# Network Basics 2

Advanced Social Computing

Department of Computer Science
University of Massachusetts, Lowell
Spring 2020

Hadi Amiri
hadi@cs.uml.edu

# Lecture Topics

- Connected Components
- Breadth-First Search
- Depth-First Search
- Shortest Path Algorithm
  - Dijkstra's algorithm

# Connected Components

- Connected component of a graph is a subset of nodes such that:
  - every node in the subset has a path to every other; and
  - the subset is not part of a bigger component.

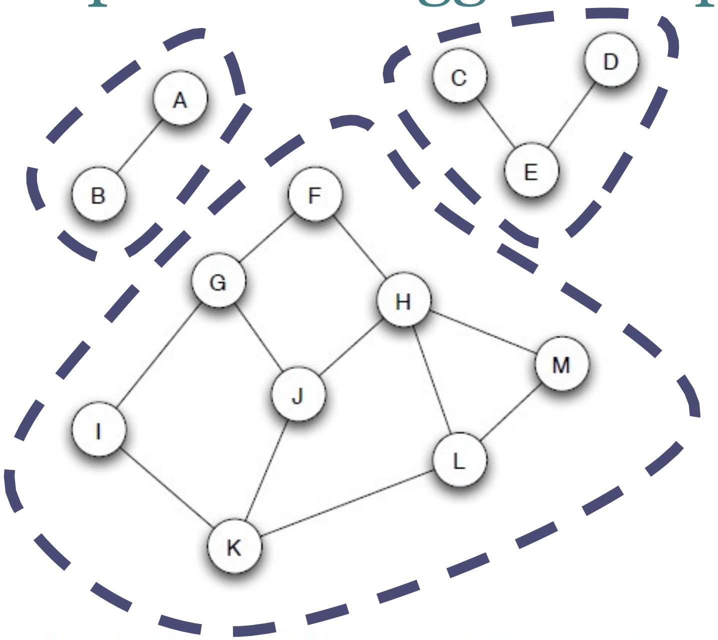



Figure 2.5: A graph with three connected components.

# Connected Components

- Connected component of a graph is a subset of nodes such that:
  - every node in the subset has a path to every other; and
  - the subset is not part of a bigger component.

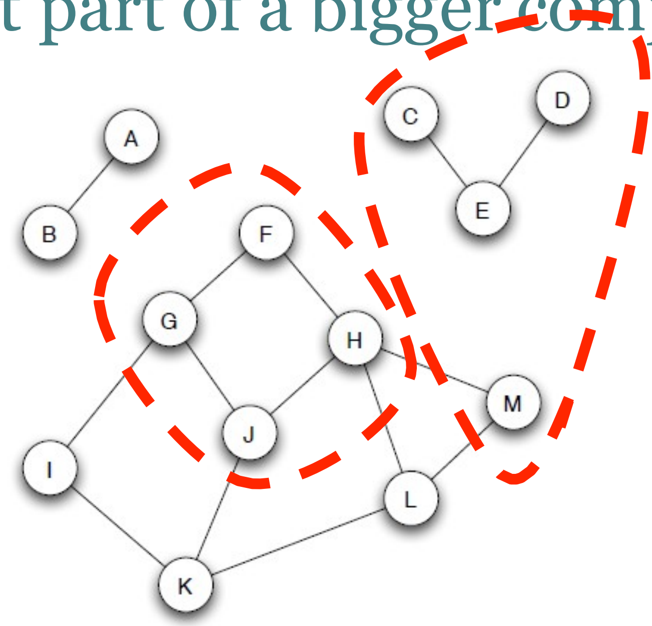
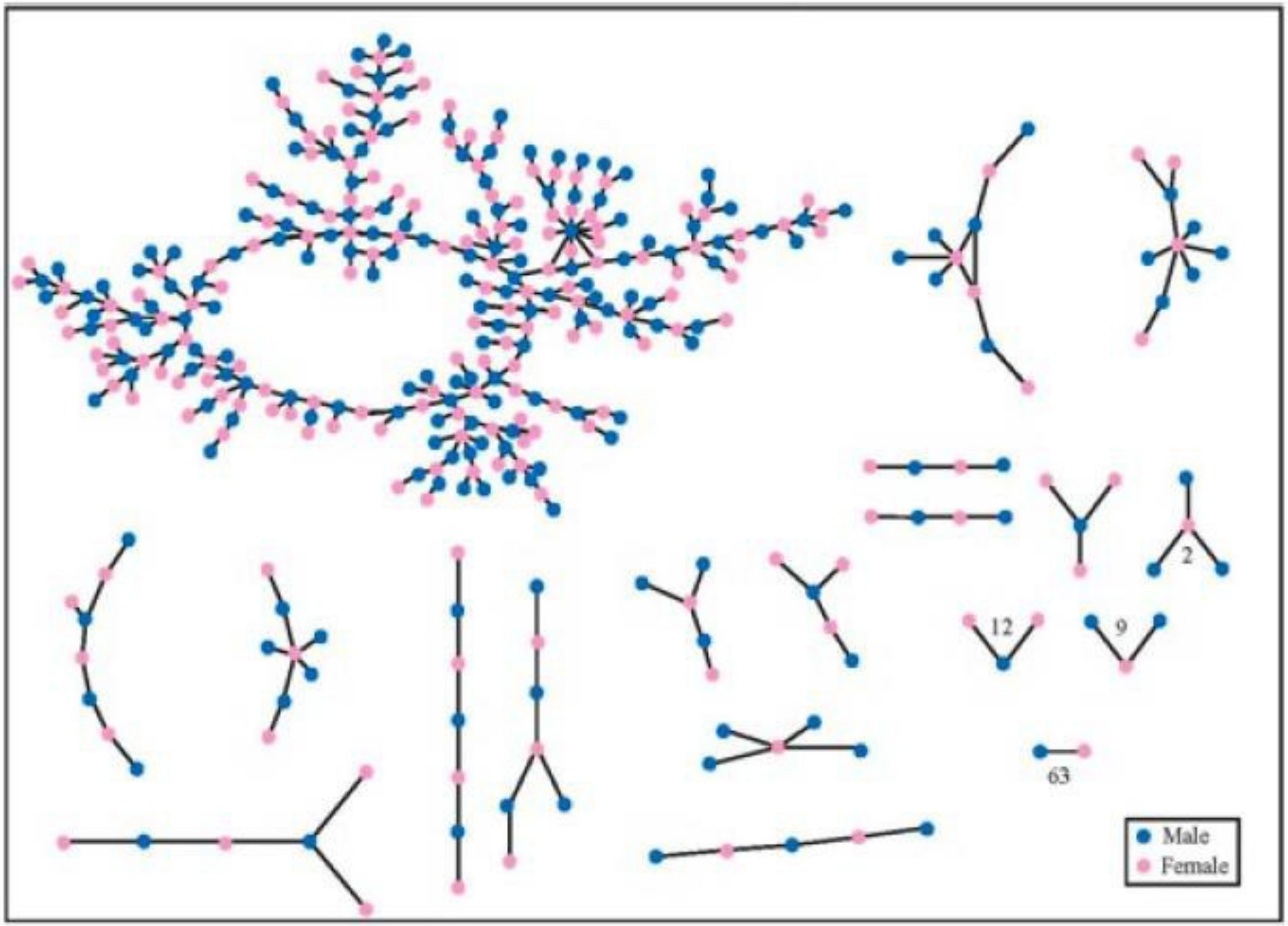



Figure 2.5: A graph with three connected components.

# Connected Components- Cnt.



# Connected Components- Cnt.

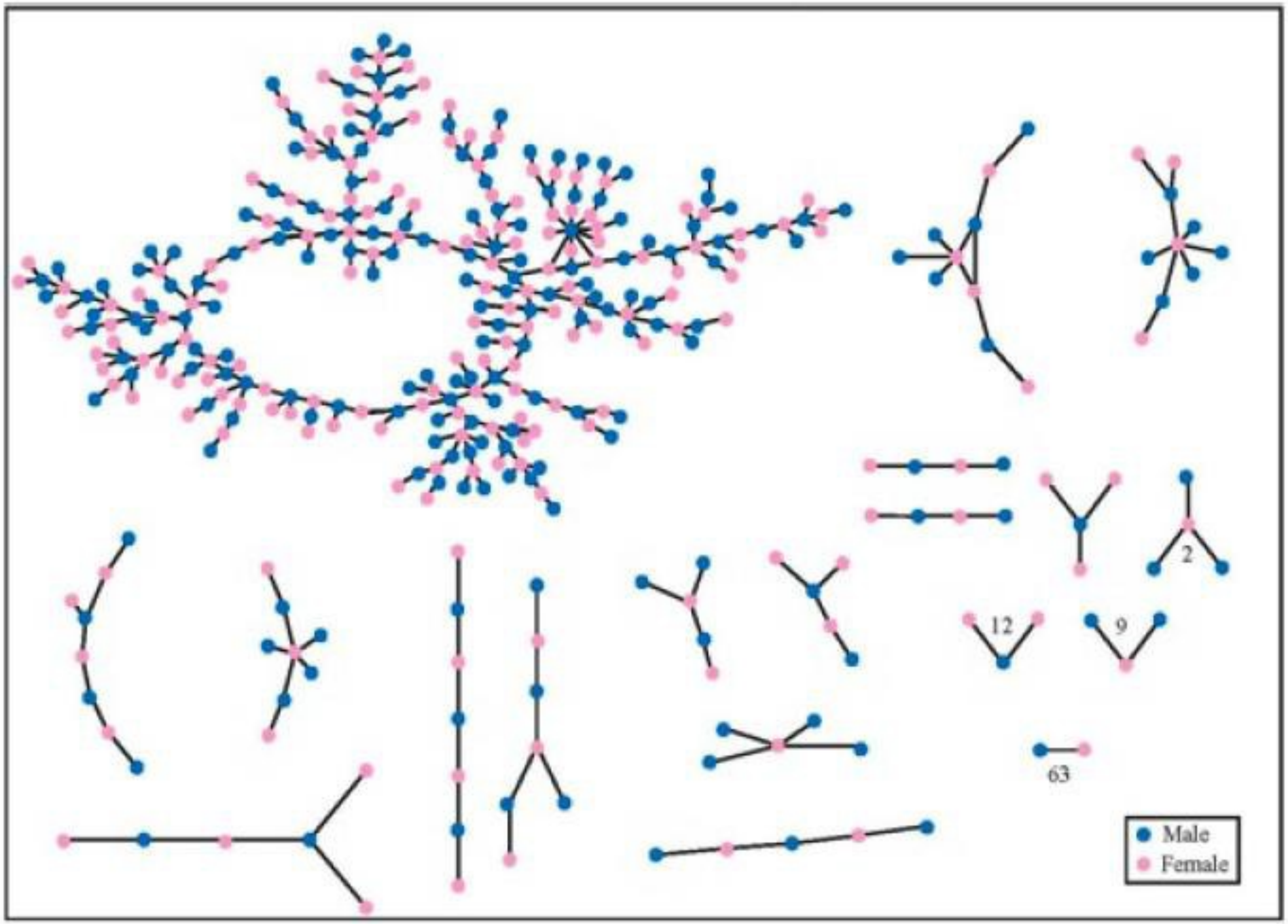



Figure 2.7: A network in which the nodes are students in a large American high school, and an edge joins two who had a romantic relationship at some point during the 18-month period in which the study was conducted [49].

# Breadth- Depth-First Search

- General techniques for traversing graphs!
  - Start from a given node *s* (i.e. start node) and visit all nodes and edges in the graph.
- Compute the connected components of graph!
  - Use components to determine whether graph is connected!
    - How?
  - Use components to determine if there is a path btw node pairs!
    - How?

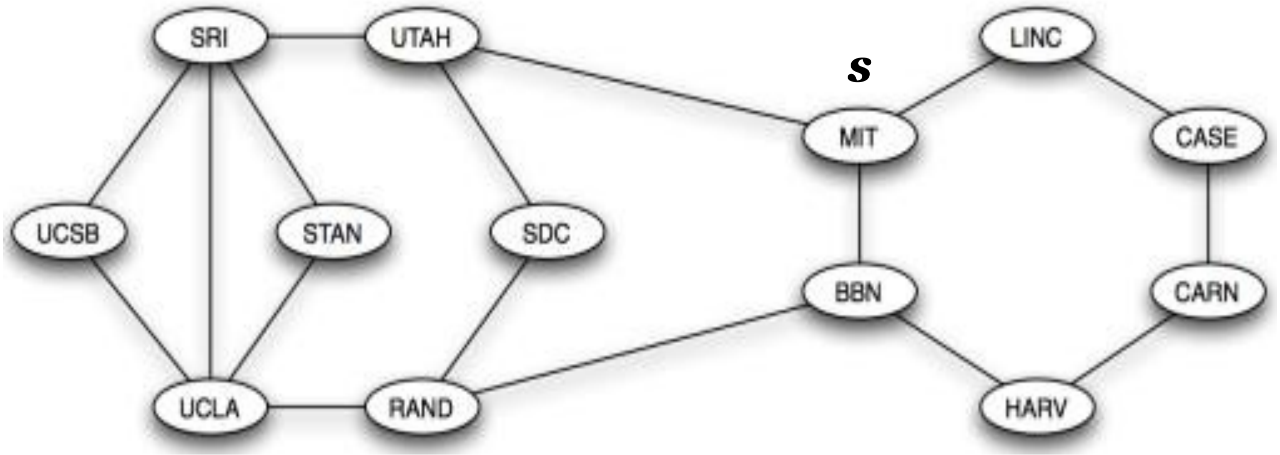
# Breadth-First Search

- Start with *s*
- Visit all neighbors of *s*
  - these are called **level-1 nodes**
- Visit all neighbors of level-1 nodes
  - these are called **level-2 nodes**
- Repeat until all nodes are visited.
  - Each Node is only visited once.

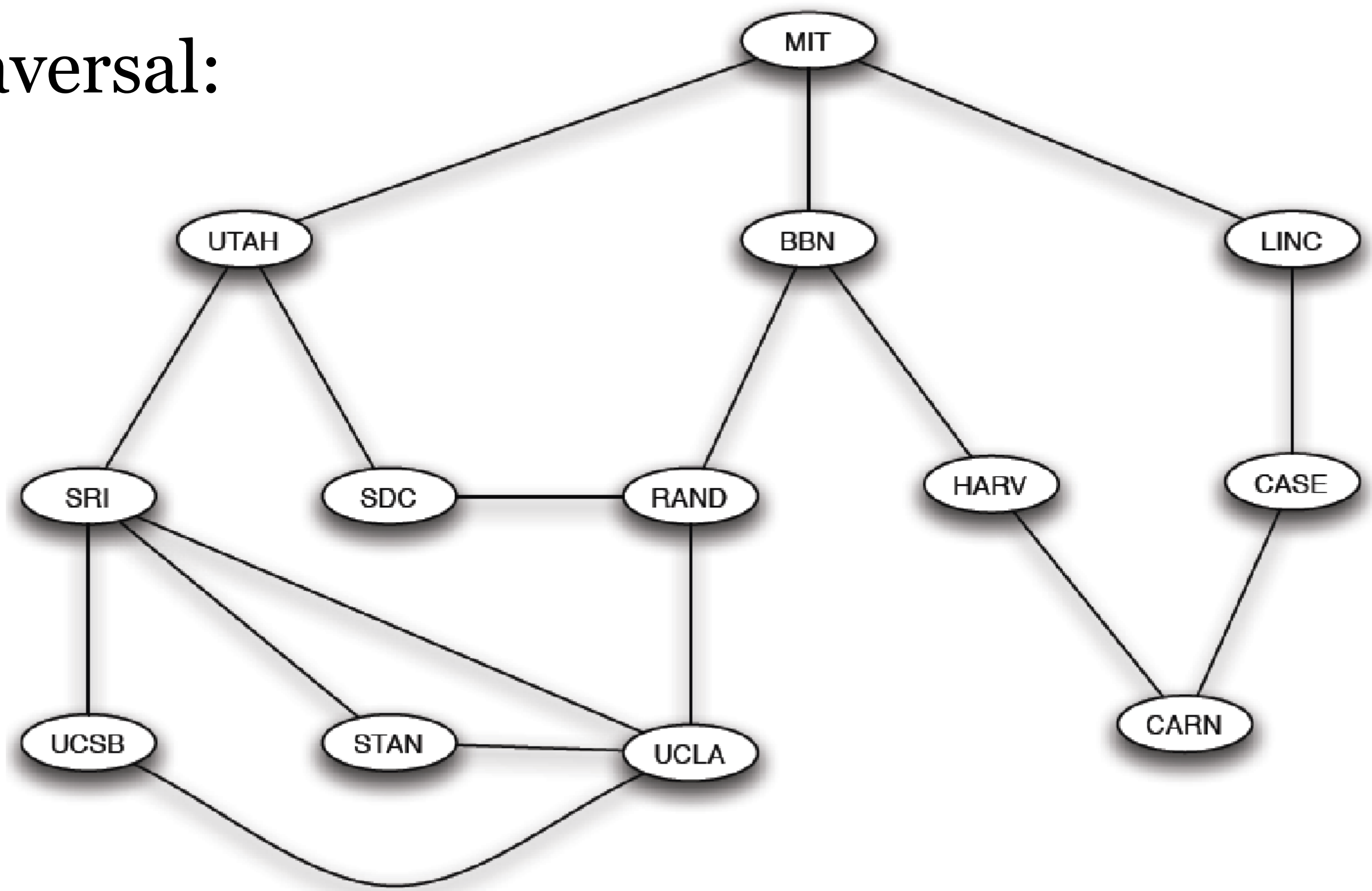
- Key Point:
  - All level-k nodes should be visited before any level-(k+1) node!

# Example 1.

- Graph G:

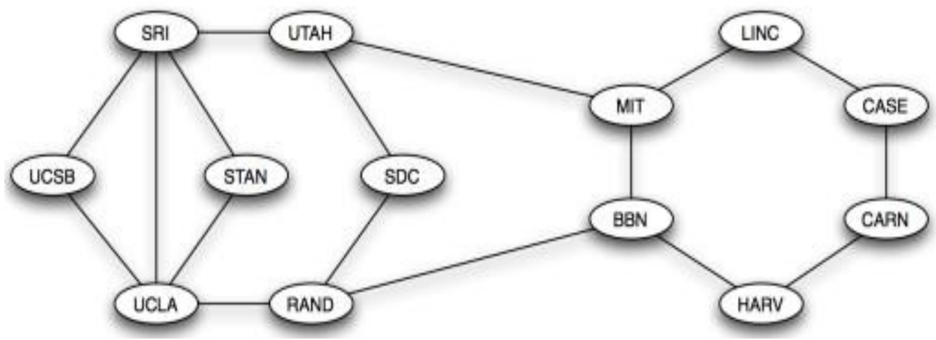


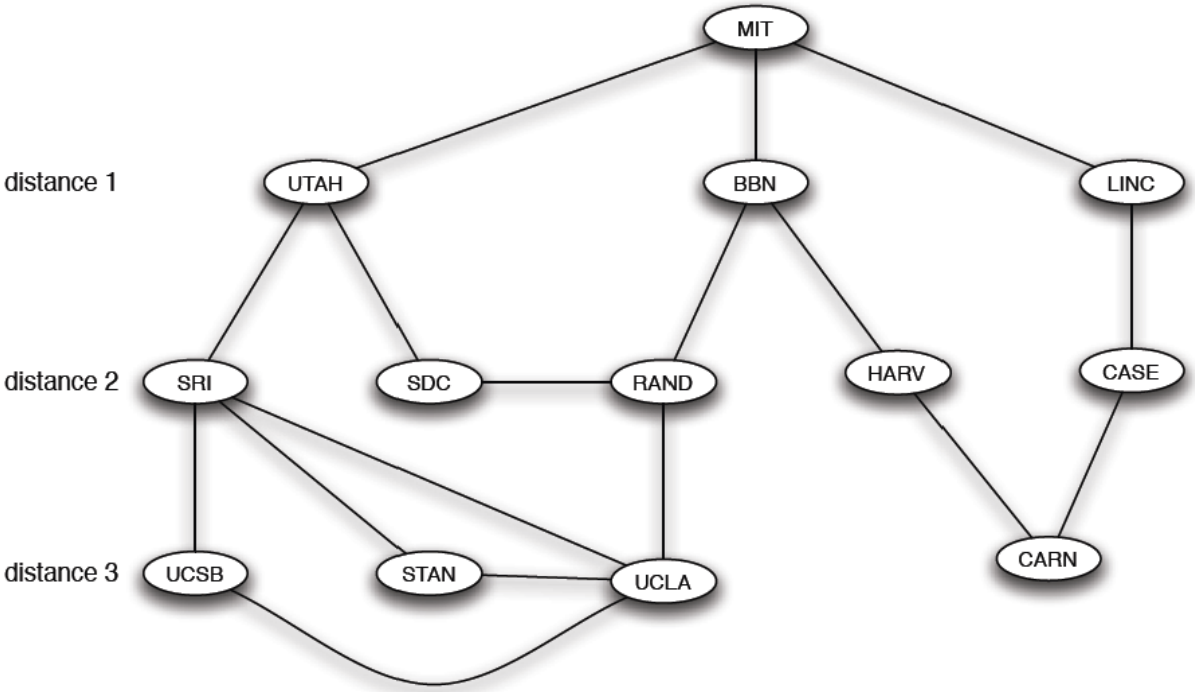

- Its BFS traversal:



# Example 1. BFS -Cnt.

- BFS traversal:
  - Distance to root at level-*i*?
  - Components?
    - Connectivity?
    - Paths?



# Depth-First Search

- Starts from s
- Explores as far as possible along each branch before backtracking.
  - Visit a neighbor of *s* [say $v_1$]
  - Visit a neighbor of $v_1$ [say $v_2$]
  - Repeat until all nodes are visited.

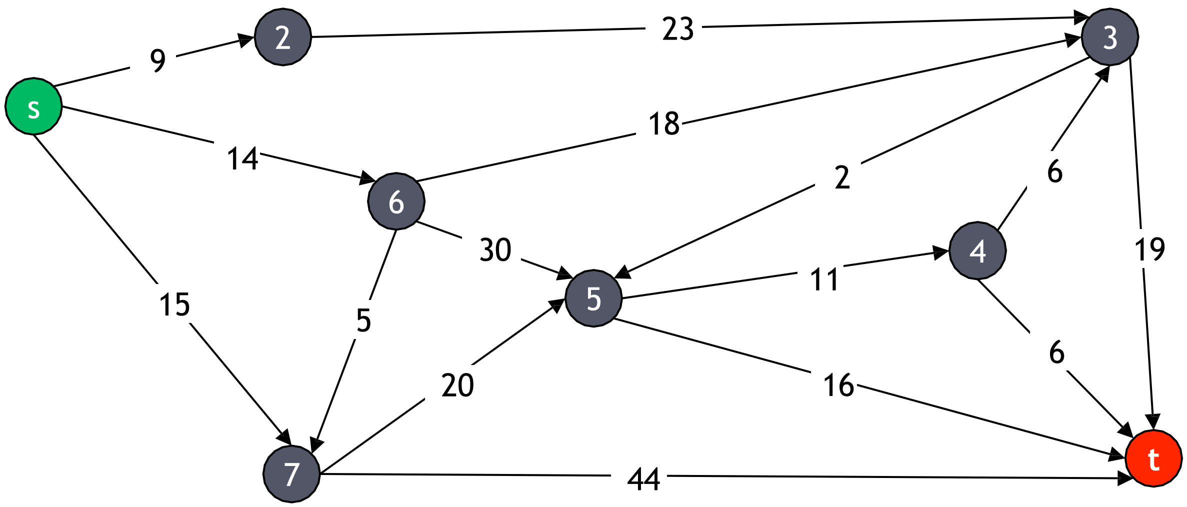
# Shortest Path Algorithms

- Given a weighted directed graph and two nodes *s* and *t*, find the shortest path from *s* to *t*.
  - Cost of path = sum of edge weights in path

# Shortest Path Algorithms- Cnt.

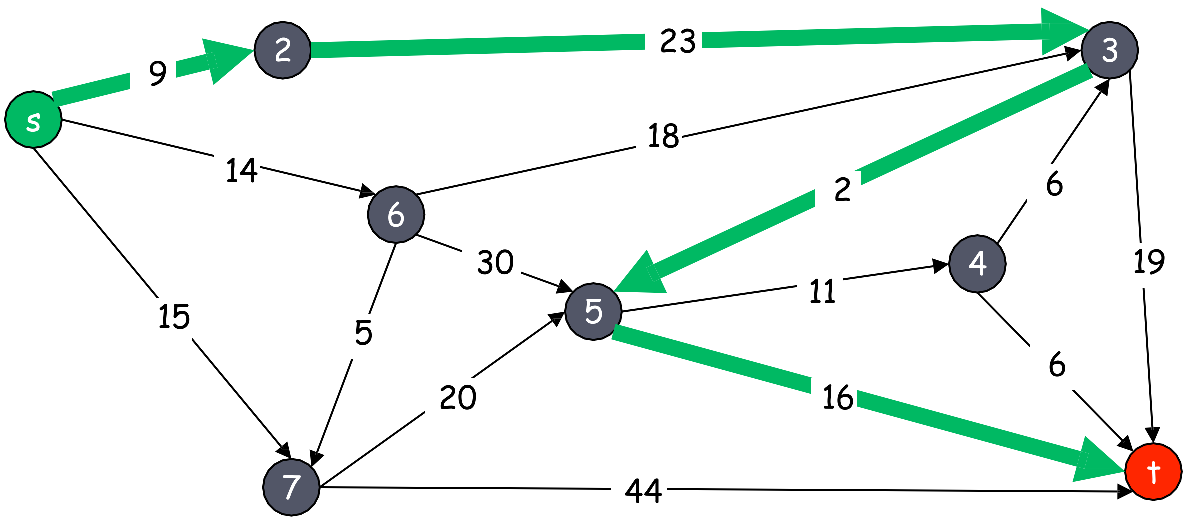
- Dijkstra’s algorithm
- The Bellman-Ford algorithm
- The Floyd-Warshall algorithm
- Johnson's algorithm
- Etc.

# Shortest Path Algorithms- Cnt.




- Shortest path from *s* to *t*?

# Shortest Path Algorithms- Cnt.



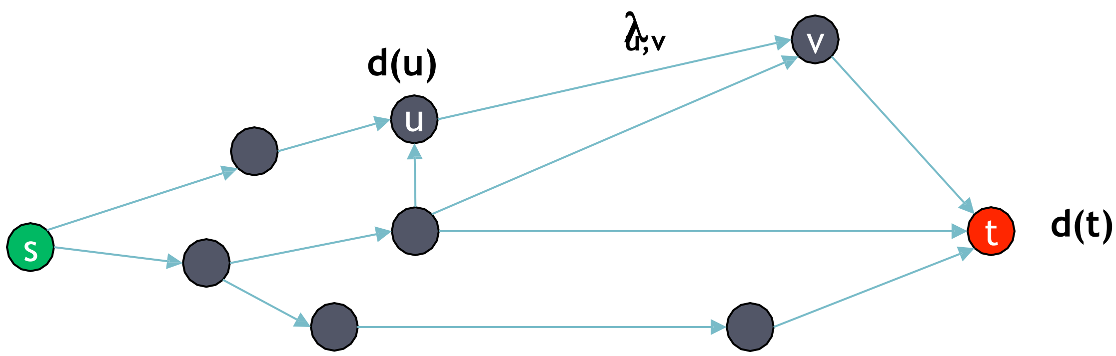

- Shortest Path= s-2-3-5-t
- Cost of path = 9 + 23 + 2 + 16 = 48.

# Shortest Path Algorithms- Cnt.

- Applications
  - Small World Phenomenon
  - Internet packet routing
  - Flight reservations
  - Driving directions
  - …

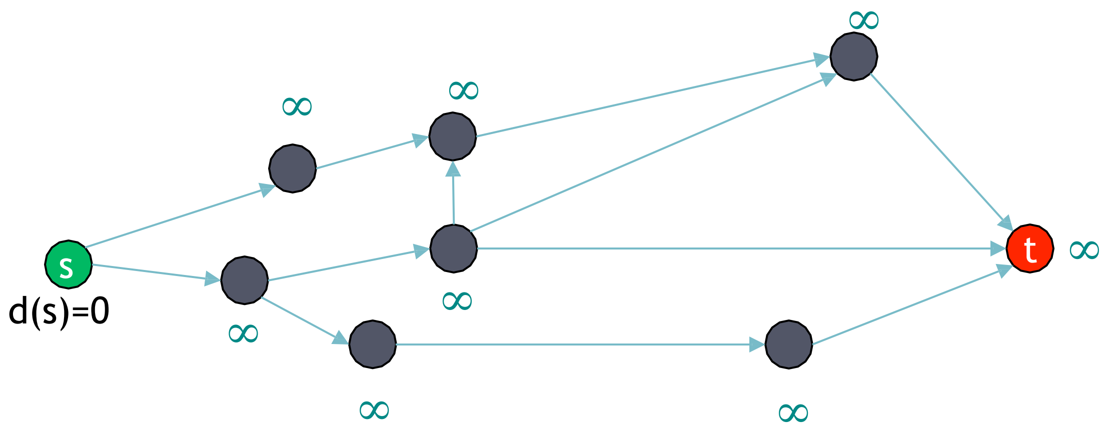
# Dijkstra algorithm

- Weighted Directed graph G = (**N**, **E**),
  - $s$: source node
  - $t$: target node
  - $l_{(u,v)}$: weight of the edge btw nodes $u$ and $v$
  - $d$(u): shortest path distance from s to u.
    - sum of edge weights in path
- We aim to compute d($t$)!



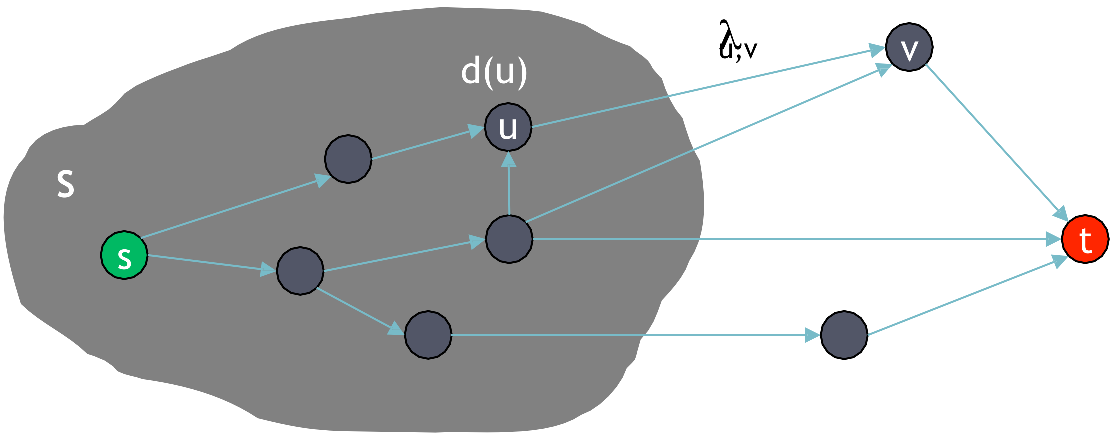
# Dijkstra algorithm- Cnt.

- Initialization?
  - d(s) = 0
  - d(u)= ∞ for all other nodes



# Dijkstra algorithm- Cnt.

- To find the shortest path from *s* to *t*:
  - Maintain a set of ***explored nodes*** **S** for which we have determined the shortest path distance from s to any u ϵ **S**.
  - Repeatedly expand **S**.

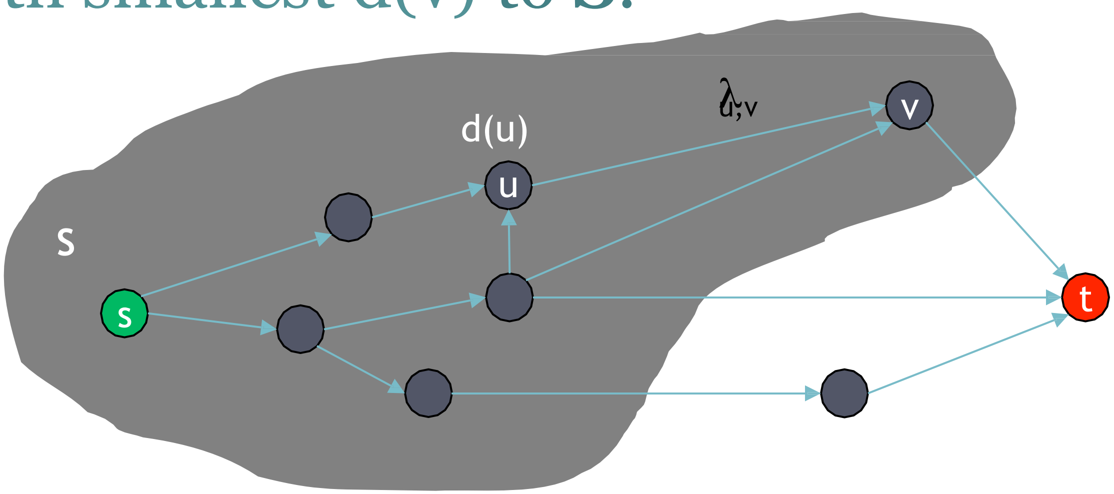


# Dijkstra algorithm- Cnt.

- Repeatedly expand **S**?
  - Repeatedly update *d(.)* for the unexplored nodes:

$$\textbf{if } d(v) > d(u) + l_{(u,v)}$$
$$\textbf{then } d(v) \leftarrow d(u) + l_{(u,v)}$$

  - add $v$ with smallest d(v) to **S**.



# Dijkstra algorithm- Cnt.

- $d(s) \leftarrow 0$
- **for** each $v \in N - \{s\}$
  - **do** $d(v) \leftarrow \infty$
- $S \leftarrow \emptyset$
- $Q \leftarrow N$ ▷ $Q$ is a set maintaining $N - S$
- **while** $Q \neq \emptyset$
  - **do** $u \leftarrow$ EXTRACT-MIN($Q$)
    - $S \leftarrow S \cup \{u\}$
    - **for** each $v \in Adj(u)$
      - **do if** $d(v) > d(u) + l_{(u,v)}$
        - **then** $d(v) \leftarrow d(u) + l_{(u,v)}$

Set of explored nodes

Set of unexplored nodes

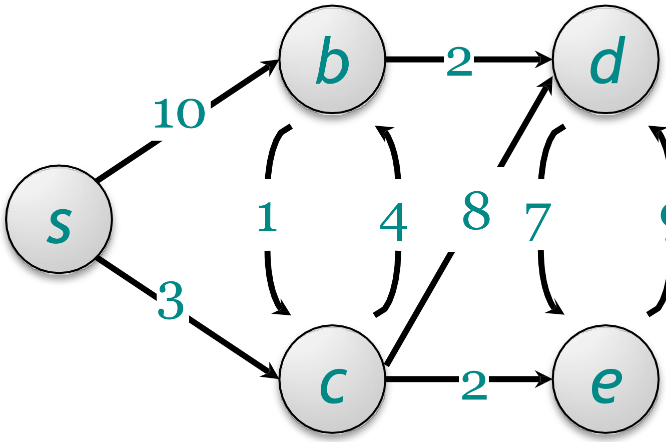
Returns node u ∈ Q that has minimum d(u)

Add it to explored nodes

Update d(.) for all neighbors of u: this is called **relaxation**!

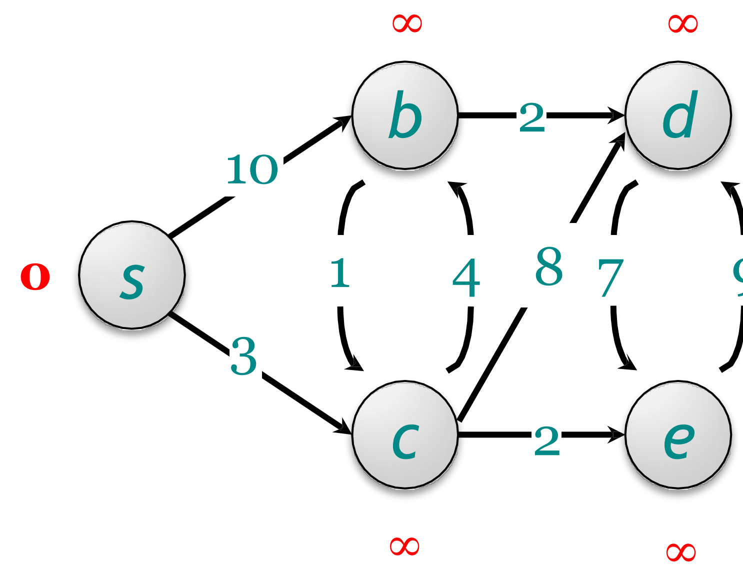
# Example 1.

- $d(s) \leftarrow 0$
- **for** each $v \in \boldsymbol{N} - \{s\}$
  - **do** $d(v) \leftarrow \infty$
- $\boldsymbol{S} \leftarrow \emptyset$
- $\boldsymbol{Q} \leftarrow \boldsymbol{N}$
- **while** $\boldsymbol{Q} \neq \emptyset$
  - **do** $u \leftarrow$ EXTRACT-MIN($\boldsymbol{Q}$)
    - $\boldsymbol{S} \leftarrow \boldsymbol{S} \cup \{u\}$
    - **for** each $v \in Adj(u)$
      - **do if** $d(v) > d(u) + l_{(u, v)}$
        - **then** $d(v) \leftarrow d(u) + l_{(u,v)}$



# Example 1.

- $d(s) \leftarrow 0$
- **for each** $v \in \boldsymbol{N} - \{s\}$
  - **do** $d(v) \leftarrow \infty$
- $\boldsymbol{S} \leftarrow \emptyset$
- $\boldsymbol{Q} \leftarrow \boldsymbol{N}$
- **while** $\boldsymbol{Q} \neq \emptyset$
  - **do** $u \leftarrow$ EXTRACT-MIN($\boldsymbol{Q}$)
    - $\boldsymbol{S} \leftarrow \boldsymbol{S} \cup \{u\}$
    - **for each** $v \in Adj(u)$
      - **do if** $d(v) > d(u) + l_{(u, v)}$
        - **then** $d(v) \leftarrow d(u) + l_{(u, v)}$

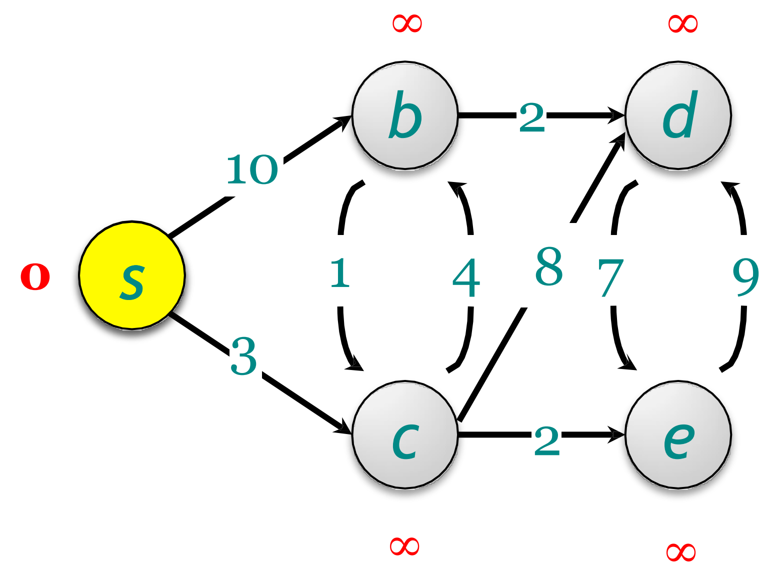



**S**={}

**Q**={s, b, c, d, e}

# Example 1.

- $d(s) \leftarrow 0$
- **for** each $v \in N - \{s\}$
  - **do** $d(v) \leftarrow \infty$
- $S \leftarrow \emptyset$
- $Q \leftarrow N$
- **while** $Q \neq \emptyset$
  - do $u \leftarrow$ EXTRACT-MIN($Q$)
    - $S \leftarrow S \cup \{u\}$
    - for each $v \in Adj(u)$
      - do if $d(v) > d(u) + l_{(u,v)}$
        - then $d(v) \leftarrow d(u) + l_{(u,v)}$



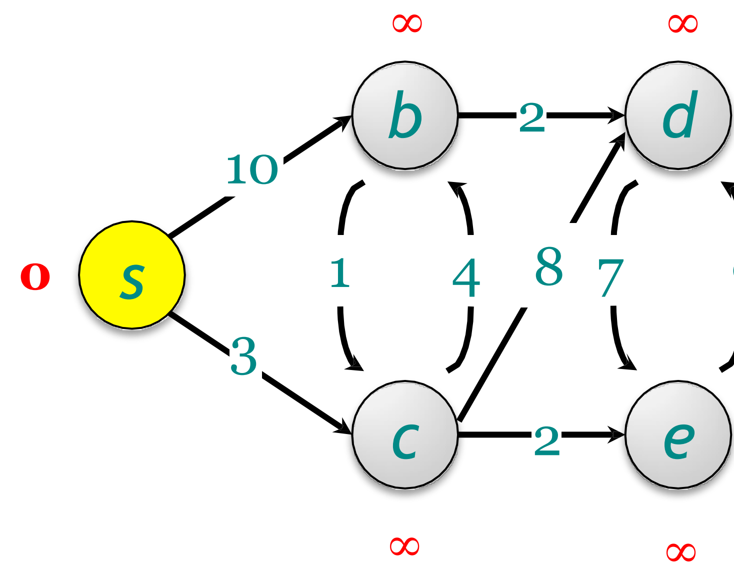

**S**={}

s

**Q**={b, c, d, e}

# Example 1.

- $d(s) \leftarrow 0$
- **for** each $v \in \boldsymbol{N} - \{s\}$
  - **do** $d(v) \leftarrow \infty$
- $\boldsymbol{S} \leftarrow \emptyset$
- $\boldsymbol{Q} \leftarrow \boldsymbol{N}$
- **while** $\boldsymbol{Q} \neq \emptyset$
  - **do** $u \leftarrow$ EXTRACT-MIN($\boldsymbol{Q}$)
    - $\boldsymbol{S} \leftarrow \boldsymbol{S} \cup \{u\}$
    - **for** each $v \in Adj(u)$
      - **do if** $d(v) > d(u) + l_{(u,v)}$
        - **then** $d(v) \leftarrow d(u) + l_{(u,v)}$

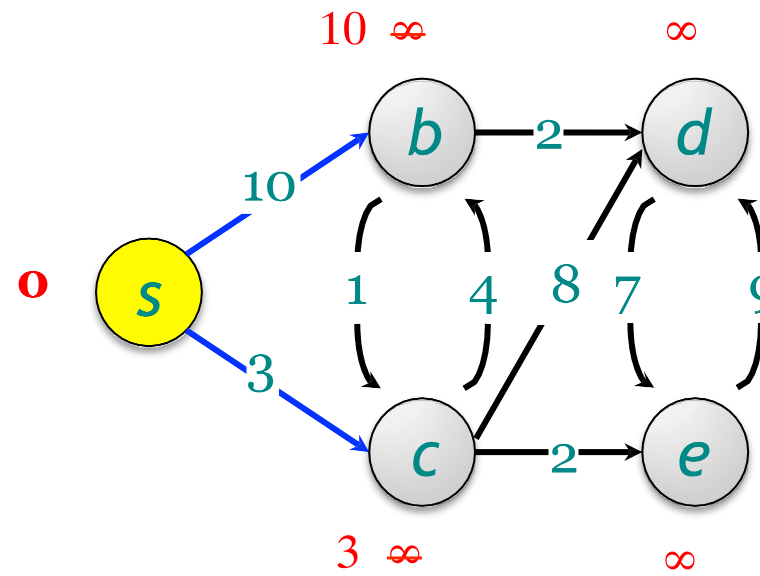



**S**={s}

**Q**={b, c, d, e}

# Example 1.

- $d(s) \leftarrow 0$
- **for** each $v \in \boldsymbol{N} - \{s\}$
  - **do** $d(v) \leftarrow \infty$
- $\boldsymbol{S} \leftarrow \emptyset$
- $\boldsymbol{Q} \leftarrow \boldsymbol{N}$
- **while** $\boldsymbol{Q} \neq \emptyset$
  - **do** $u \leftarrow$ EXTRACT-MIN($\boldsymbol{Q}$)
    - $\boldsymbol{S} \leftarrow \boldsymbol{S} \cup \{u\}$
    - **for** each $v \in Adj(u)$
      - **do if** $d(v) > d(u) + l_{(u,v)}$
        - **then** $d(v) \leftarrow d(u) + l_{(u,v)}$

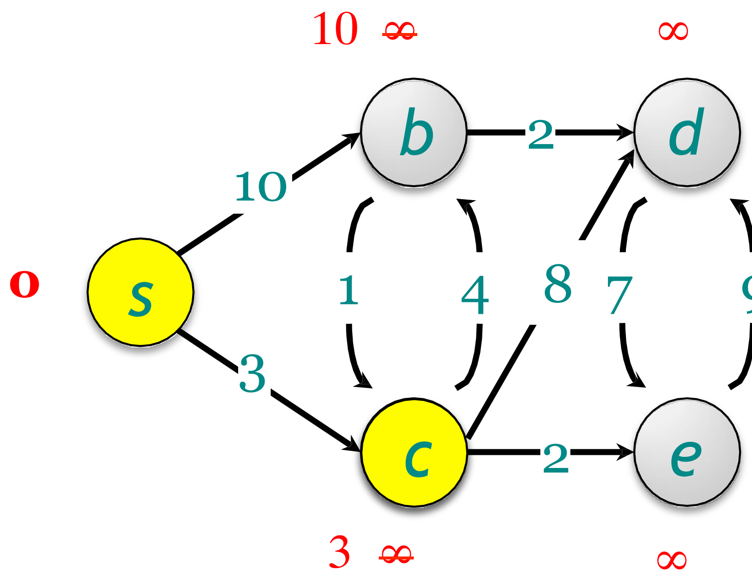



**S**={s}

**Q**={b, c, d, e}

# Example 1.

- $d(s) \leftarrow 0$
- **for** each $v \in N - \{s\}$
  - **do** $d(v) \leftarrow \infty$
- $S \leftarrow \varnothing$
- $Q \leftarrow N$
- **while** $Q \neq \varnothing$
  - **do** $u \leftarrow$ EXTRACT-MIN($Q$)
    - $S \leftarrow S \cup \{u\}$
    - **for** each $v \in Adj(u)$
      - **do if** $d(v) > d(u) + l_{(u, v)}$
        - **then** $d(v) \leftarrow d(u) + l_{(u, v)}$



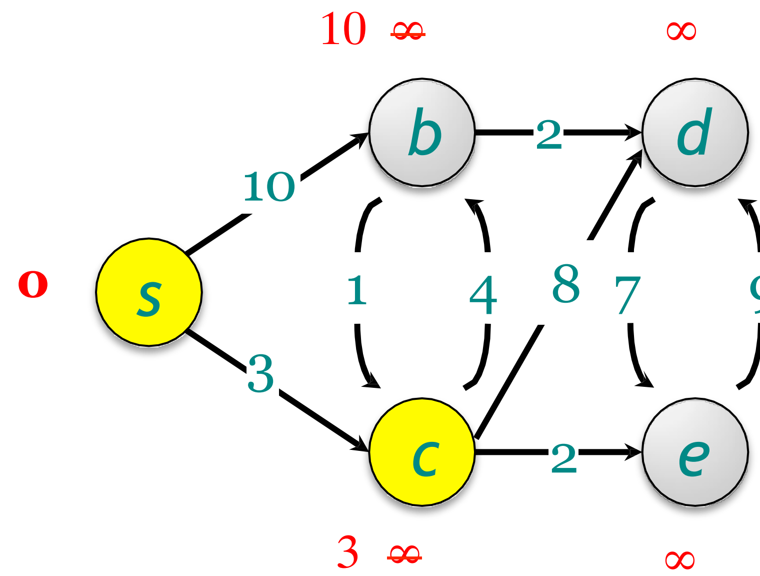

**S**={s}

c

**Q**={b, d, e}

# Example 1.

- $d(s) \leftarrow 0$
- **for** each $v \in N - \{s\}$
  - **do** $d(v) \leftarrow \infty$
- $S \leftarrow \emptyset$
- $Q \leftarrow N$
- **while** $Q \neq \emptyset$
  - **do** $u \leftarrow$ EXTRACT-MIN($Q$)
    - $S \leftarrow S \cup \{u\}$
    - **for** each $v \in Adj(u)$
      - **do if** $d(v) > d(u) + l_{(u, v)}$
        - **then** $d(v) \leftarrow d(u) + l_{(u, v)}$

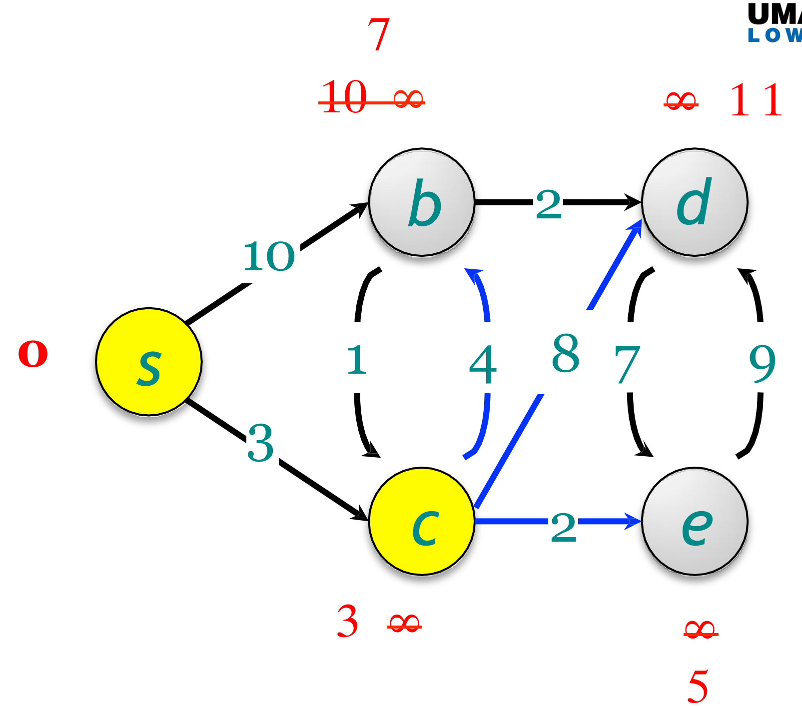



**S**={s, c}

**Q**={b, d, e}

# Example 1.

- $d(s) \leftarrow 0$
- **for** each $v \in N - \{s\}$
  - **do** $d(v) \leftarrow \infty$
- $S \leftarrow \emptyset$
- $Q \leftarrow N$
- **while** $Q \neq \emptyset$
  - **do** $u \leftarrow$ EXTRACT-MIN($Q$)
    - $S \leftarrow S \cup \{u\}$
    - **for** each $v \in Adj(u)$
      - **do if** $d(v) > d(u) + l_{(u,v)}$
        - **then** $d(v) \leftarrow d(u) + l_{(u,v)}$

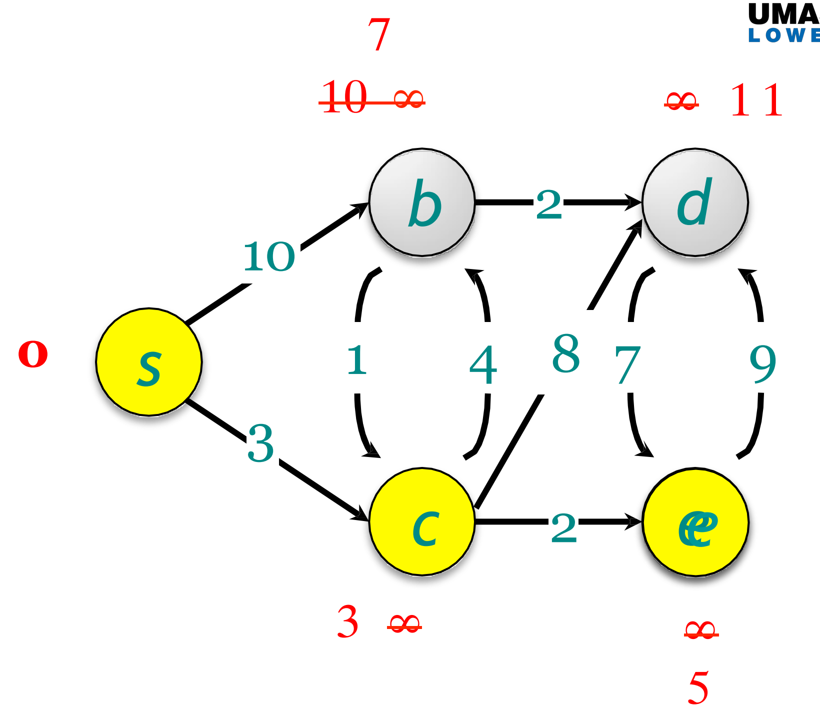



**S**={s, c}

**Q**={b, d, e}

# Example 1.

- $d(s) \leftarrow 0$
- **for** each $v \in N - \{s\}$
  - **do** $d(v) \leftarrow \infty$
- $S \leftarrow \emptyset$
- $Q \leftarrow N$
- **while** $Q \neq \emptyset$
  - **do** $u \leftarrow$ EXTRACT-MIN($Q$)
    - $S \leftarrow S \cup \{u\}$
    - **for** each $v \in Adj(u)$
      - **do if** $d(v) > d(u) + l_{(u, v)}$
        - **then** $d(v) \leftarrow d(u) + l_{(u, v)}$



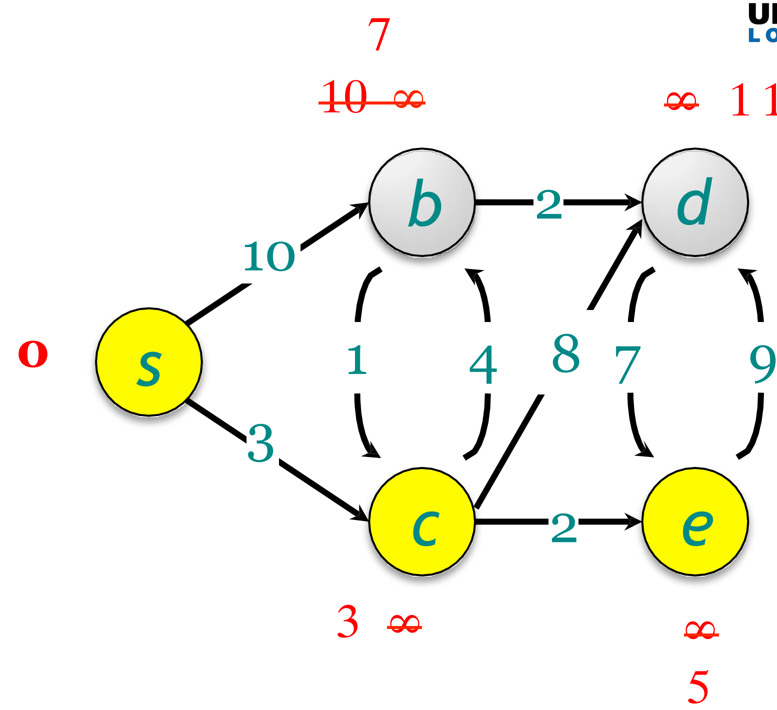

**S**={s, c}

e

**Q**={b, d}

# Example 1.

- $d(s) \leftarrow 0$
- **for** each $v \in \boldsymbol{N} - \{s\}$
  - **do** $d(v) \leftarrow \infty$
- $\boldsymbol{S} \leftarrow \varnothing$
- $\boldsymbol{Q} \leftarrow \boldsymbol{N}$
- **while** $\boldsymbol{Q} \neq \varnothing$
  - **do** $u \leftarrow$ EXTRACT-MIN($\boldsymbol{Q}$)
    - $\boldsymbol{S} \leftarrow \boldsymbol{S} \cup \{u\}$
    - **for** each $v \in Adj(u)$
      - **do if** $d(v) > d(u) + l_{(u,v)}$
        - **then** $d(v) \leftarrow d(u) + l_{(u,v)}$

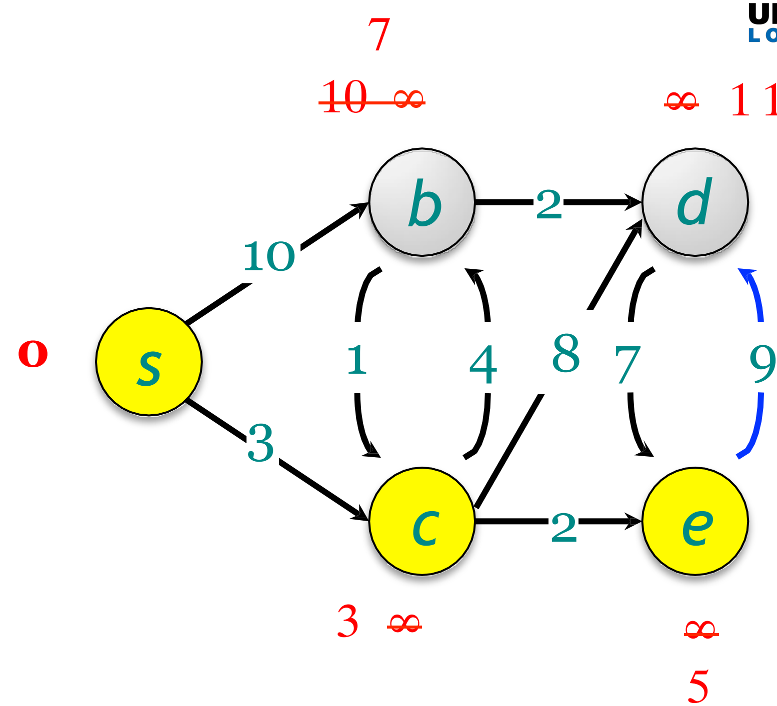



**S**={s, c, e}

**Q**={b, d}

# Example 1.

- $d(s) \leftarrow 0$
- **for** each $v \in N - \{s\}$
  - **do** $d(v) \leftarrow \infty$
- $S \leftarrow \varnothing$
- $Q \leftarrow N$
- **while** $Q \neq \varnothing$
  - **do** $u \leftarrow$ EXTRACT-MIN($Q$)
    - $S \leftarrow S \cup \{u\}$
    - **for** each $v \in Adj(u)$
      - **do if** $d(v) > d(u) + l_{(u, v)}$
        - **then** $d(v) \leftarrow d(u) + l_{(u, v)}$

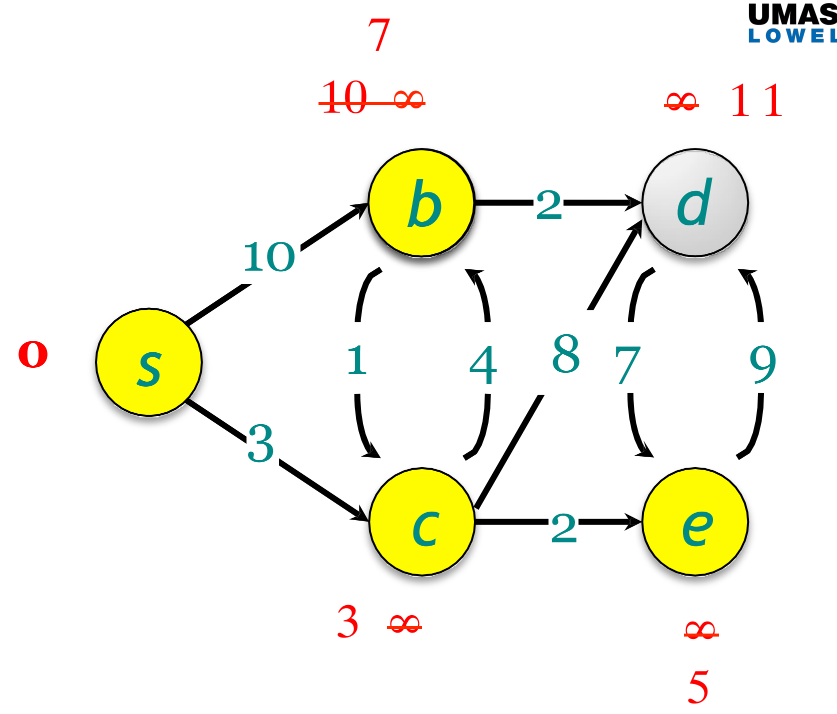



**S**={s, c, e}

**Q**={b, d}

# Example 1.

- $d(s) \leftarrow 0$
- **for** each $v \in N - \{s\}$
  - **do** $d(v) \leftarrow \infty$
- $S \leftarrow \varnothing$
- $Q \leftarrow N$
- **while** $Q \neq \varnothing$
  - **do** $u \leftarrow$ EXTRACT-MIN($Q$)
    - $S \leftarrow S \cup \{u\}$
    - **for** each $v \in Adj(u)$
      - **do if** $d(v) > d(u) + l_{(u, v)}$
        - **then** $d(v) \leftarrow d(u) + l_{(u, v)}$



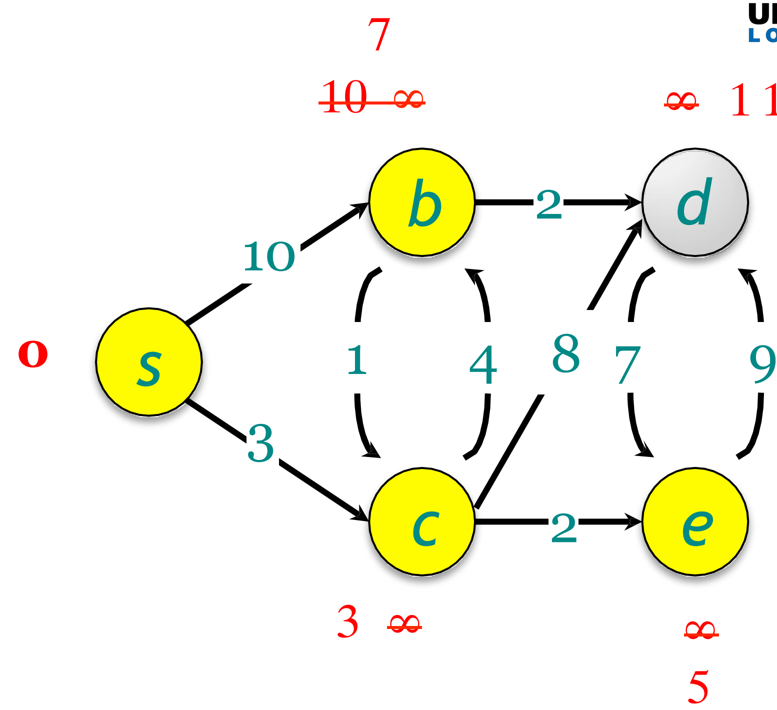

**S**={s, c, e}

b

**Q**={d}

# Example 1.

- $d(s) \leftarrow 0$
- **for each** $v \in N - \{s\}$
  - **do** $d(v) \leftarrow \infty$
- $S \leftarrow \emptyset$
- $Q \leftarrow N$
- **while** $Q \neq \emptyset$
  - **do** $u \leftarrow$ EXTRACT-MIN($Q$)
    - $S \leftarrow S \cup \{u\}$
    - **for each** $v \in Adj(u)$
      - **do if** $d(v) > d(u) + l_{(u,v)}$
        - **then** $d(v) \leftarrow d(u) + l_{(u,v)}$

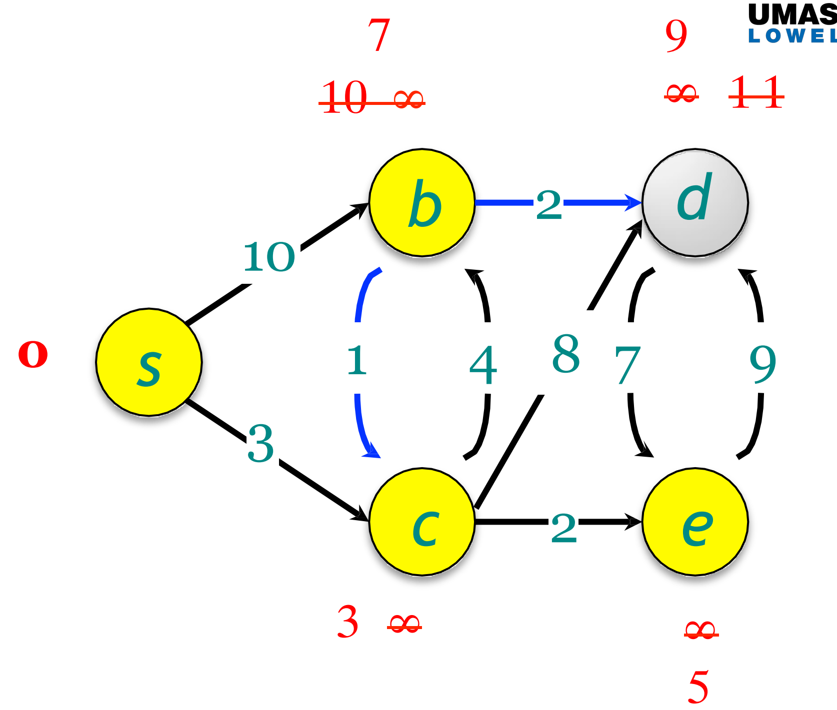



**S**={s, c, e, b}

**Q**={d}

# Example 1.

- $d(s) \leftarrow 0$
- **for** each $v \in N - \{s\}$
  - **do** $d(v) \leftarrow \infty$
- $S \leftarrow \varnothing$
- $Q \leftarrow N$
- **while** $Q \neq \varnothing$
  - **do** $u \leftarrow$ EXTRACT-MIN($Q$)
    - $S \leftarrow S \cup \{u\}$
    - **for** each $v \in Adj(u)$
      - **do if** $d(v) > d(u) + l_{(u,v)}$
        - **then** $d(v) \leftarrow d(u) + l_{(u,v)}$

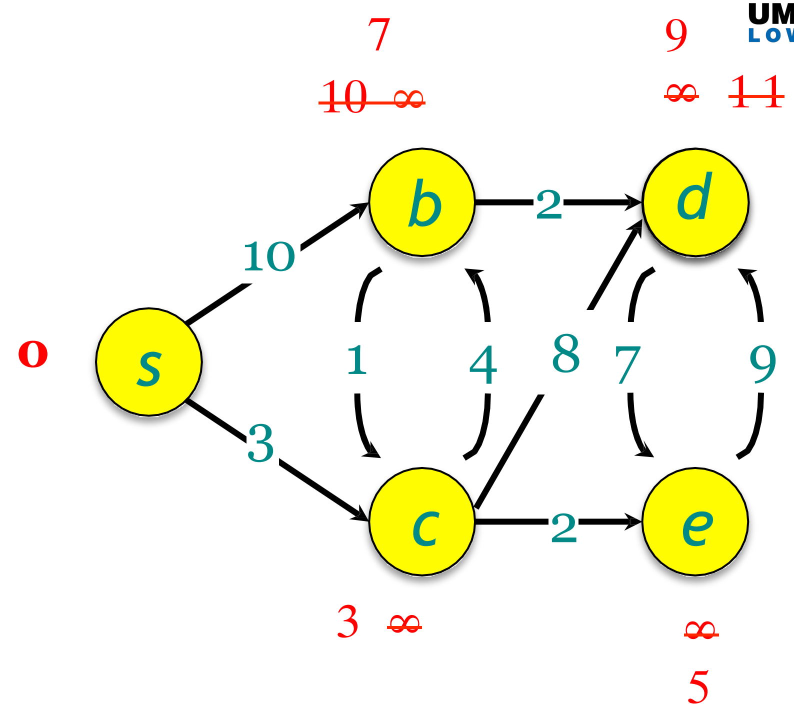



**S**={s, c, e, b}

**Q**={d}

# Example 1.

- $d(s) \leftarrow 0$
- **for each** $v \in N - \{s\}$
  - **do** $d(v) \leftarrow \infty$
- $S \leftarrow \emptyset$
- $Q \leftarrow N$
- **while** $Q \neq \emptyset$
  - do $u \leftarrow$ EXTRACT-MIN($Q$)
    - $S \leftarrow S \cup \{u\}$
    - for each $v \in Adj(u)$
      - do if $d(v) > d(u) + l_{(u, v)}$
        - then $d(v) \leftarrow d(u) + l_{(u, v)}$



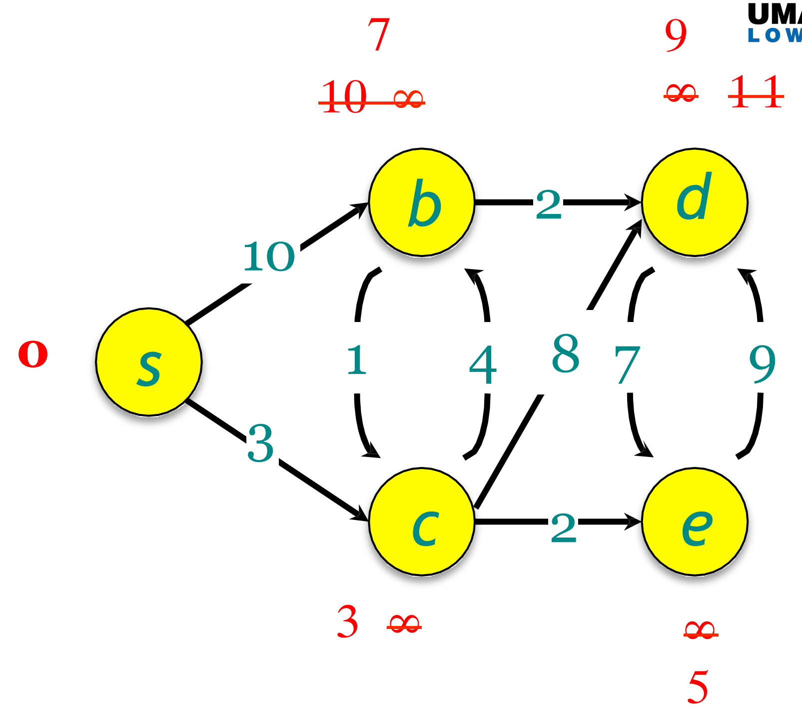

**S**={s, c, e, b}

d

**Q**={}

# Example 1.

- $d(s) \leftarrow 0$
- **for** each $v \in N - \{s\}$
  - **do** $d(v) \leftarrow \infty$
- $S \leftarrow \varnothing$
- $Q \leftarrow N$
- **while** $Q \neq \varnothing$
  - **do** $u \leftarrow$ EXTRACT-MIN($Q$)
    - $S \leftarrow S \cup \{u\}$
    - **for** each $v \in Adj(u)$
      - **do if** $d(v) > d(u) + l_{(u,v)}$
        - **then** $d(v) \leftarrow d(u) + l_{(u,v)}$

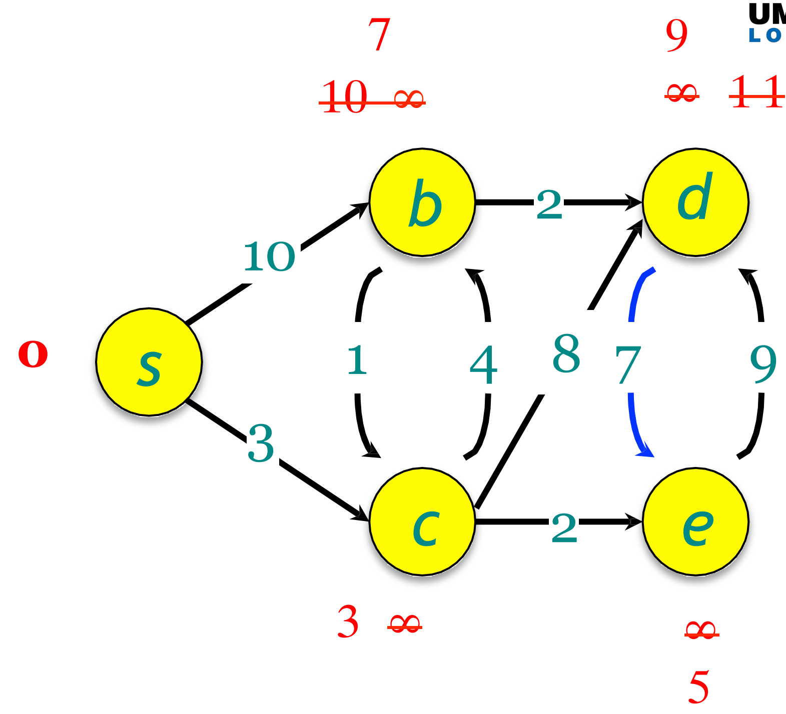



**S**={s, c, e, b, d}

**Q**={}

# Example 1.

- $d(s) \leftarrow 0$
- **for** each $v \in \boldsymbol{N} - \{s\}$
  - **do** $d(v) \leftarrow \infty$
- $\boldsymbol{S} \leftarrow \varnothing$
- $\boldsymbol{Q} \leftarrow \boldsymbol{N}$
- **while** $\boldsymbol{Q} \neq \varnothing$
  - **do** $u \leftarrow$ EXTRACT-MIN($\boldsymbol{Q}$)
    - $\boldsymbol{S} \leftarrow \boldsymbol{S} \cup \{u\}$
    - **for** each $v \in Adj(u)$
      - **do if** $d(v) > d(u) + l_{(u,v)}$
        - **then** $d(v) \leftarrow d(u) + l_{(u,v)}$

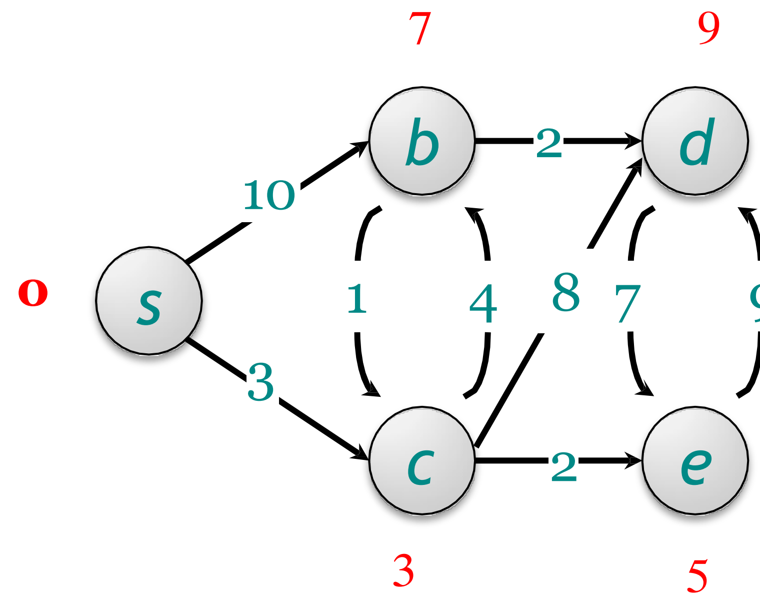



**S**={s, c, e, b, d}

**Q**={}

# Example 1.

- $d(s) \leftarrow 0$
- **for each** $v \in \boldsymbol{N} - \{s\}$
  - **do** $d(v) \leftarrow \infty$
- $\boldsymbol{S} \leftarrow \emptyset$
- $\boldsymbol{Q} \leftarrow \boldsymbol{N}$
- **while** $\boldsymbol{Q} \neq \emptyset$
  - **do** $u \leftarrow$ EXTRACT-MIN($\boldsymbol{Q}$)
    - $\boldsymbol{S} \leftarrow \boldsymbol{S} \cup \{u\}$
    - **for each** $v \in Adj(u)$
      - **do if** $d(v) > d(u) + l_{(u,v)}$
        - **then** $d(v) \leftarrow d(u) + l_{(u,v)}$




**S**={s, c, e, b, d}

**Q**={}

# Dijkstra’s algorithm- Cnt.

- Dijkstra’s algorithm computes the shortest distances btw a start node and all other nodes in the graph (not only a target node)!

- Assumptions:
  - the graph is connected, and
  - the weights are nonnegative

# Dijkstra's algorithm- Analysis

- $d(s) \leftarrow 0$
- **for** each $v \in \boldsymbol{N} - \{s\}$
  - **do** $d(v) \leftarrow \infty$
- $\boldsymbol{S} \leftarrow \varnothing$
- $\boldsymbol{Q} \leftarrow \boldsymbol{N}$
- **while** $\boldsymbol{Q} \neq \varnothing$
  - **do** $u \leftarrow$ EXTRACT-MIN($\boldsymbol{Q}$)
    - $\boldsymbol{S} \leftarrow \boldsymbol{S} \cup \{u\}$
    - **for** each $v \in Adj(u)$
      - **do if** $d(v) > d(u) + l_{(u,v)}$
        - **then** $d(v) \leftarrow d(u) + l_{(u,v)}$

**degree (u)** times

$|N|$ times

Time = $\Theta(N \cdot T_{\text{EXTRACT-MIN}} + E \cdot T_{\text{Relaxation}})$ , Handshaking Lemma!

# Dijkstra's algorithm- Analysis- Cnt.

$$\text{Time} = \Theta(N \cdot T_{\text{EXTRACT-MIN}} + E \cdot T_{\text{Relaxation}})$$

| $Q$ | $T_{\text{EXTRACT-MIN}}$ | $T_{\text{DECREASE-KEY}}$ | Total |
|---|---|---|---|
| Array | $O(N)$ | $O(1)$ | $O(N^2)$ |

# Reading

- Ch.24 Single Source Shortest Paths [CLRS]
- What is Twitter, a social network or a news media? Kwak, H., et al. WWW 2010.
- Global connectivity and multilinguals in the Twitter network. Hale, S.A., SIGCHI'14.
- Fragile online relationship: a first look at unfollow dynamics in twitter. Kwak, H., et al. SIGCHI'11.
- Understanding the demographics of twitter users. Mislove, A., et al. AAAI'11