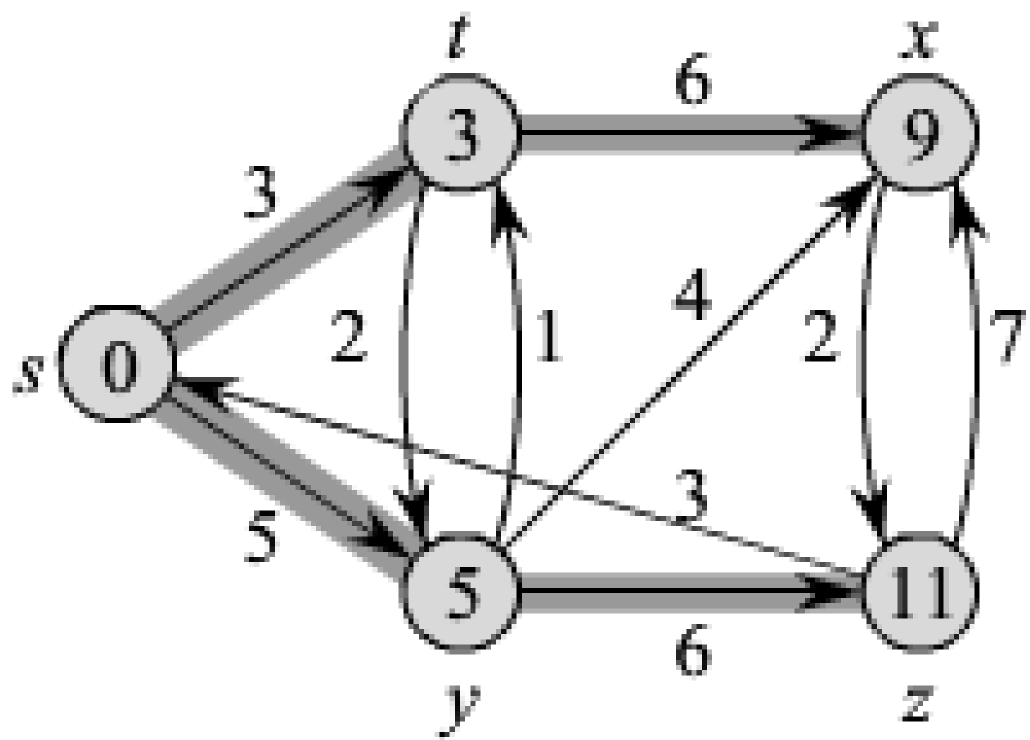
- Given a graph G=(V, E), for all v ∈ V:
  - Predecessor is maintained as **v.π**
  - At completion of Shortest Paths Algorithm, v.π stores the shortest path from s to v backwards from v to s.

Weighted, Directed Graph



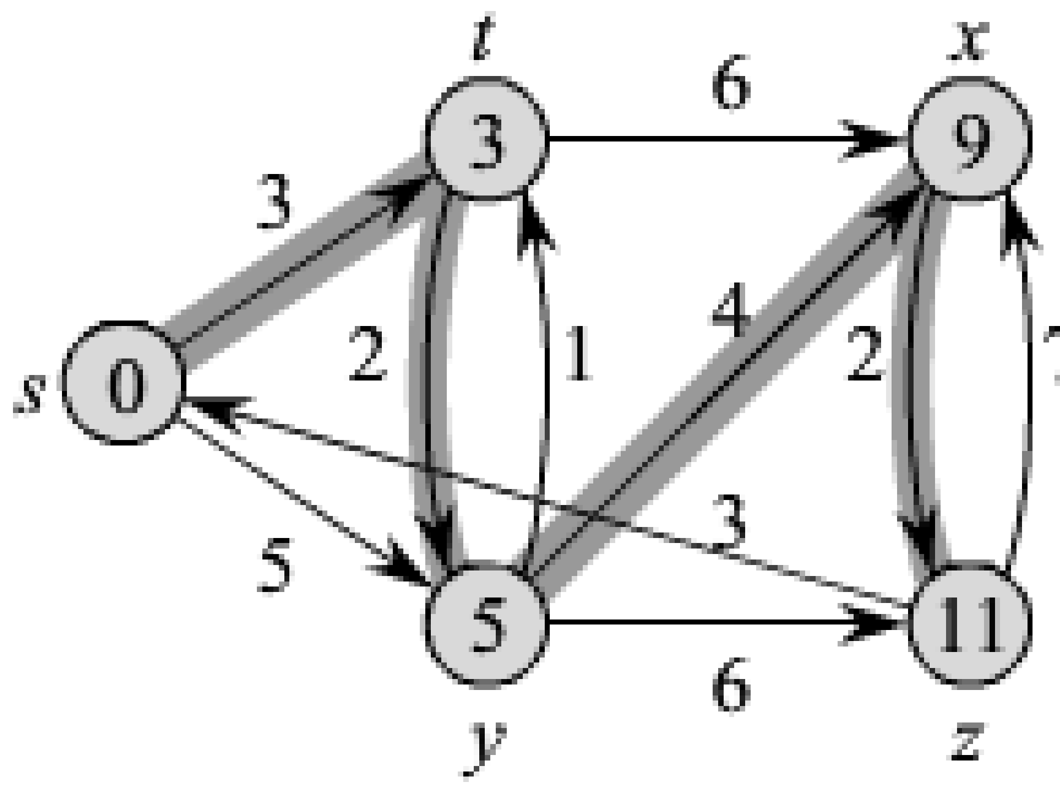
(a)

Weighted, Directed Graph
Shortest-Paths Tree Rooted @ S



(b)

Weighted, Directed Graph
2^{nd} Shortest-Paths Tree Rooted @ S



(c)

Relaxation & V.D

- Textbook Algorithms for Shortest Paths use Relaxation Technique
- Vertices have an attribute v.d that is an upper bound on their shortest path weight from source vertex.
- **V.D** is called **Shortest-Path Estimate**

Relaxation
Step 1: Initialize-Single-Source

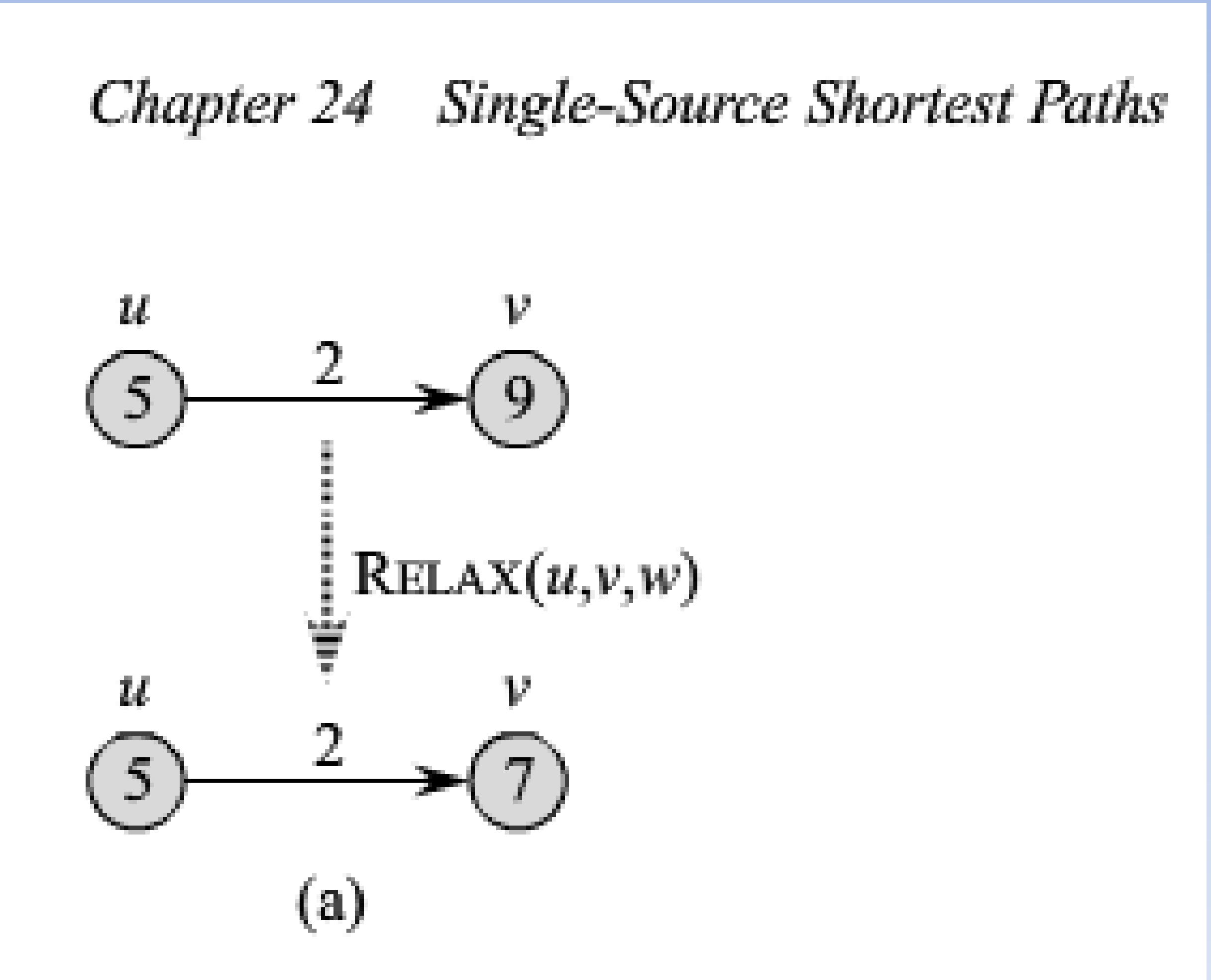
INITIALIZE-SINGLE-SOURCE(G, s)

```
for each vertex v ∈ G.V
    v.d = ∞
    v.π = NIL
s.d = 0
```

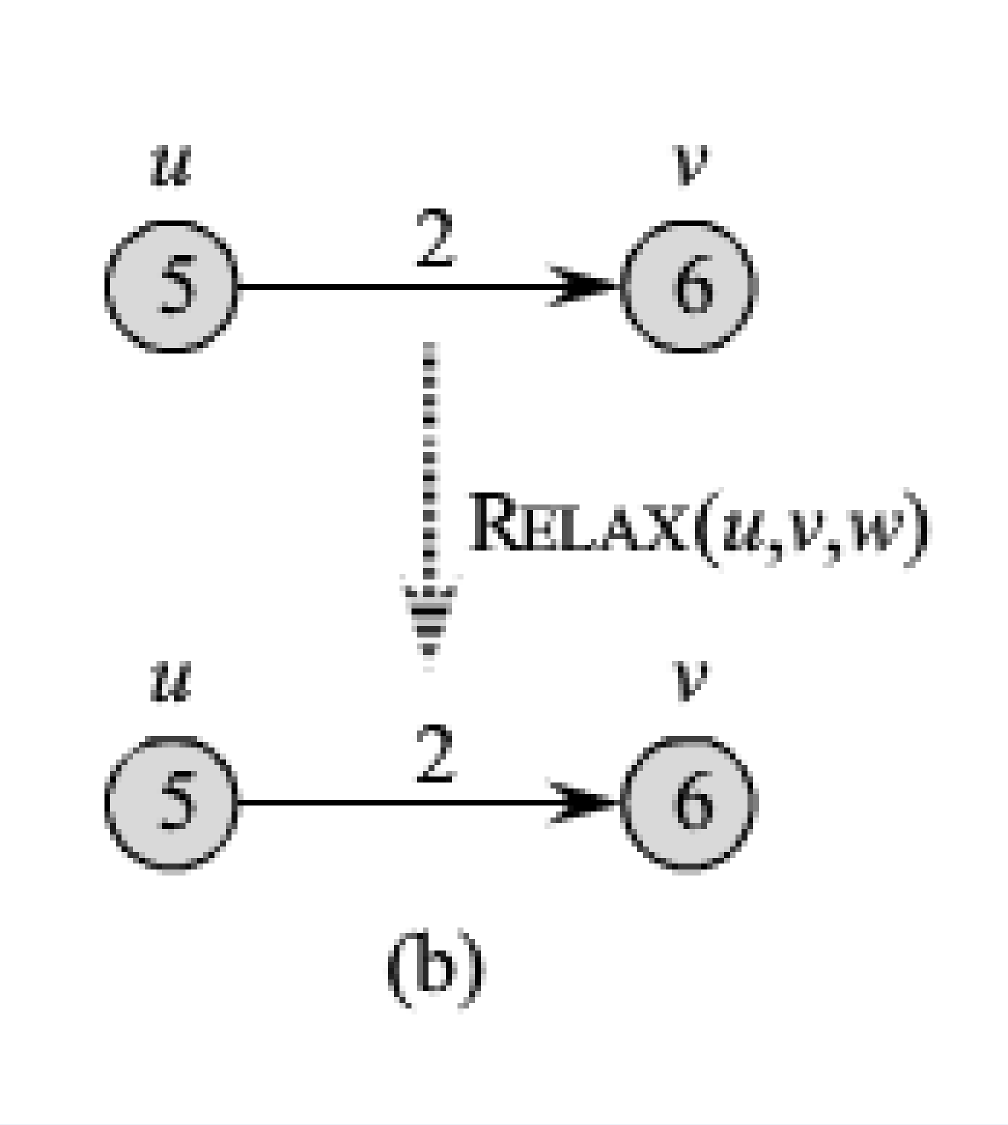
Relaxation Process

- Relax Edge (u, v) By:
  - Testing possible shortest path improvement to **v** by using current path to **u**
  - When improvements are possible update:
    - v.d: estimated shortest-path weight
    - v.π: v's parent

Relax Edge


Chapter 24 Single-Source Shortest Paths
u
v
2
5
9
RELAX(u,v,w)
u
v
2
5
7
(a)

Relax Edge (not possible)



Relax (Pseudocode)

$\textsc{Relax}(u, v, w)$

if $v.d > u.d + w(u, v)$
  $v.d = u.d + w(u, v)$
  $v.\pi = u$

Relax (Pseudocode)

$\text{RELAX}(u, v, w)$

$\textbf{if } v.d > u.d + w(u, v)$

$\quad v.d = u.d + w(u, v)$

$\quad v.\pi = u$

Update Path Weight

Relax (Pseudocode)

```
RELAX(u, v, w)
    if v.d > u.d + w(u, v)
        v.d = u.d + w(u, v)
        v.π = u
```

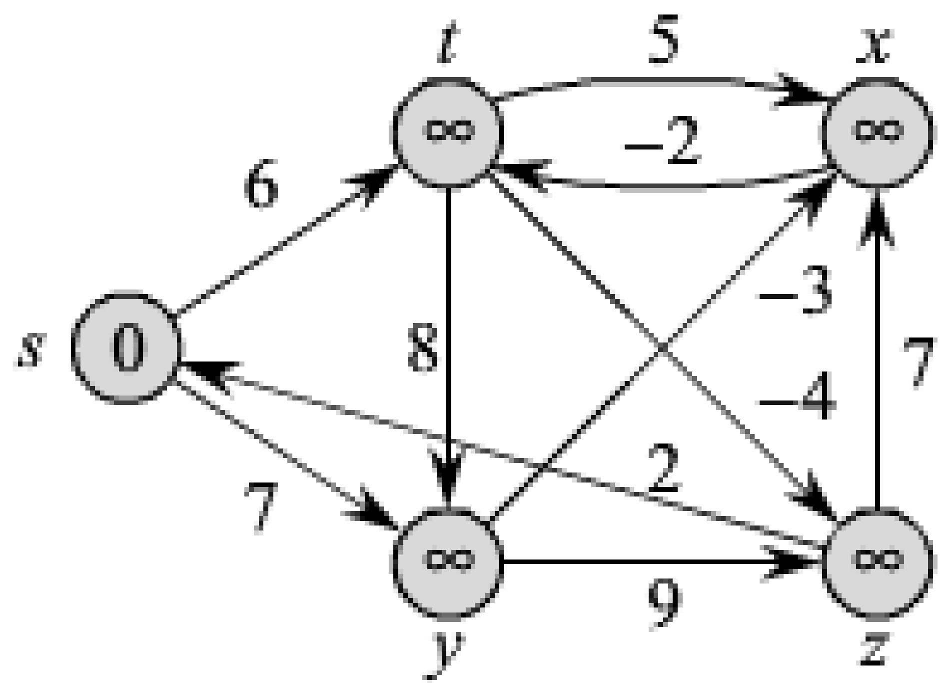
Parent Updated

Bellman-Ford

- Solves Single-Source Shortest-Paths Problem in General Case.
  - Weights may be negative.
- Given Graph G=(V,E) with source s and weight function w returns:
  - False: if negative-weight cycle exist
  - True: Otherwise

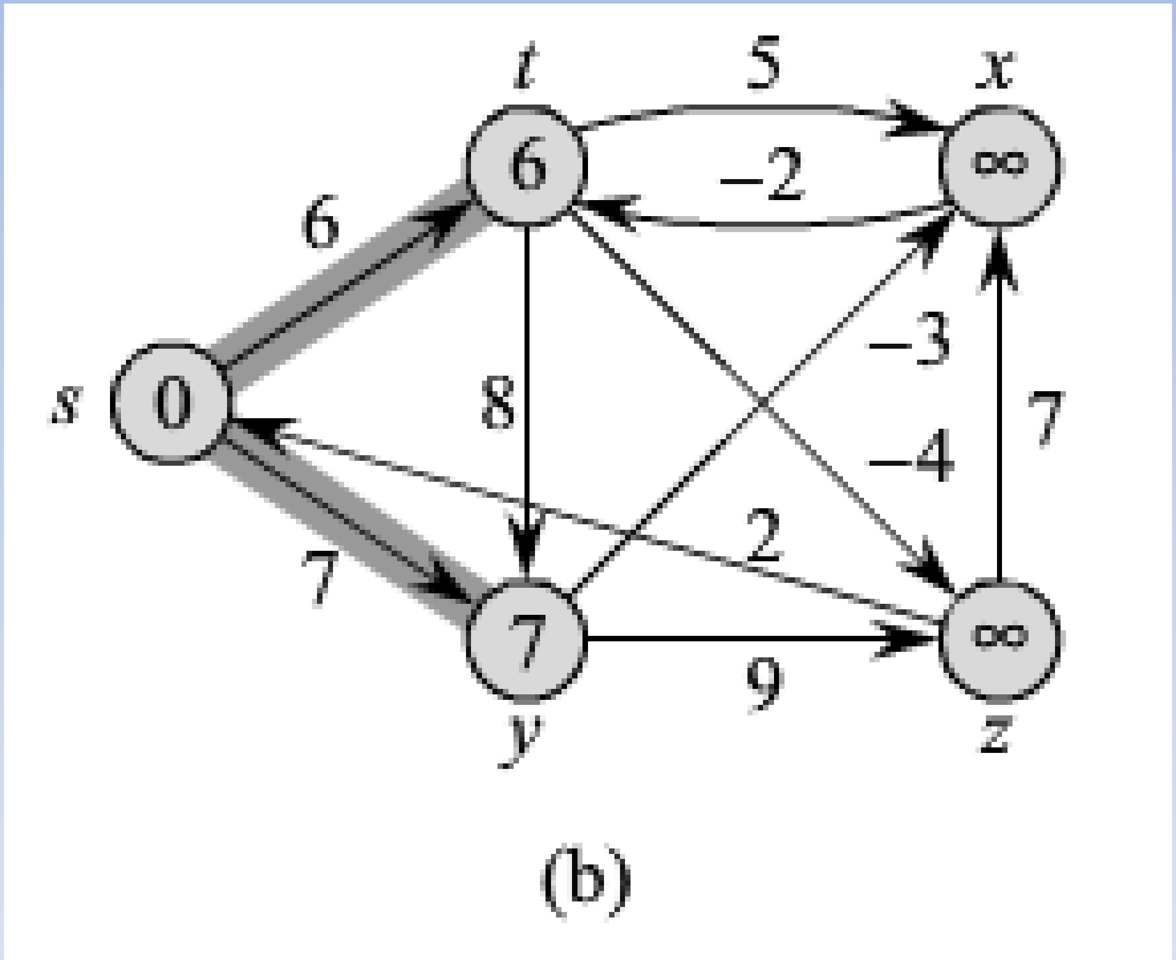
```
BELLMAN-FORD(G, w, s)
INITIALIZE-SINGLE-SOURCE(G, s)
for i = 1 to |G.V| - 1
    for each edge (u, v) ∈ G.E
        RELAX(u, v, w)
for each edge (u, v) ∈ G.E
    if v.d > u.d + w(u, v)
        return FALSE
return TRUE
```

Bellman-Ford Example



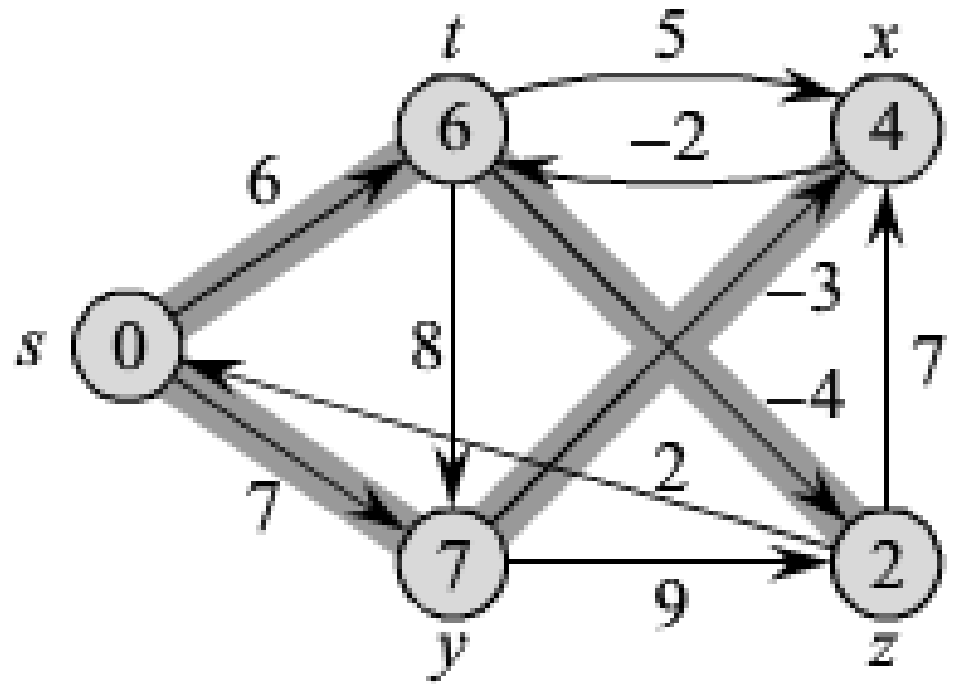
(a)

Bellman-Ford Example



- First Iteration: t & y relaxed

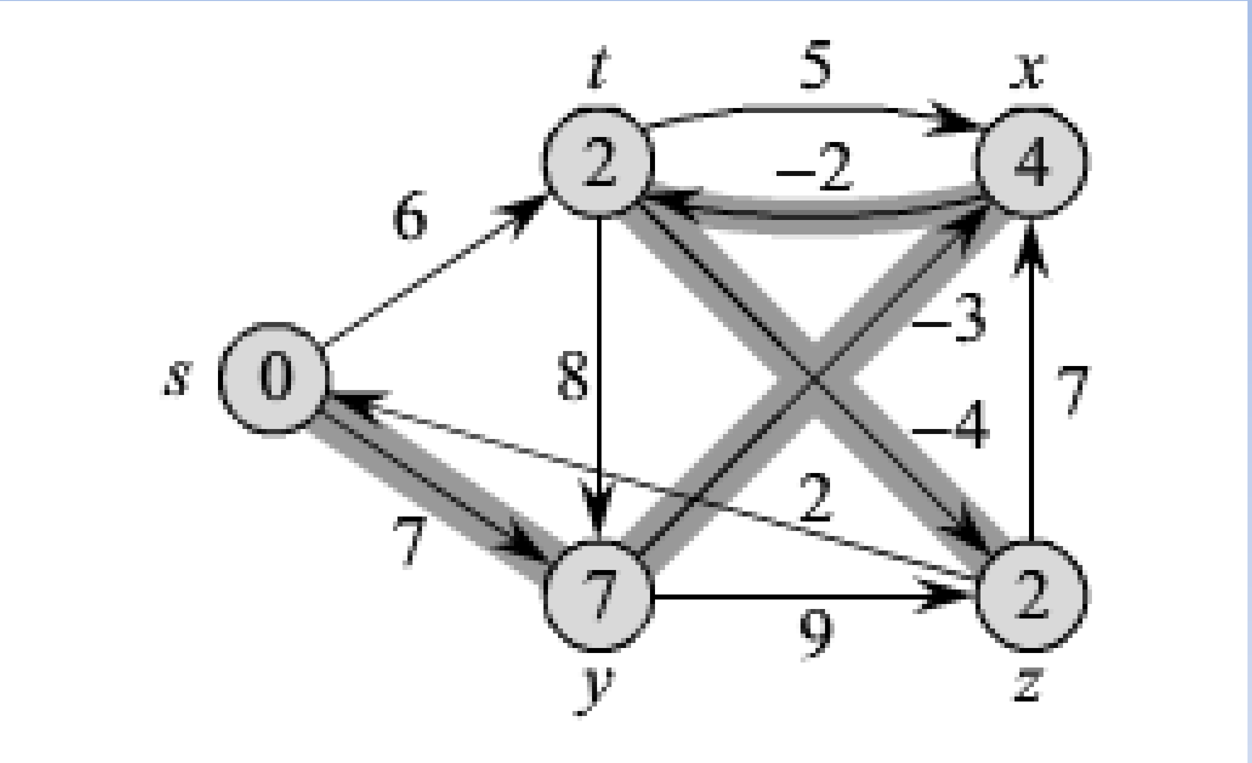
Bellman-Ford Example



(c)

- Second Iteration: x & z relaxed

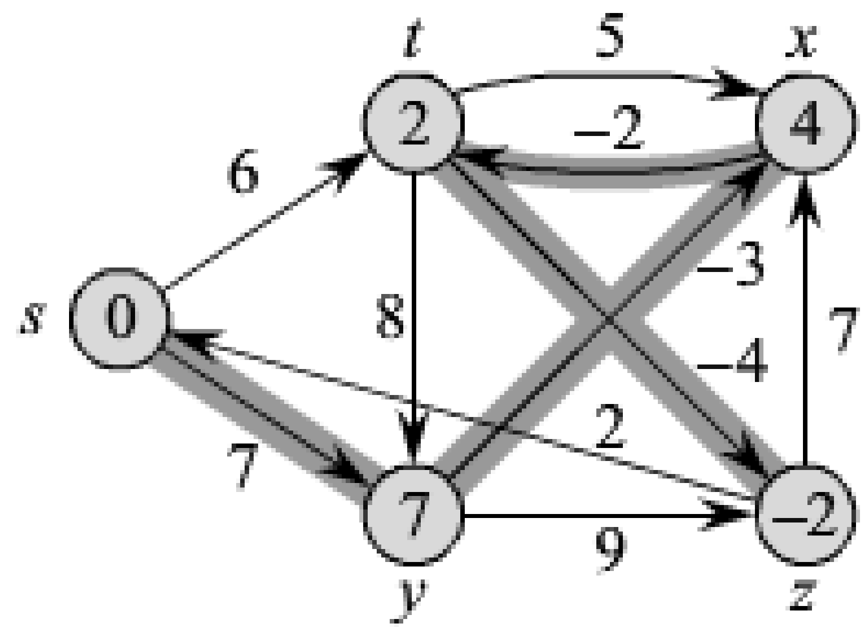
Bellman-Ford Example



(d)

- Third Iteration: t relaxed
  - **Via: s, y, x, t**

Bellman-Ford Example



(e)

- Fourth Iteration: z relaxed
  - **Via: s, y, x, z**

Bellman-Ford Complexity

```
BELLMAN-FORD(G, w, s)
1  INITIALIZE-SINGLE-SOURCE(G, s)
2  for i = 1 to |G.V| - 1
3      for each edge (u, v) ∈ G.E
4          RELAX(u, v, w)
5  for each edge (u, v) ∈ G.E
6      if v.d > u.d + w(u, v)
7          return FALSE
8  return TRUE
```

- Lines 2-4: |V|-1 passes over Edges in E
- O(VE)

Complexity Improvements

- Assume Directed Acyclic Graph (DAG)
- Begin by Topologically Sorting Vertices
- Make one pass over ordered vertices and relax their edges.

Complexity Improvements

- Assume Directed Acyclic Graph (DAG)
- Begin by Topologically Sorting Vertices
- Make one pass over ordered vertices and relax their edges.

DAG-SHORTEST-PATHS(G, w, s)

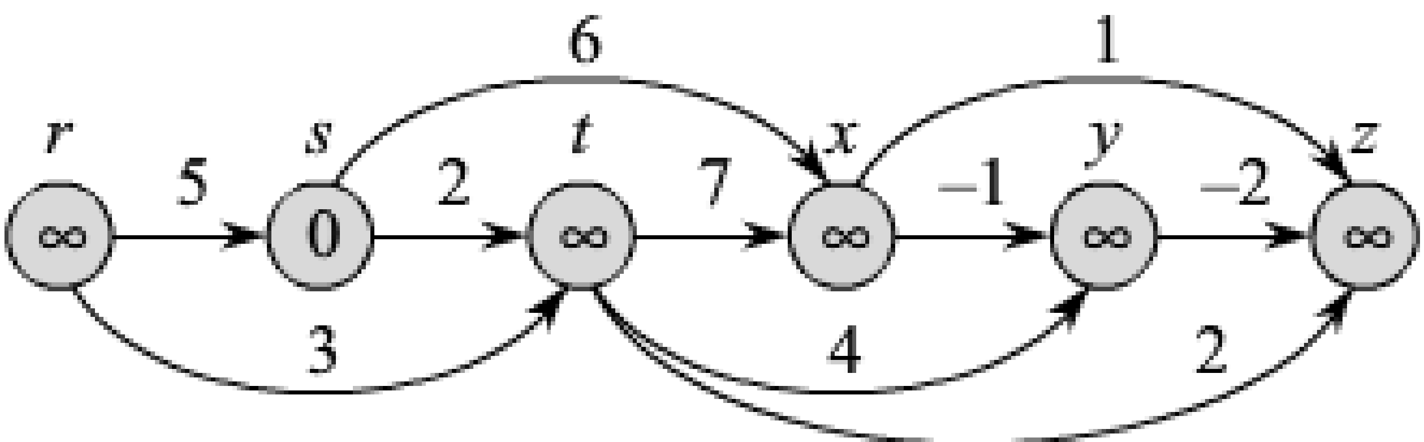
topologically sort the vertices of G
INITIALIZE-SINGLE-SOURCE(G, s)
for each vertex u, taken in topologically sorted order
    for each vertex $v \in G.Adj[u]$
        RELAX(u, v, w)

Complexity Improvements
Running Time: O(V + E)

- Assume Directed Acyclic Graph (DAG)
- Begin by Topologically Sorting Vertices
- Make one pass over ordered vertices and relax their edges.

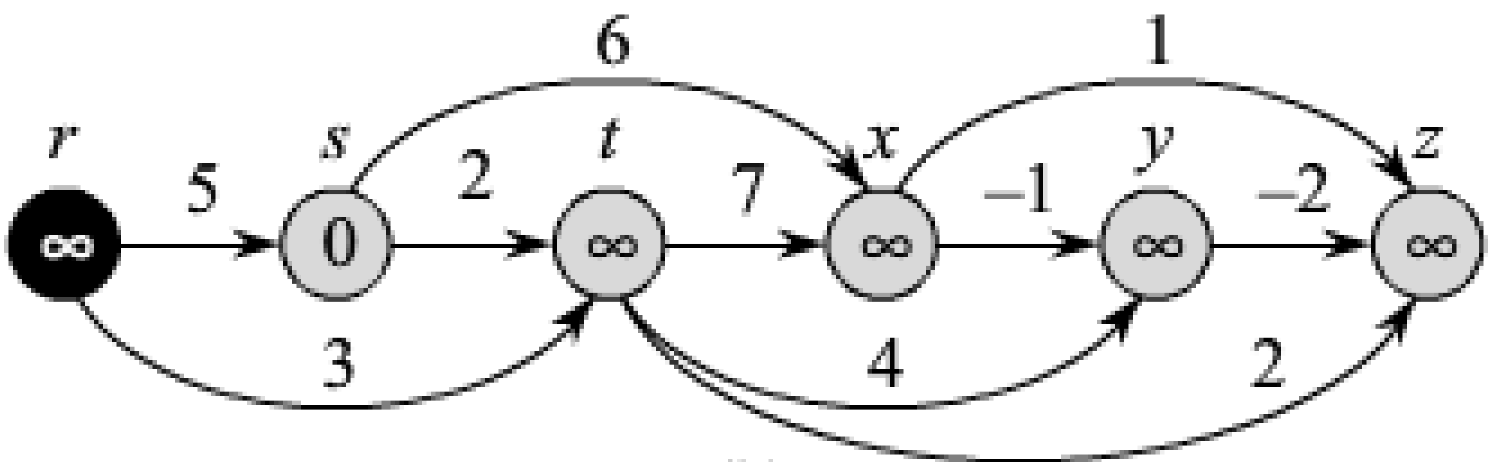
```
DAG-SHORTEST-PATHS(G, w, s)
topologically sort the vertices of G
INITIALIZE-SINGLE-SOURCE(G, s)
for each vertex u, taken in topologically sorted order
    for each vertex v ∈ G.Adj[u]
        RELAX(u, v, w)
```

DAG-Shortest-Paths Example



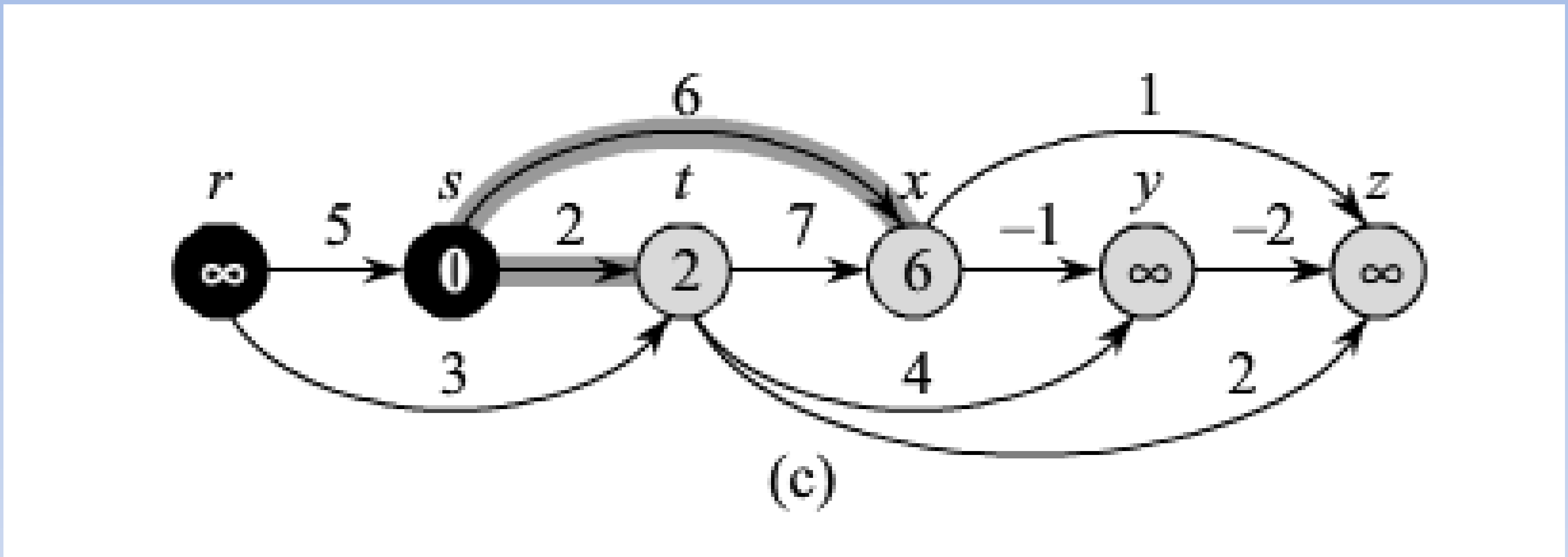
(a)

DAG-Shortest-Paths Example

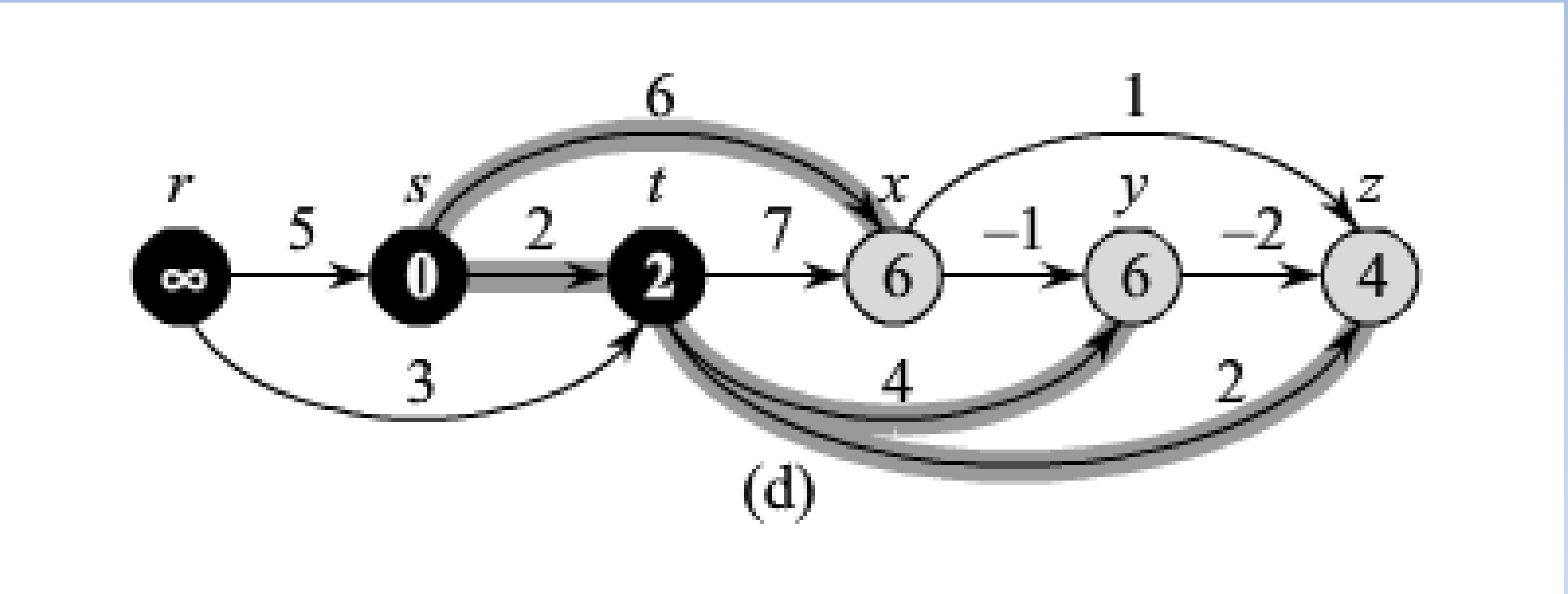


(b)

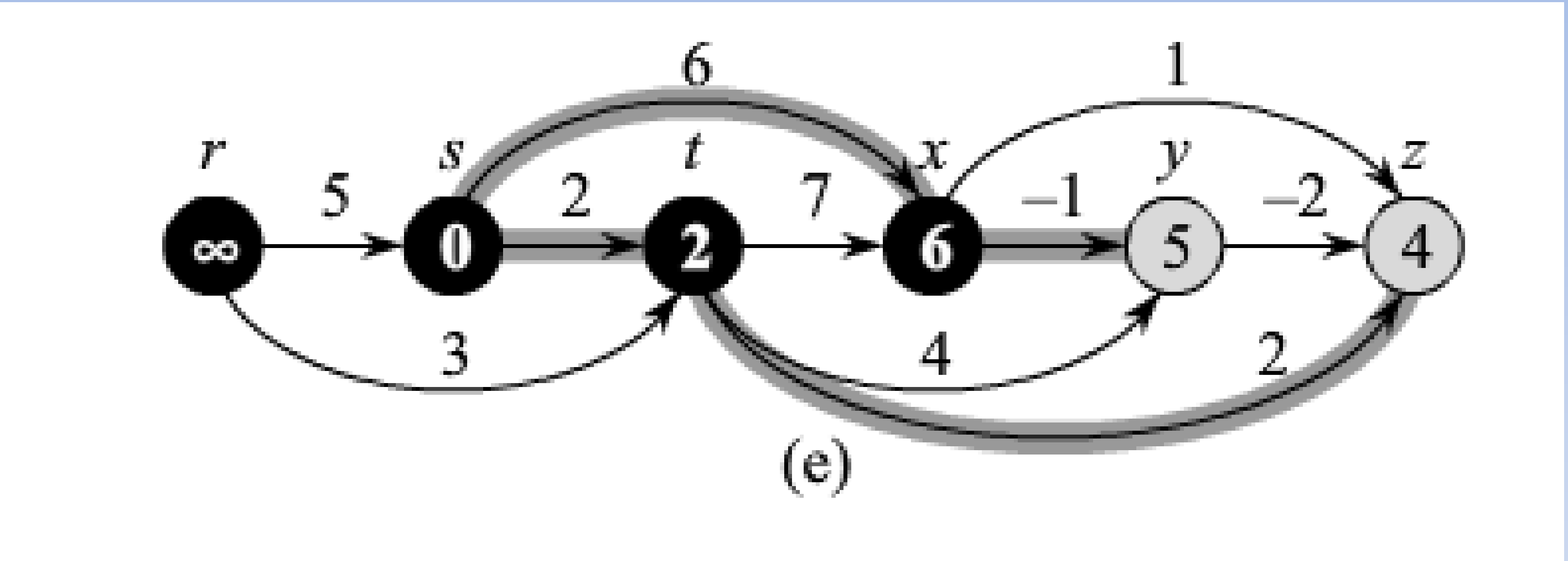
DAG-Shortest-Paths Example



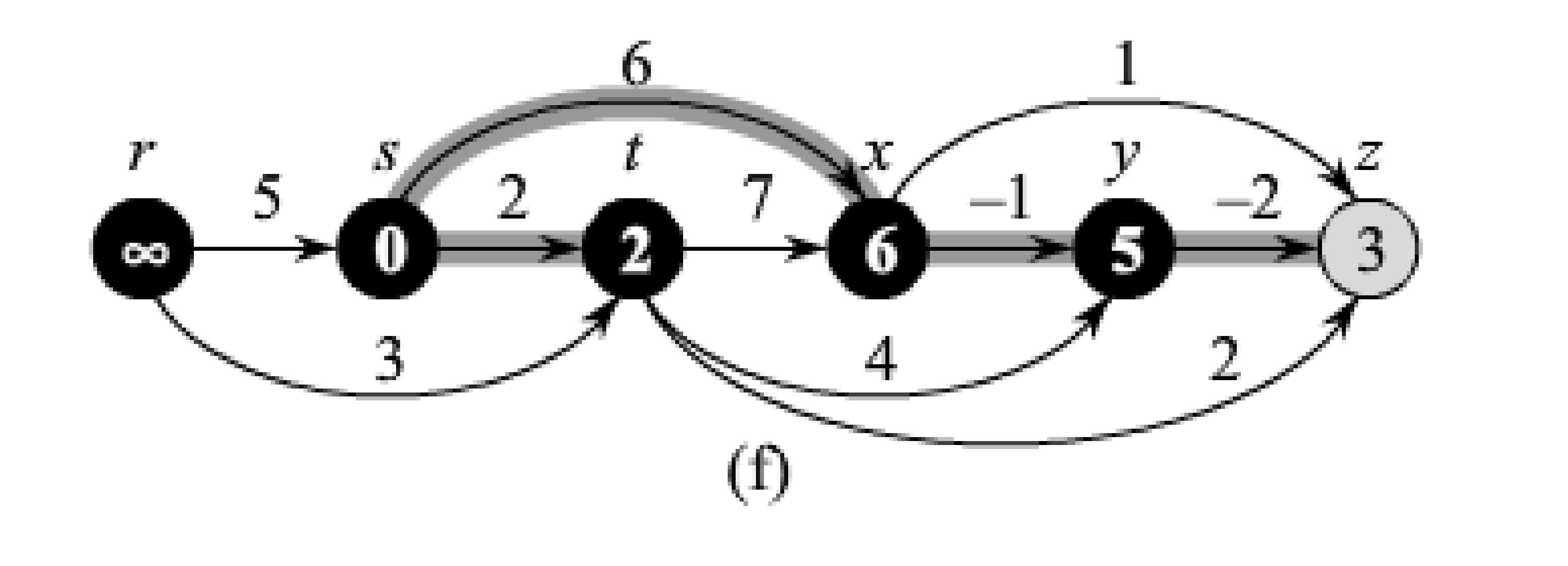
DAG-Shortest-Paths Example



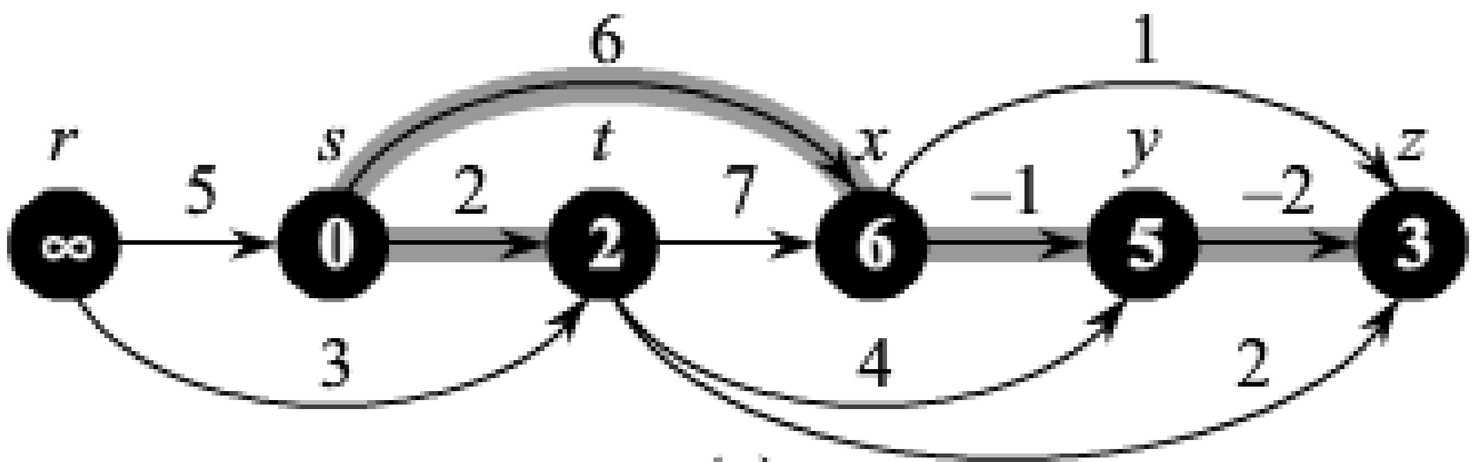
DAG-Shortest-Paths Example



DAG-Shortest-Paths Example



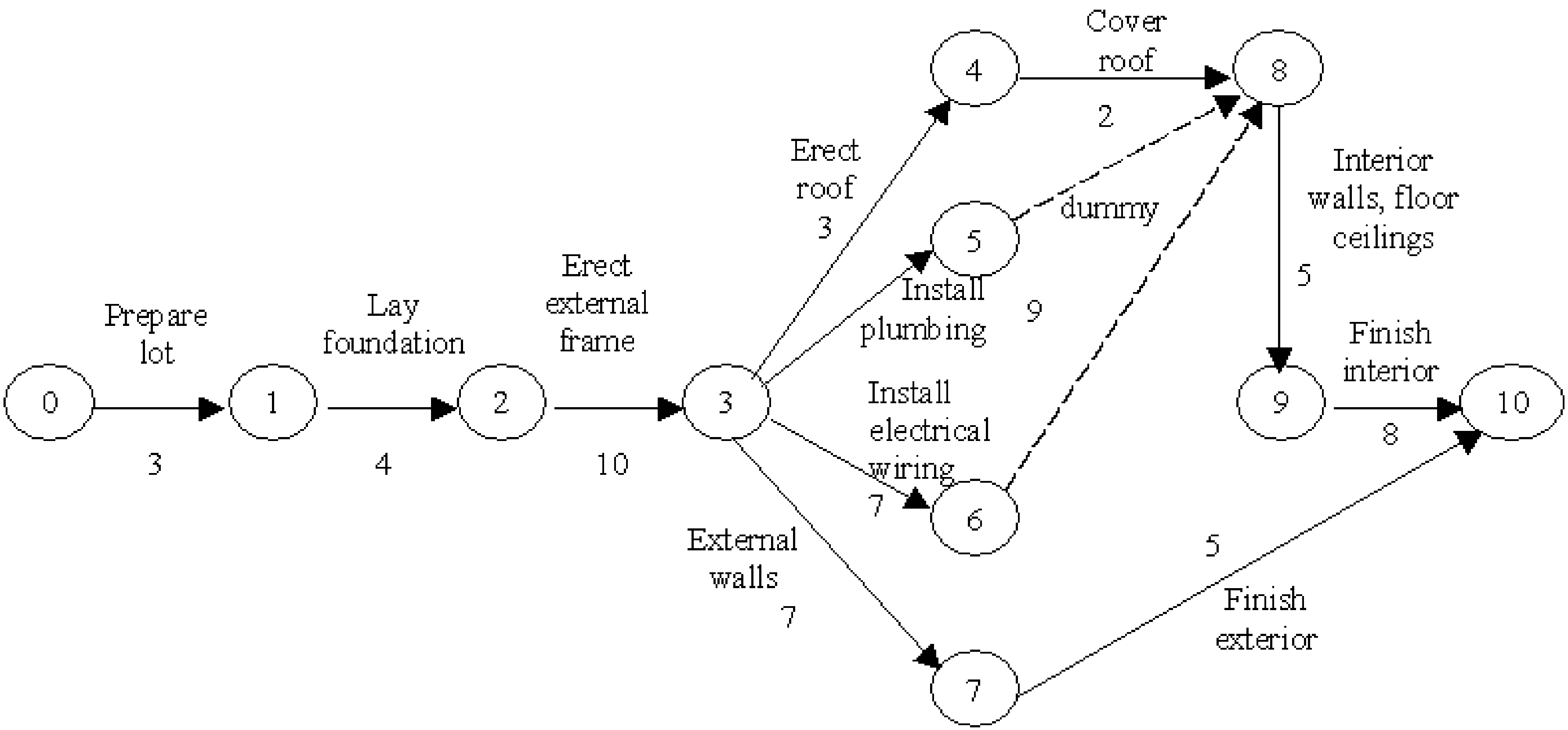
DAG-Shortest-Paths Example

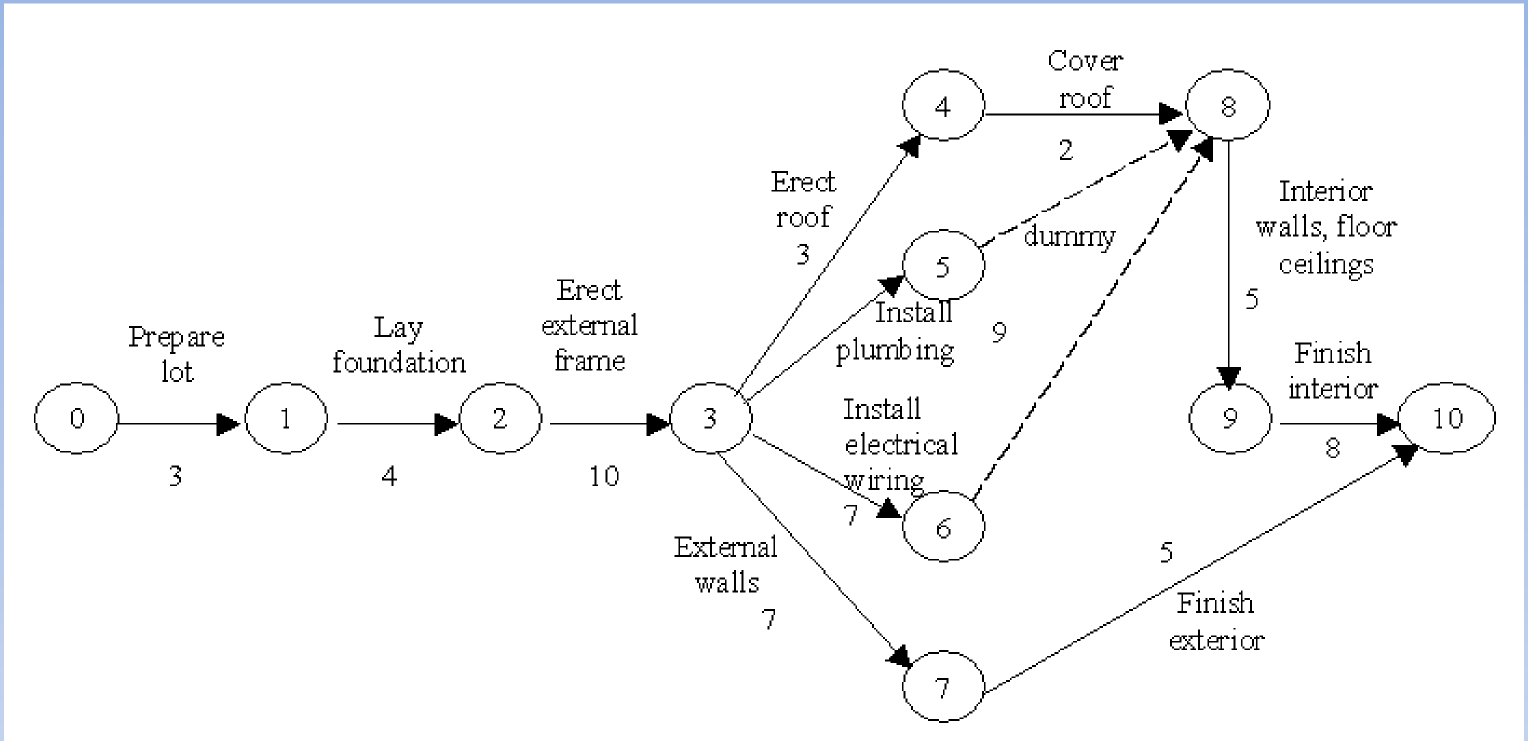


(g)

PERT Charts

- Program Evaluation Review Technique
- Used to schedule, organize, coordinate tasks within a project.


0
1
2
3
4
5
6
7
8
9
10
Prepare lot
3
Lay foundation
4
Erect external frame
10
Erect roof
3
Cover roof
2
dummy
Install plumbing
9
Install electrical wiring
7
External walls
7
Interior walls, floor ceilings
5
Finish interior
8
Finish exterior
5



- Critical Path is a longest path through DAG.
- Find Critical Path:
  - Negating weights and using Dag-Shortest-Path
  - Use Dag-Shortest-Path with modifications:
    - Replace infinity with –infinity in line 2 of Initialize-Single-Source
    - Replace > with < in the Relax Procedure.

Optimal Substructure

- Subpaths of Shortest Paths are ALSO Shortest Paths
  - Otherwise, it's improvable by replacing the subpath with its shorter version.

Optimal Substructure

Lemma 24.1 (Subpaths of shortest paths are shortest paths)
Given a weighted, directed graph $G = (V, E)$ with weight function $w : E \to \mathbb{R}$, let $p = \langle v_0, v_1, \ldots, v_k \rangle$ be a shortest path from vertex v_0 to vertex v_k and, for any i and j such that $0 \le i \le j \le k$, let $p_{ij} = \langle v_i, v_{i+1}, \ldots, v_j \rangle$ be the subpath of p from vertex v_i to vertex v_j. Then, p_{ij} is a shortest path from v_i to v_j.

Proof If we decompose path p into $v_0 \overset{p_{0i}}{\leadsto} v_i \overset{p_{ij}}{\leadsto} v_j \overset{p_{jk}}{\leadsto} v_k$, then we have that $w(p) = w(p_{0i}) + w(p_{ij}) + w(p_{jk})$. Now, assume that there is a path p'_{ij} from v_i to v_j with weight $w(p'_{ij}) < w(p_{ij})$. Then, $v_0 \overset{p_{0i}}{\leadsto} v_i \overset{p'_{ij}}{\leadsto} v_j \overset{p_{jk}}{\leadsto} v_k$ is a path from v_0 to v_k whose weight $w(p_{0i}) + w(p'_{ij}) + w(p_{jk})$ is less than $w(p)$, which contradicts the assumption that p is a shortest path from v_0 to v_k. ■

Properties

Triangle inequality (Lemma 24.10)
For any edge $(u, v) \in E$, we have $\delta(s, v) \leq \delta(s, u) + w(u, v)$.

Upper-bound property (Lemma 24.11)
We always have $v.d \geq \delta(s, v)$ for all vertices $v \in V$, and once $v.d$ achieves the value $\delta(s, v)$, it never changes.

No-path property (Corollary 24.12)
If there is no path from s to v, then we always have $v.d = \delta(s, v) = \infty$.

Convergence property (Lemma 24.14)
If $s \leadsto u \to v$ is a shortest path in G for some $u, v \in V$, and if $u.d = \delta(s, u)$ at any time prior to relaxing edge (u, v), then $v.d = \delta(s, v)$ at all times afterward.

Path-relaxation property (Lemma 24.15)
If $p = \langle v_0, v_1, \ldots, v_k \rangle$ is a shortest path from $s = v_0$ to v_k, and we relax the edges of p in the order $(v_0, v_1), (v_1, v_2), \ldots, (v_{k-1}, v_k)$, then $v_k.d = \delta(s, v_k)$. This property holds regardless of any other relaxation steps that occur, even if they are intermixed with relaxations of the edges of p.

Predecessor-subgraph property (Lemma 24.17)
Once $v.d = \delta(s, v)$ for all $v \in V$, the predecessor subgraph is a shortest-paths tree rooted at s.

Lemma 24.2

Let $G = (V, E)$ be a weighted, directed graph with source s and weight function $w : E \to \mathbb{R}$, and assume that G contains no negative-weight cycles that are reachable from s. Then, after the $|V| - 1$ iterations of the **for** loop of lines 2–4 of BELLMAN-FORD, we have $v.d = \delta(s, v)$ for all vertices v that are reachable from s.

Proof We prove the lemma by appealing to the path-relaxation property. Consider any vertex v that is reachable from s, and let $p = \langle v_0, v_1, \ldots, v_k \rangle$, where $v_0 = s$ and $v_k = v$, be any shortest path from s to v. Because shortest paths are simple, p has at most $|V| - 1$ edges, and so $k \leq |V| - 1$. Each of the $|V| - 1$ iterations of the **for** loop of lines 2–4 relaxes all $|E|$ edges. Among the edges relaxed in the ith iteration, for $i = 1, 2, \ldots, k$, is (v_{i-1}, v_i). By the path-relaxation property, therefore, $v.d = v_k.d = \delta(s, v_k) = \delta(s, v)$. ■

Relaxation Property

Path-relaxation property (Lemma 24.15)

If $p = \langle v_0, v_1, \ldots, v_k \rangle$ is a shortest path from $s = v_0$ to v_k, and we relax the edges of p in the order $(v_0, v_1), (v_1, v_2), \ldots, (v_{k-1}, v_k)$, then $v_k.d = \delta(s, v_k)$. This property holds regardless of any other relaxation steps that occur, even if they are intermixed with relaxations of the edges of p.

Upper-Bound Property

Lemma 24.11 (Upper-bound property)

Let $G = (V, E)$ be a weighted, directed graph with weight function $w : E \to \mathbb{R}$. Let $s \in V$ be the source vertex, and let the graph be initialized by INITIALIZE-SINGLE-SOURCE(G, s). Then, $v.d \geq \delta(s, v)$ for all $v \in V$, and this invariant is maintained over any sequence of relaxation steps on the edges of G. Moreover, once $v.d$ achieves its lower bound $\delta(s, v)$, it never changes.

Invariant: $v.d \geq \delta(s, v)$

Proof We prove the invariant $v.d \geq \delta(s, v)$ for all vertices $v \in V$ by induction over the number of relaxation steps.

For the basis, $v.d \geq \delta(s, v)$ is certainly true after initialization, since $v.d = \infty$ implies $v.d \geq \delta(s, v)$ for all $v \in V - \{s\}$, and since $s.d = 0 \geq \delta(s, s)$ (note that $\delta(s, s) = -\infty$ if s is on a negative-weight cycle and 0 otherwise).

For the inductive step, consider the relaxation of an edge (u, v). By the inductive hypothesis, $x.d \geq \delta(s, x)$ for all $x \in V$ prior to the relaxation. The only d value that may change is $v.d$. If it changes, we have

$$\begin{aligned} v.d &= u.d + w(u, v) \\ &\geq \delta(s, u) + w(u, v) && \text{(by the inductive hypothesis)} \\ &\geq \delta(s, v) && \text{(by the triangle inequality)}\ , \end{aligned}$$

and so the invariant is maintained.

To see that the value of $v.d$ never changes once $v.d = \delta(s, v)$, note that having achieved its lower bound, $v.d$ cannot decrease because we have just shown that $v.d \geq \delta(s, v)$, and it cannot increase because relaxation steps do not increase d values. ∎

Corollary 24.12 (No-path property)
Suppose that in a weighted, directed graph $G = (V, E)$ with weight function $w : E \to \mathbb{R}$, no path connects a source vertex $s \in V$ to a given vertex $v \in V$. Then, after the graph is initialized by INITIALIZE-SINGLE-SOURCE(G, s), we have $v.d = \delta(s, v) = \infty$, and this equality is maintained as an invariant over any sequence of relaxation steps on the edges of G.

Proof By the upper-bound property, we always have $\infty = \delta(s, v) \leq v.d$, and thus $v.d = \infty = \delta(s, v)$. ■

Lemma 24.13
Let $G = (V, E)$ be a weighted, directed graph with weight function $w : E \to \mathbb{R}$, and let $(u, v) \in E$. Then, immediately after relaxing edge (u, v) by executing RELAX(u, v, w), we have $v.d \leq u.d + w(u, v)$.

Proof If, just prior to relaxing edge (u, v), we have $v.d > u.d + w(u, v)$, then $v.d = u.d + w(u, v)$ afterward. If, instead, $v.d \leq u.d + w(u, v)$ just before the relaxation, then neither $u.d$ nor $v.d$ changes, and so $v.d \leq u.d + w(u, v)$ afterward. ■

Lemma 24.14 (Convergence property)

Let $G = (V, E)$ be a weighted, directed graph with weight function $w : E \to \mathbb{R}$, let $s \in V$ be a source vertex, and let $s \leadsto u \to v$ be a shortest path in G for some vertices $u, v \in V$. Suppose that G is initialized by INITIALIZE-SINGLE-SOURCE(G, s) and then a sequence of relaxation steps that includes the call RELAX(u, v, w) is executed on the edges of G. If $u.d = \delta(s, u)$ at any time prior to the call, then $v.d = \delta(s, v)$ at all times after the call.

Proof By the upper-bound property, if $u.d = \delta(s, u)$ at some point prior to relaxing edge (u, v), then this equality holds thereafter. In particular, after relaxing edge (u, v), we have

$$
\begin{aligned}
v.d &\leq u.d + w(u, v) && \text{(by Lemma 24.13)} \\
&= \delta(s, u) + w(u, v) \\
&= \delta(s, v) && \text{(by Lemma 24.1)} .
\end{aligned}
$$

By the upper-bound property, $v.d \geq \delta(s, v)$, from which we conclude that $v.d = \delta(s, v)$, and this equality is maintained thereafter. ■

Using Convergence Property

Lemma 24.15 (Path-relaxation property)
Let $G = (V, E)$ be a weighted, directed graph with weight function $w : E \to \mathbb{R}$, and let $s \in V$ be a source vertex. Consider any shortest path $p = \langle v_0, v_1, \ldots, v_k \rangle$ from $s = v_0$ to v_k. If G is initialized by INITIALIZE-SINGLE-SOURCE(G, s) and then a sequence of relaxation steps occurs that includes, in order, relaxing the edges $(v_0, v_1), (v_1, v_2), \ldots, (v_{k-1}, v_k)$, then $v_k.d = \delta(s, v_k)$ after these relaxations and at all times afterward. This property holds no matter what other edge relaxations occur, including relaxations that are intermixed with relaxations of the edges of p.

Proof We show by induction that after the ith edge of path p is relaxed, we have $v_i.d = \delta(s, v_i)$. For the basis, $i = 0$, and before any edges of p have been relaxed, we have from the initialization that $v_0.d = s.d = 0 = \delta(s, s)$. By the upper-bound property, the value of $s.d$ never changes after initialization.

For the inductive step, we assume that $v_{i-1}.d = \delta(s, v_{i-1})$, and we examine what happens when we relax edge (v_{i-1}, v_i). By the convergence property, after relaxing this edge, we have $v_i.d = \delta(s, v_i)$, and this equality is maintained at all times thereafter. ■

Shortest-Path Weights Imply Shortest-Path Sub-Graph G_π

- Lemma 24.16 shows that the G_π is a rooted tree at s.
- Lemma 24.17 shows the Bellman-Ford constructs G_π with the shortest-paths.