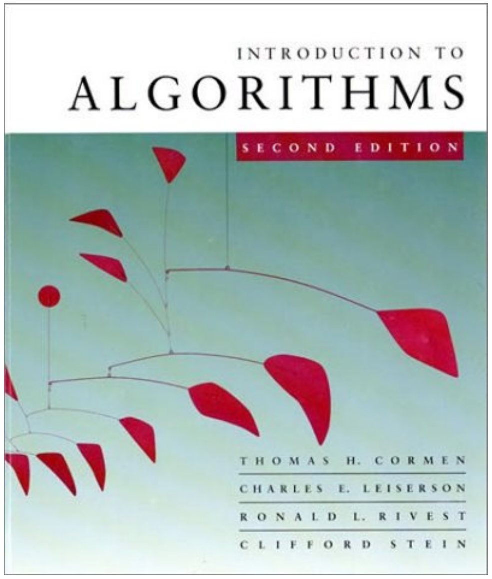
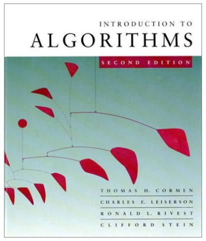
# *Introduction to Algorithms*

## 6.006



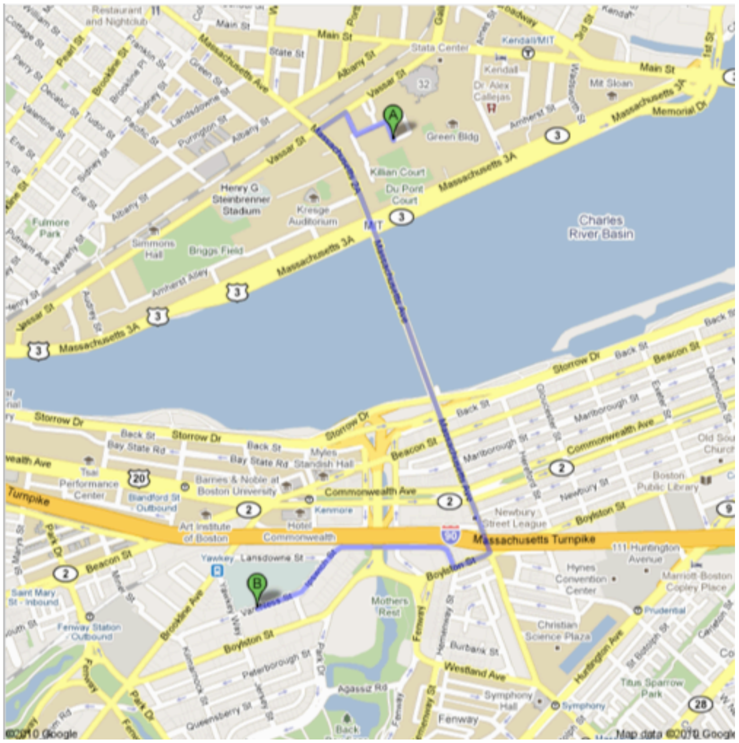
## *Lecture 16*

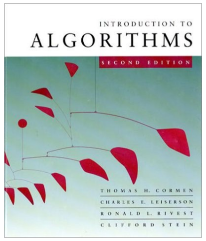
**Prof. Piotr Indyk**



# Menu

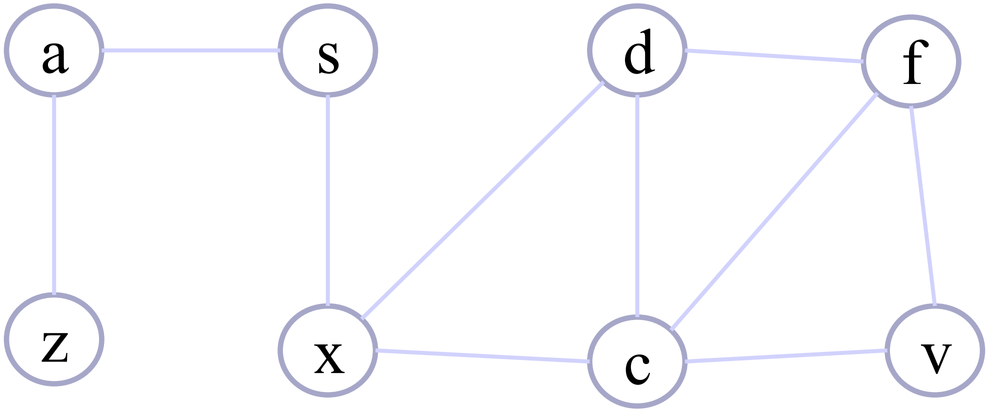
- Last week: Bellman-Ford
  - $O(VE)$ time
  - general weights
- Today: Dijkstra
  - $O((V+E)\log V)$ time
  - non-negative weights





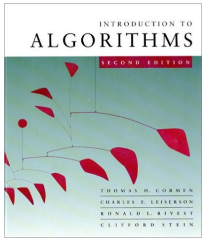
# Single source shortest path problem

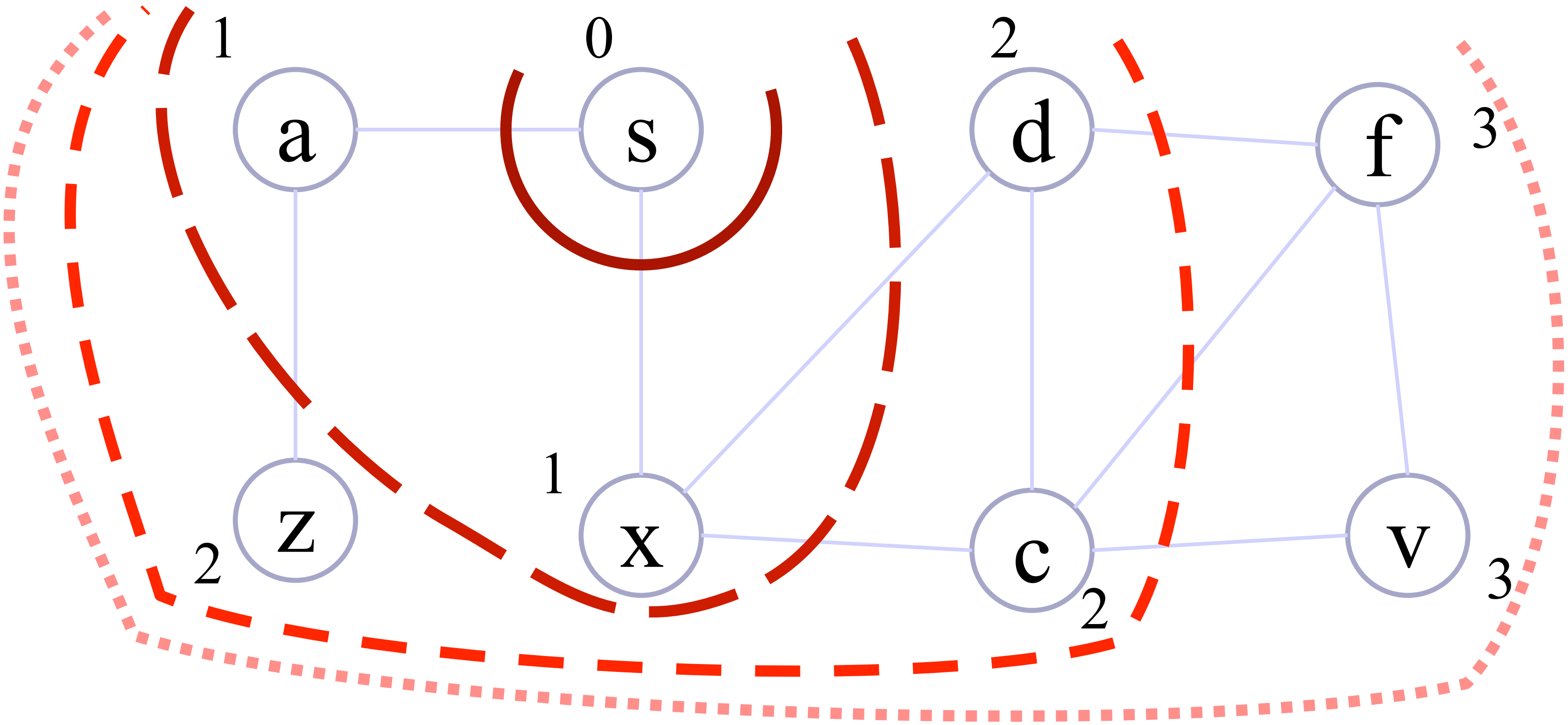
- Problem: Given a digraph $G = (V, E)$ with non-negative edge-weight function $w$, and a node $s$, find $\delta(s, v)$* for all $v$ in $V$
- Want a fast algorithm…
- Question: what if all edge weights are equal to 1 ?



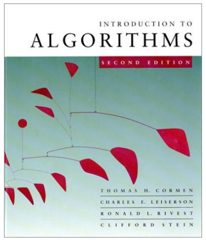
*Paths can be found as well

# BFS

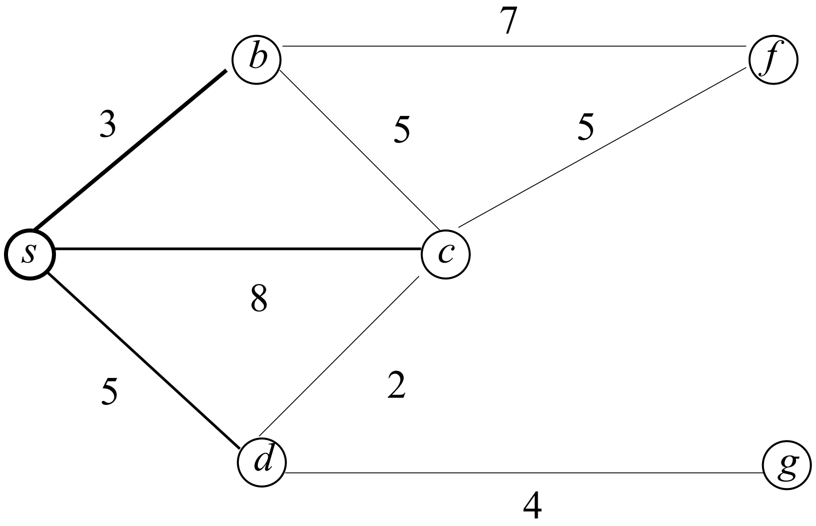




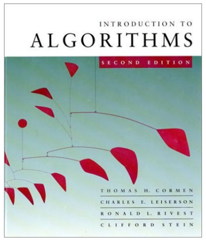
Running time: O(V+E)


INTRODUCTION TO
ALGORITHMS
SECOND EDITION
THOMAS H. CORMEN
CHARLES E. LEISERSON
RONALD L. RIVEST
CLIFFORD STEIN

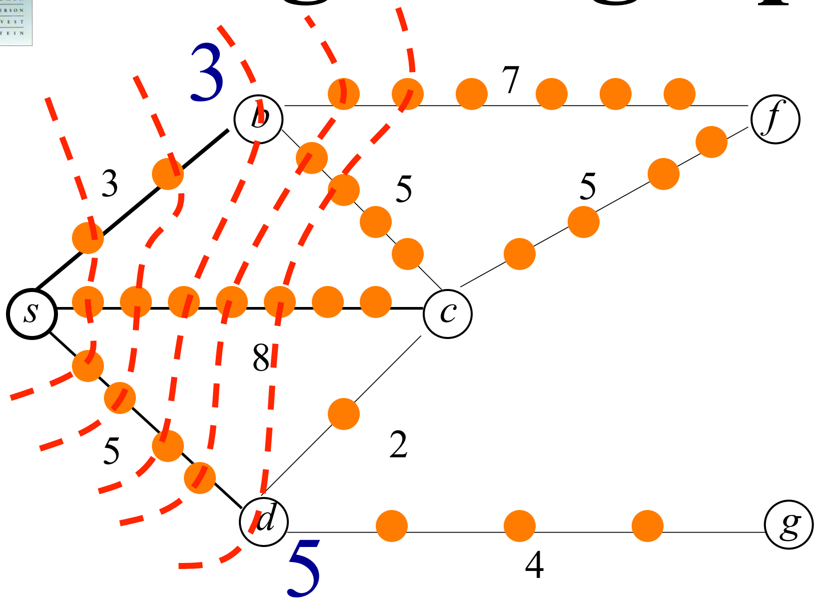
# Weighted graphs



Question II: what if all edge weights are integers in the range 1…W ?



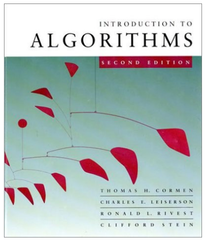
# Weighted graphs



Algorithm:

- Create an unweighted graph by splitting each edge with weight w into w pieces
- Run BFS

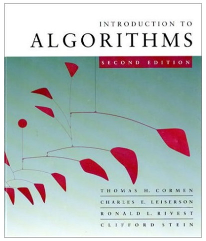
Running time: O(V+WE)



# Greedy approach

**IDEA:** Greedy.

1. Maintain a set $S$ of vertices whose shortest-path distances from $s$ are known.
2. At each step add to $S$ the vertex $v \in V - S$ whose distance estimate from $s$ is minimal.
3. Update the distance estimates of vertices adjacent to $v$.

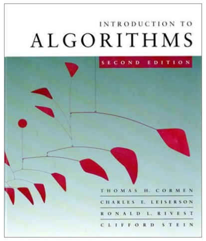


# Dijkstra's algorithm

```
$d[s] \leftarrow 0$
for each $v \in V - \{s\}$
    do $d[v] \leftarrow \infty$
$S \leftarrow \emptyset$
$Q \leftarrow V$            ▹ $Q$ is a priority queue maintaining $V - S$
while $Q \neq \emptyset$
    do $u \leftarrow$ EXTRACT-MIN($Q$)
        $S \leftarrow S \cup \{u\}$
        for each $v \in Adj[u]$
            do if $d[v] > d[u] + w(u, v)$
                then $d[v] \leftarrow d[u] + w(u, v)$
```

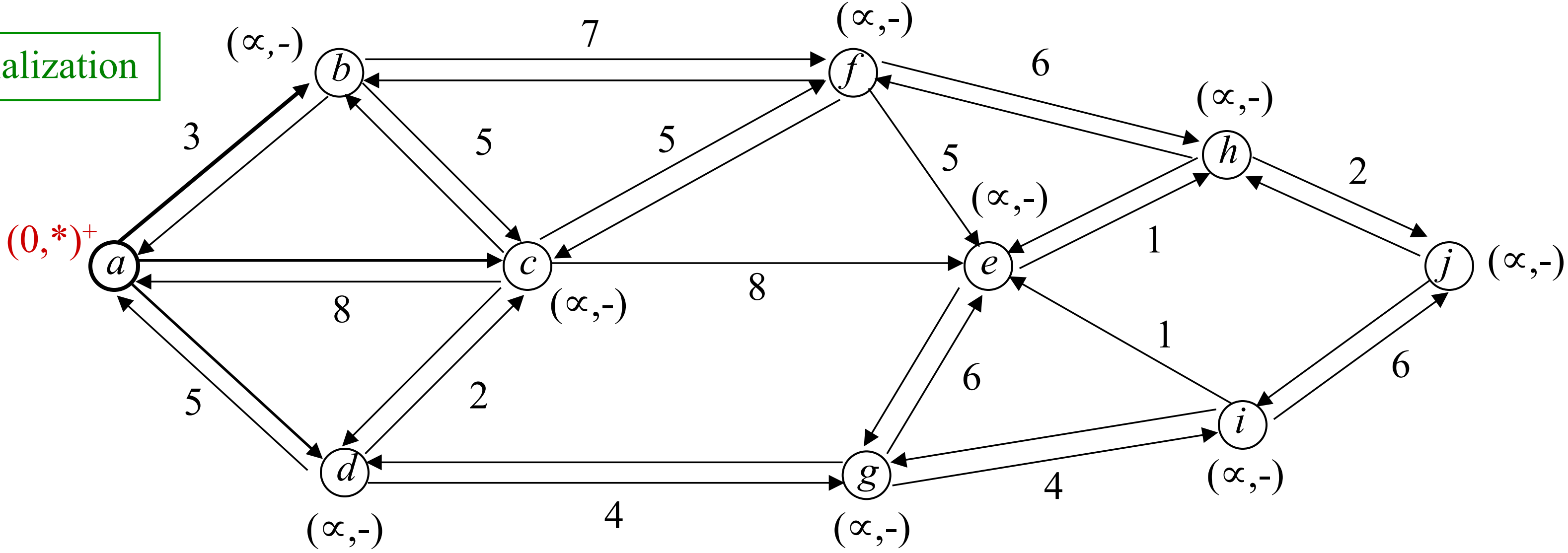
***relaxation step***

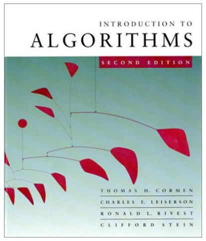
Implicit DECREASE-KEY



# Dijkstra: Example

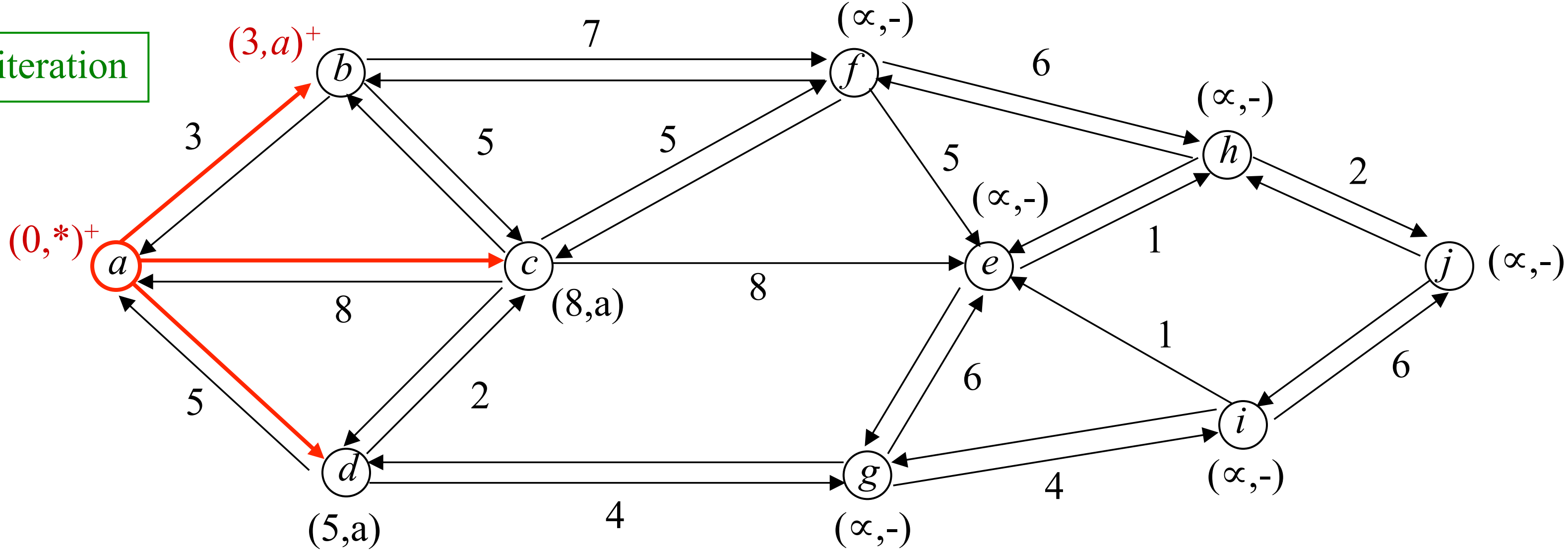
initialization

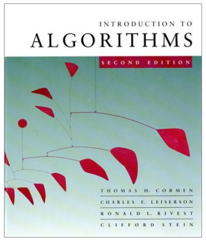




# Dijkstra: Example

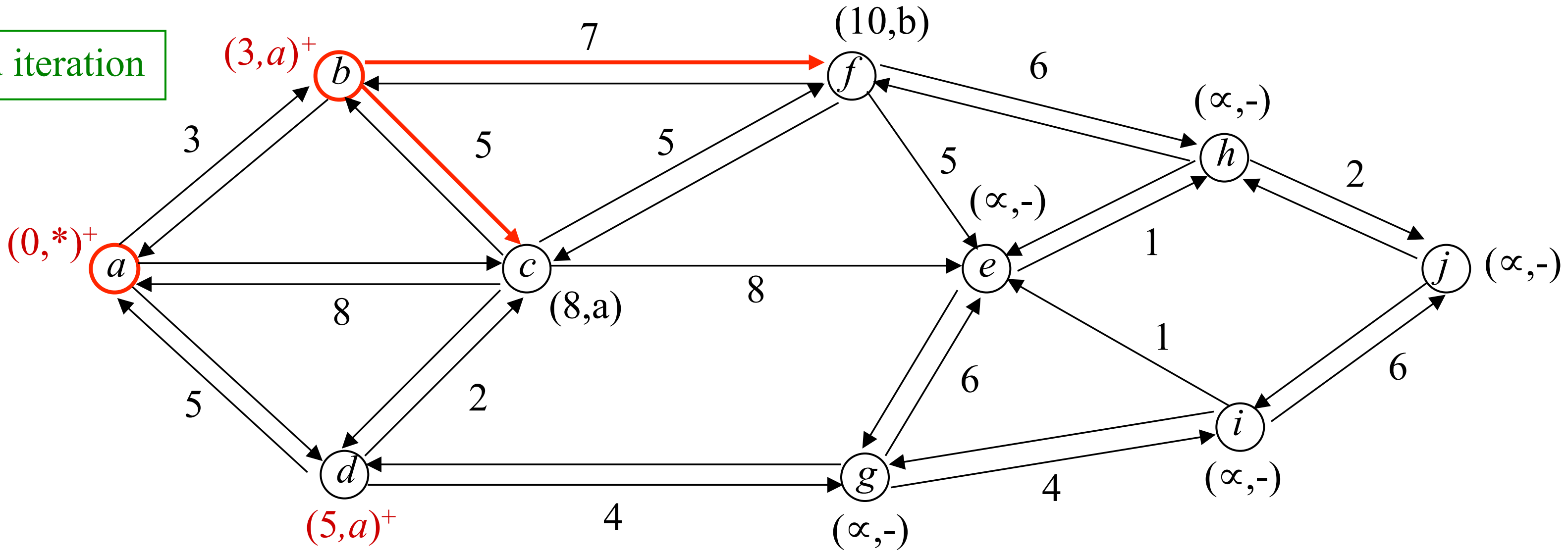
1st iteration

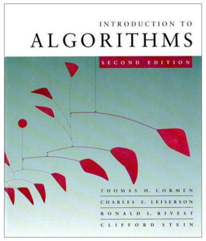




# Dijkstra: Example

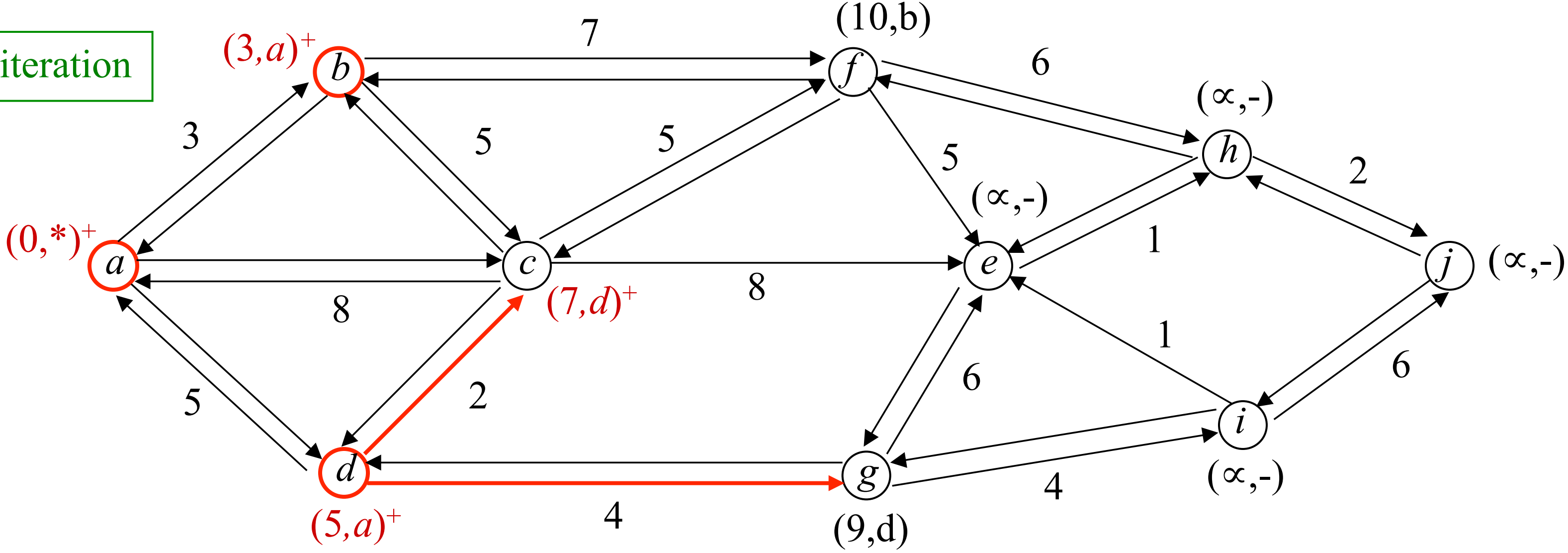
2nd iteration

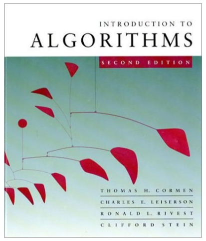




# Dijkstra: Example

3rd iteration

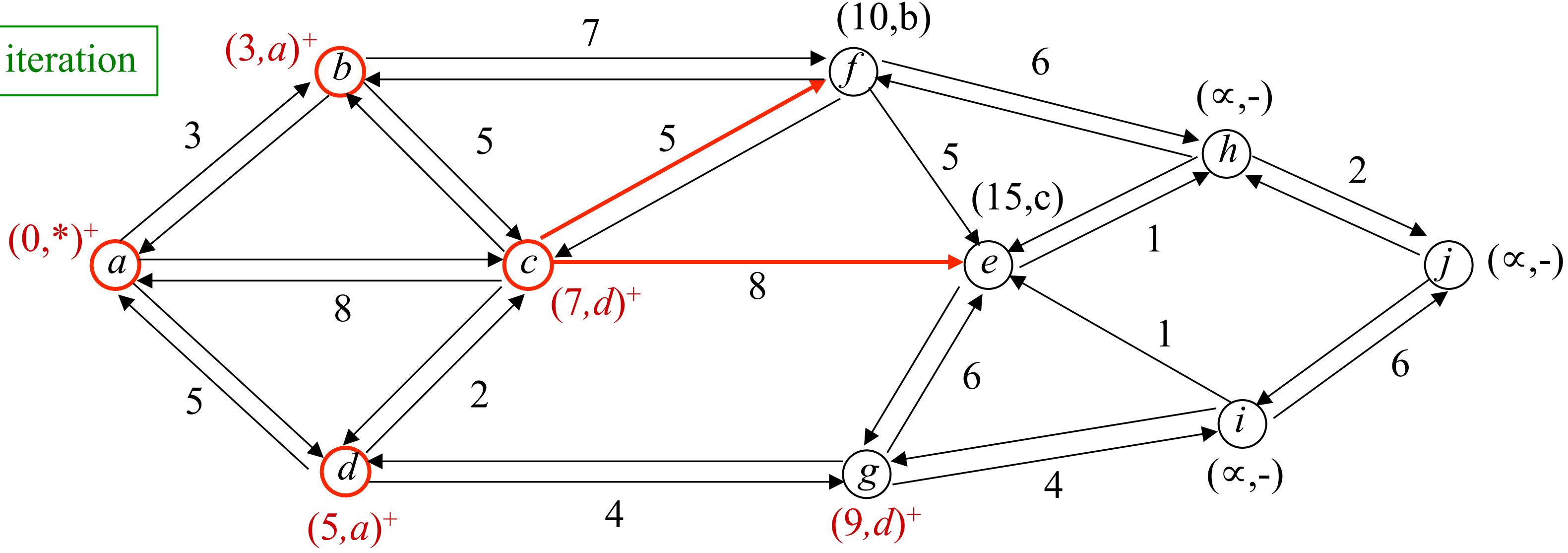


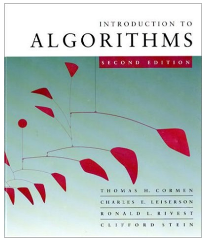


INTRODUCTION TO
ALGORITHMS
SECOND EDITION
THOMAS H. CORMEN
CHARLES E. LEISERSON
RONALD L. RIVEST
CLIFFORD STEIN

# Dijkstra: Example

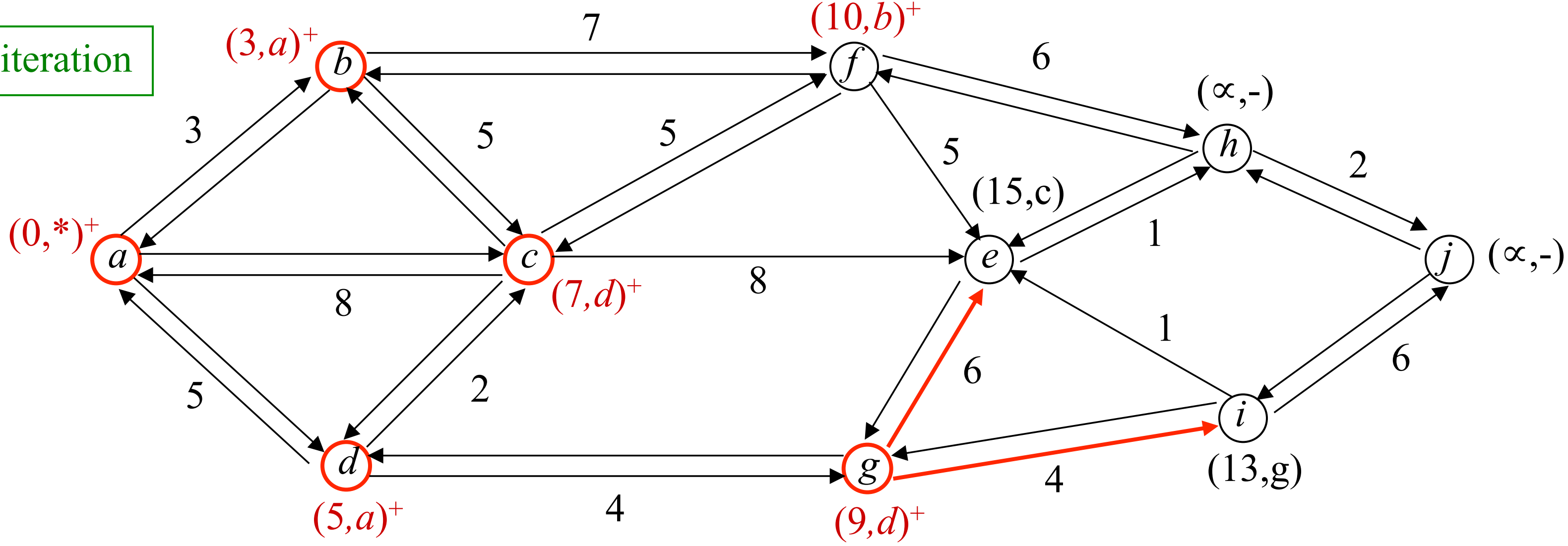
4th iteration

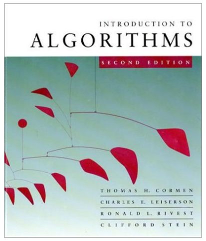




# Dijkstra: Example

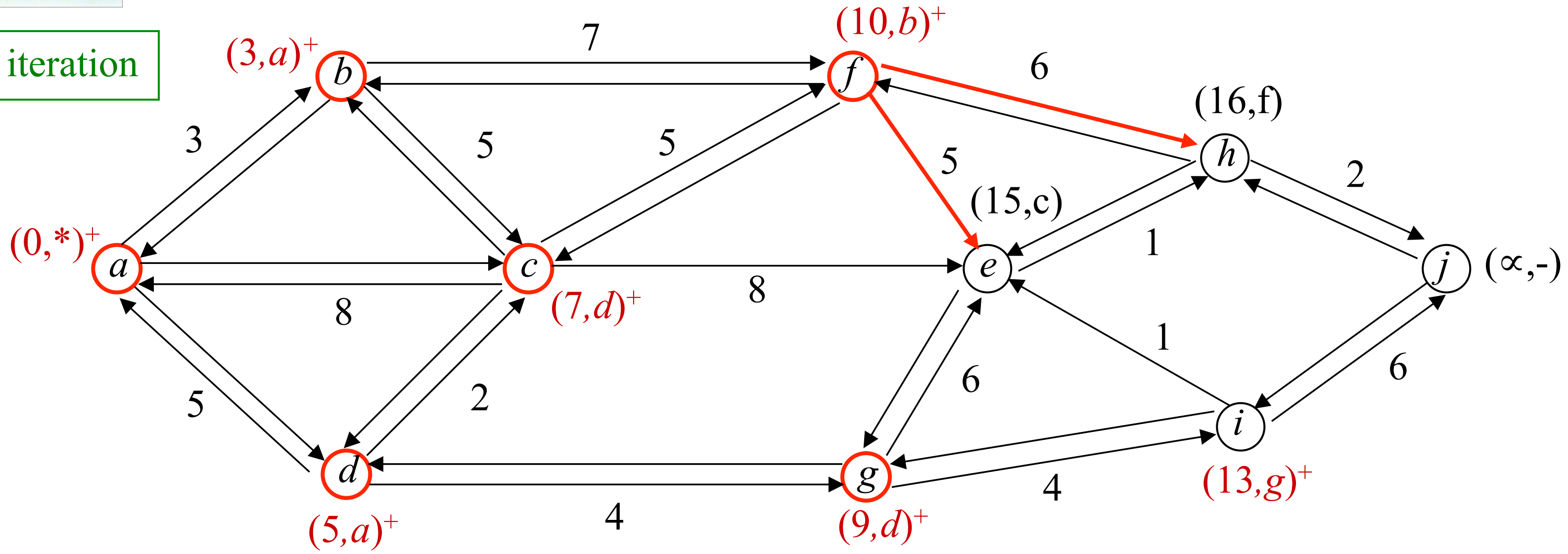
5th iteration

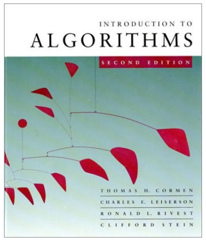




# Dijkstra: Example

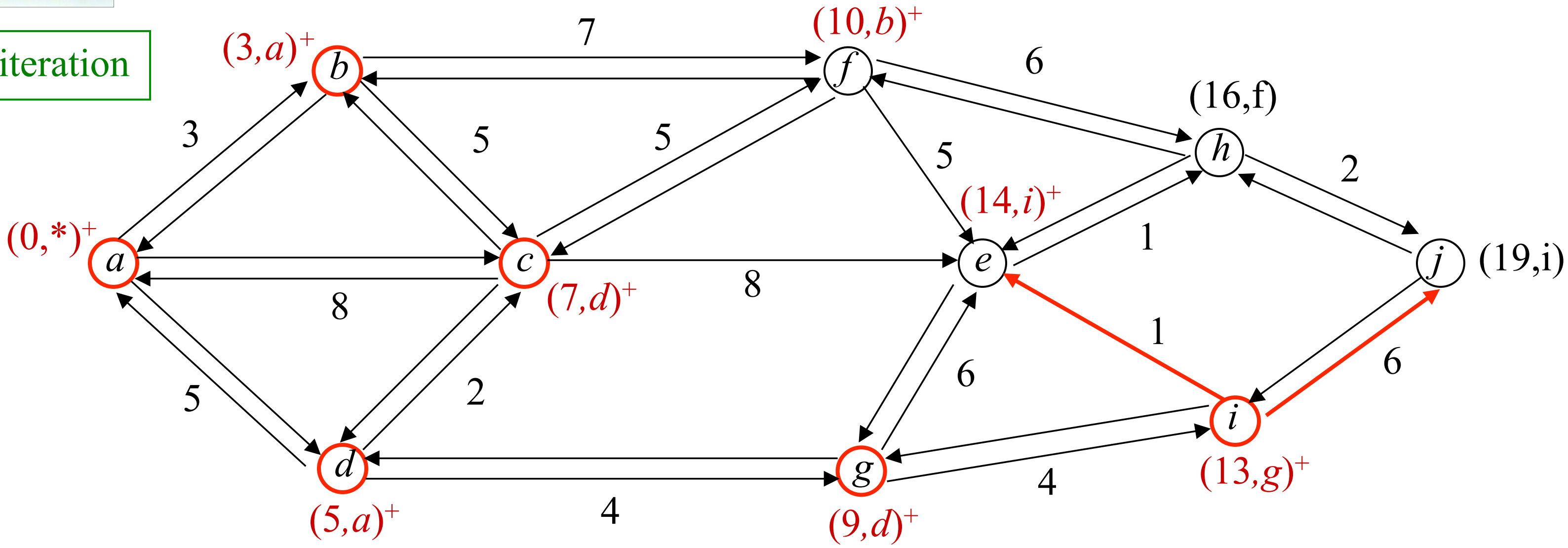
6th iteration

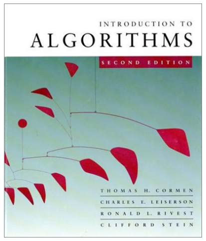




# Dijkstra: Example

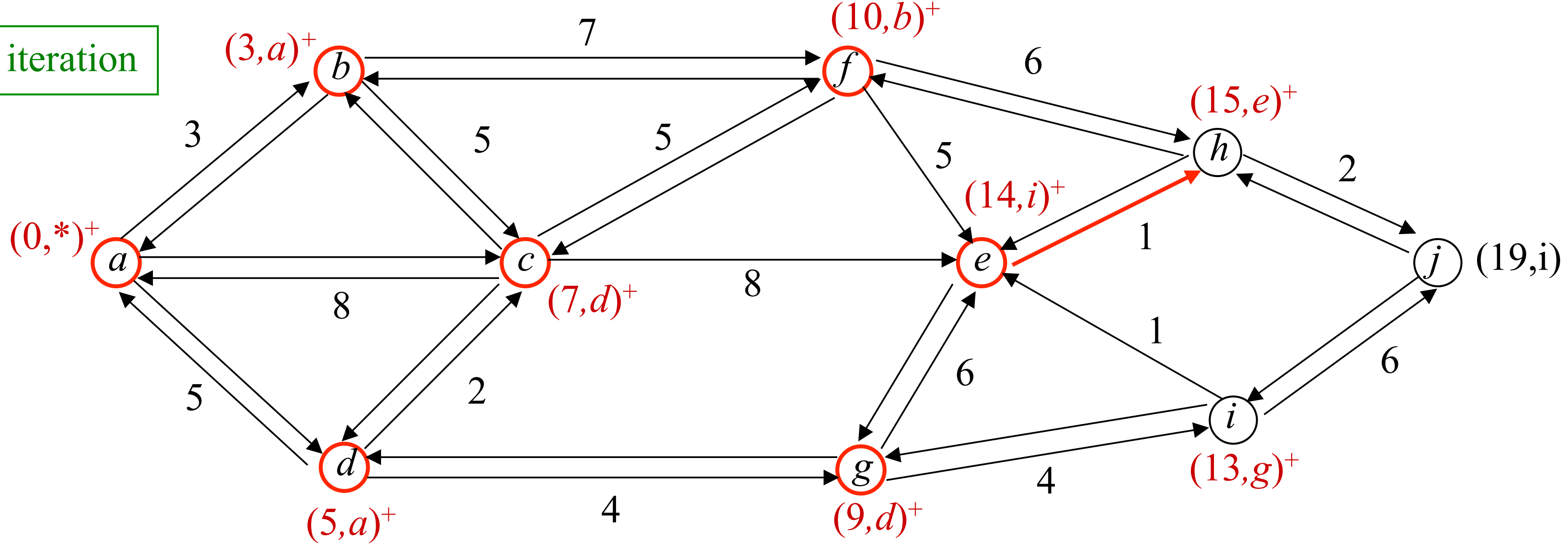
7th iteration

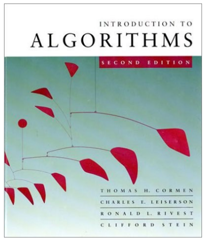




# Dijkstra: Example

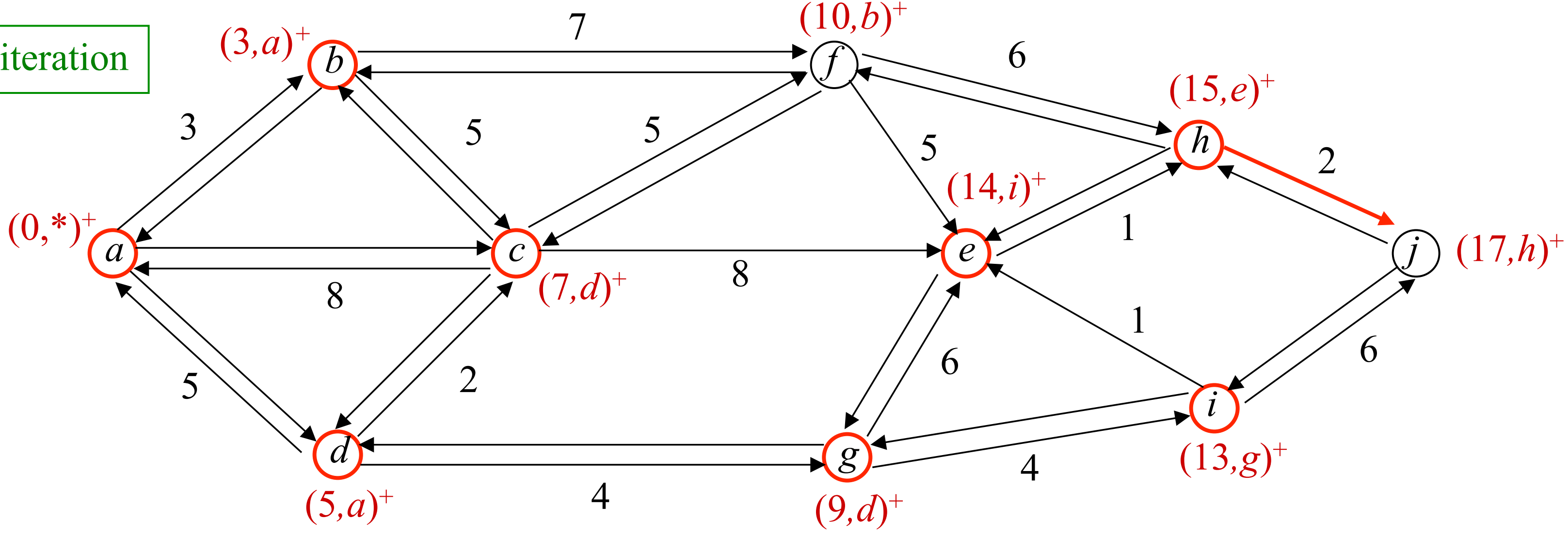
8th iteration



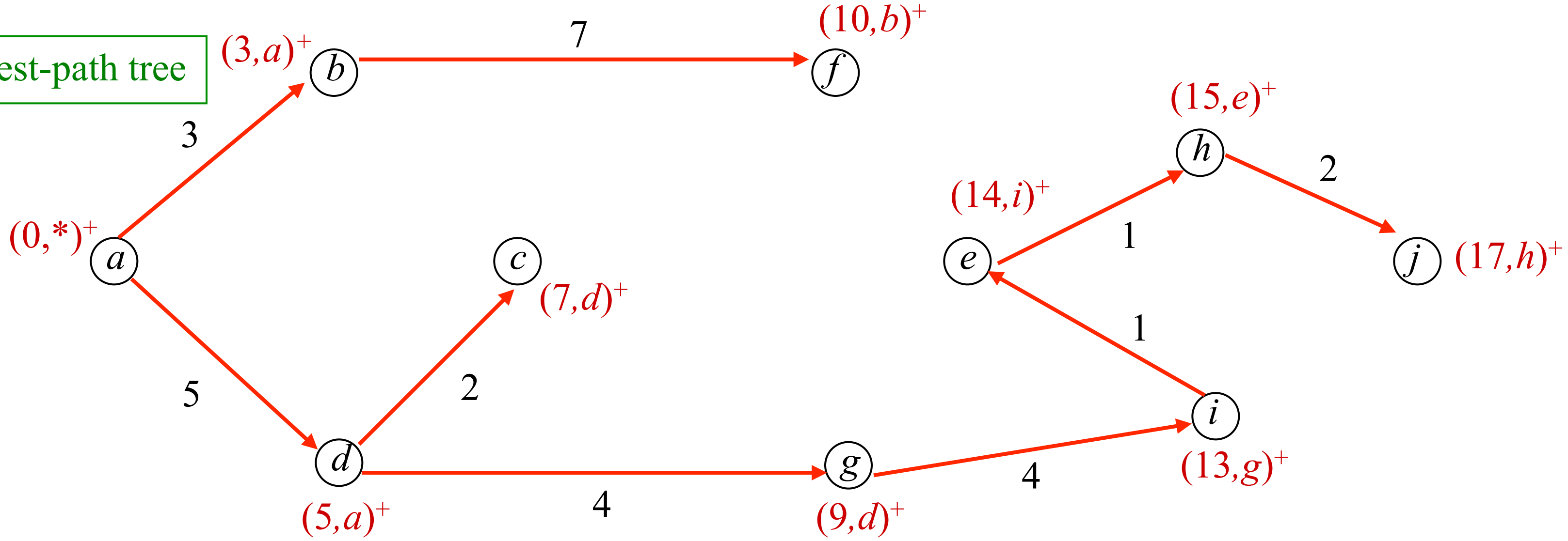


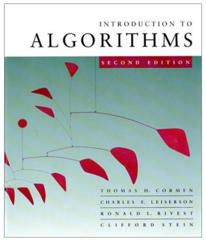
# Dijkstra: Example

9th iteration



Shortest-path tree



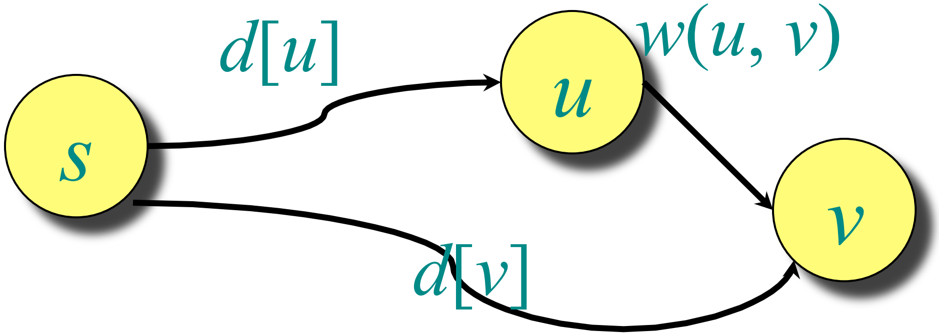


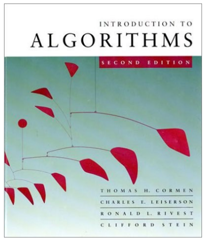
# Correctness — Part I

**Lemma.** Initializing $d[s] \leftarrow 0$ and $d[v] \leftarrow \infty$ for all $v \in V - \{s\}$ establishes $d[v] \geq \delta(s, v)$ for all $v \in V$, and this invariant is maintained over any sequence of relaxation steps.

*Proof.* Recall relaxation step:

**if** $d[v] > d[u] + w(u, v)$ **set** $d[v] \leftarrow d[u] + w(u, v)$



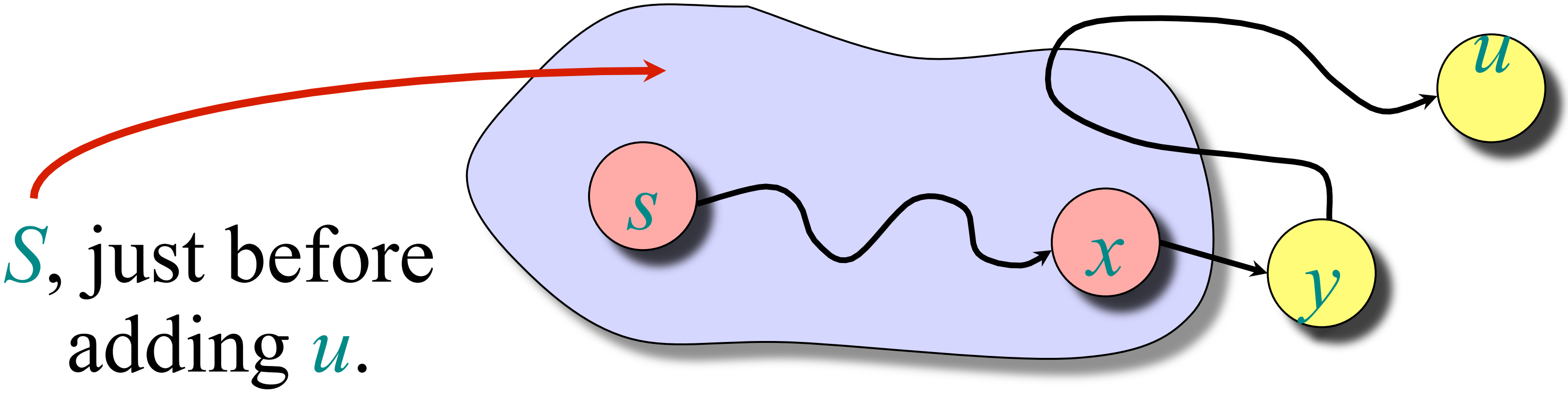


# Correctness — Part II

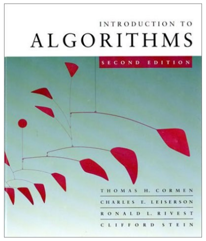
**Theorem.** Dijkstra's algorithm terminates with $d[v] = \delta(s, v)$ for all $v \in V$.

*Proof.*

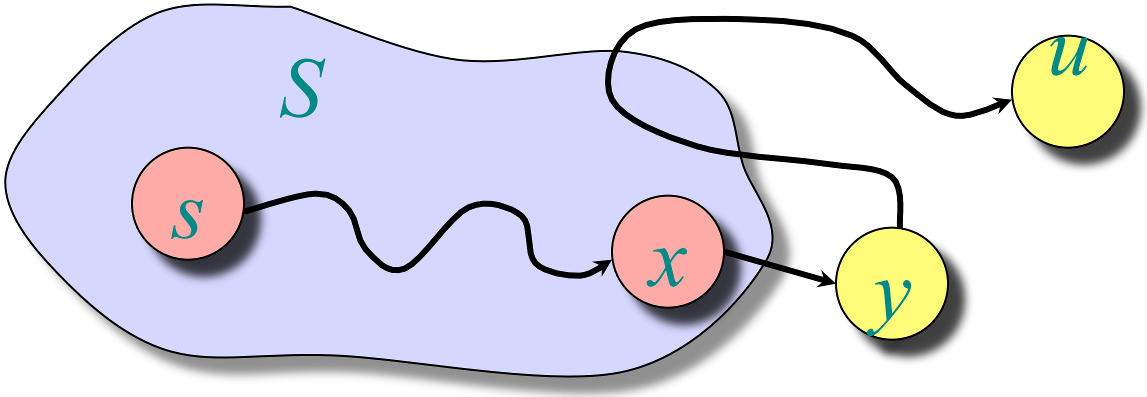
- It suffices to show that $d[v] = \delta(s, v)$ for every $v \in V$ when $v$ is added to $S$
- Suppose $u$ is the first vertex added to $S$ for which $d[u] \neq \delta(s, u)$ . Let $y$ be the first vertex in $V - S$ along a shortest path from $s$ to $u$, and let $x$ be its predecessor:



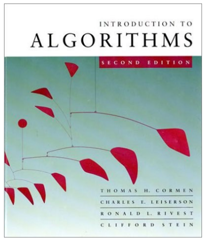
$S$, just before adding $u$.



# Correctness — Part II (continued)



- Since $u$ is the first vertex violating the claimed invariant, we have $d[x] = \delta(s, x)$
- Since subpaths of shortest paths are shortest paths, it follows that $d[y]$ was set to $\delta(s, x) + w(x, y) = \delta(s, y)$ just after $x$ was added to $S$
- Consequently, we have $d[y] = \delta(s, y) \leq \delta(s, u) \leq d[u]$
- But, $d[y] \geq d[u]$ since the algorithm chose $u$ first
- Hence $d[y] = \delta(s, y) = \delta(s, u) = d[u]$ - contradiction



# Analysis of Dijkstra

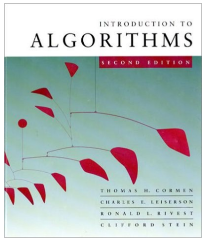
```
while $Q \neq \emptyset$
    do $u \leftarrow$ EXTRACT-MIN$(Q)$
       $S \leftarrow S \cup \{u\}$
       for each $v \in Adj[u]$
           do if $d[v] > d[u] + w(u, v)$
                 then $d[v] \leftarrow d[u] + w(u, v)$
```

$|V|$ times

$degree(u)$ times

DECREASE-KEY

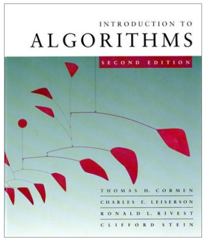
$$\text{Time} = \Theta(V) \cdot T_{\text{EXTRACT-MIN}} + \Theta(E) \cdot T_{\text{DECREASE-KEY}}$$



# Analysis of Dijkstra (continued)

Time = $\Theta(V) \cdot T_{\text{EXTRACT-MIN}} + \Theta(E) \cdot T_{\text{DECREASE-KEY}}$

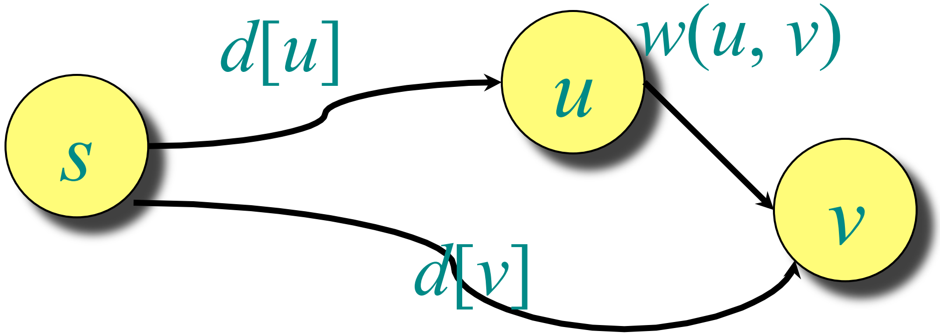
| $Q$ | $T_{\text{EXTRACT-MIN}}$ | $T_{\text{DECREASE-KEY}}$ | Total |
|---|---|---|---|
| array | $O(V)$ | $O(1)$ | $O(V^2)$ |
| binary heap | $O(\lg V)$ | $O(\lg V)$ | $O(E \lg V)$ |
| Fibonacci heap | $O(\lg V)$ amortized | $O(1)$ amortized | $O(E + V \lg V)$ worst case |



# Tuesday: generic algorithm

```
$d[s] \leftarrow 0$
for each $v \in V - \{s\}$                         } initialization
    do $d[v] \leftarrow \infty$

while there is an edge $(u, v) \in E$ s. t.
        $d[v] > d[u] + w(u, v)$ do                 } relaxation
  select one such edge "somehow"                   } step
  set $d[v] \leftarrow d[u] + w(u, v)$
endwhile
```



How to do it in O( (V+E)logV ) time ?