# Greedy algorithms

Build a solution **greedily** by making the best **local** decision in each step to build an optimal **global** solution.

local - at each step, only consider a small part of input.

greedy - at each step, make choice optimizing criterion, often a proxy for the overall criterion.

examples?

stable matching - Gale Shapley ✓

# Single-pair shortest path problem

directed graph

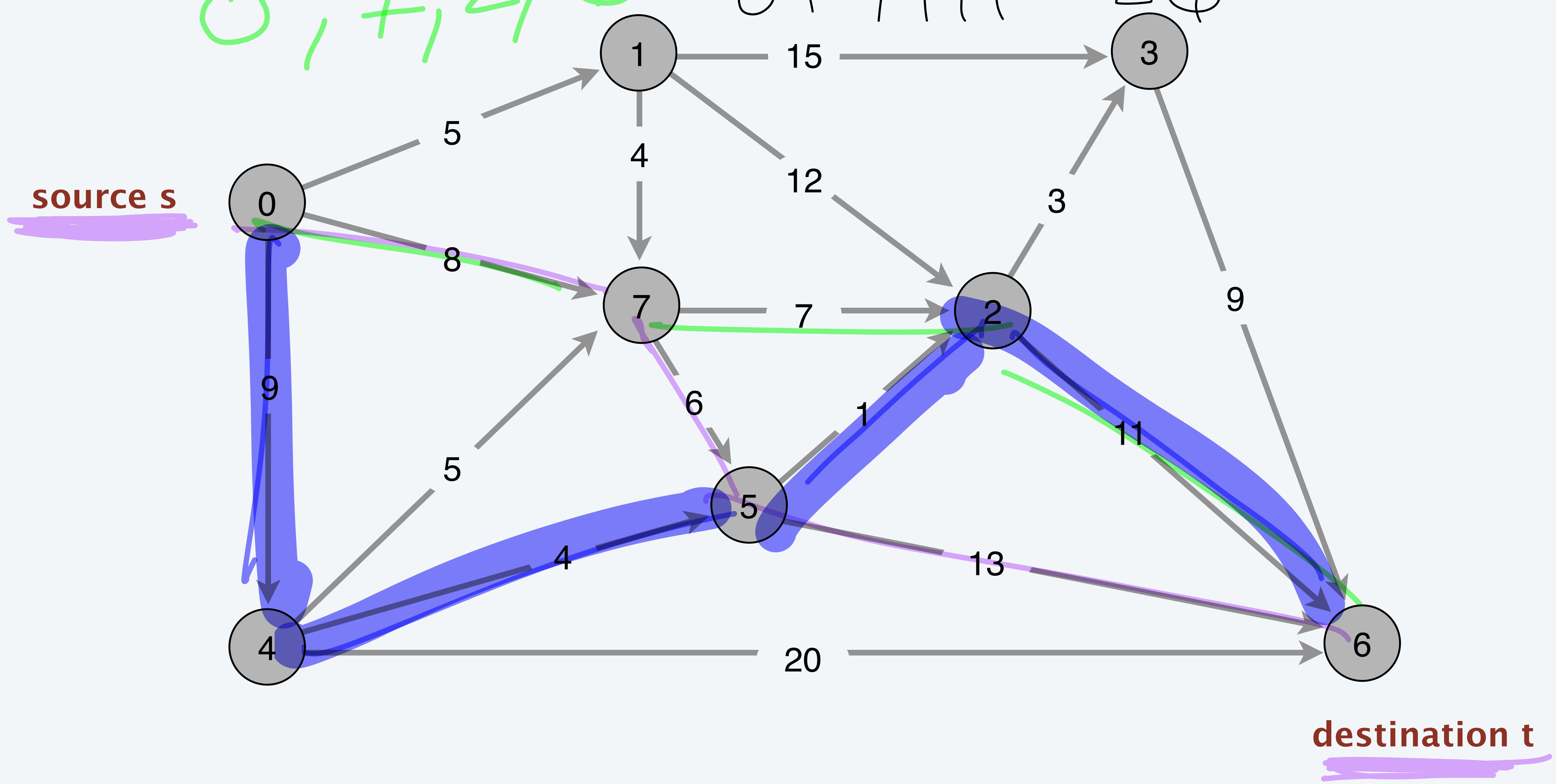
**Problem.** Given a digraph $G = (V, E)$, edge lengths $\ell_e \geq 0$, source $s \in V$, and destination $t \in V$, find a shortest directed path from $s$ to $t$.

9+4+1+11=25

8+6+13 =27 = 0,7,5,6

0,4,5,2,6

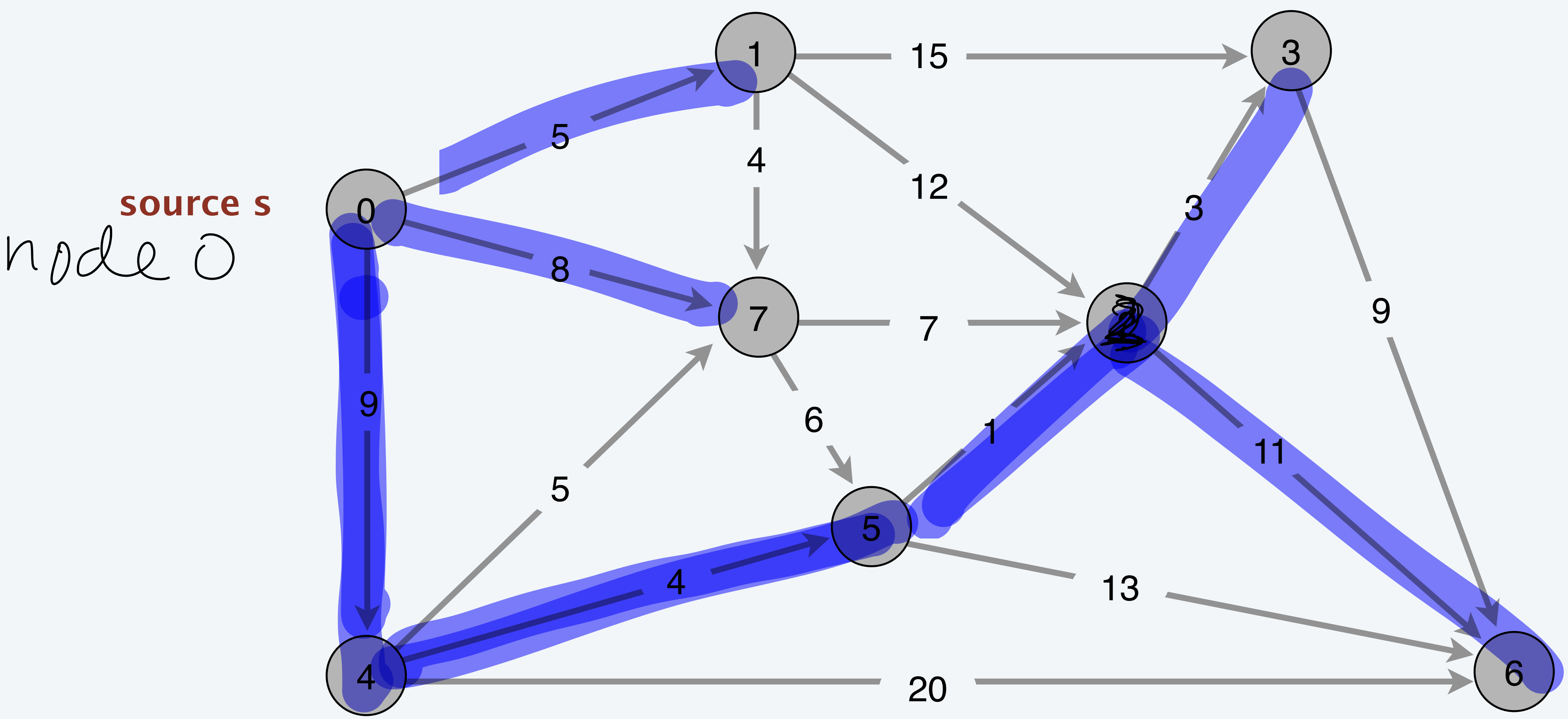
0,7,2,6 = 8+7+11=26



# Single-source shortest paths problem — s to every node

**Problem.** Given a digraph $G = (V, E)$, edge lengths $\ell_e \geq 0$, source $s \in V$, find a shortest directed path from $s$ to every node.

Single-pair shortest path:
s to t

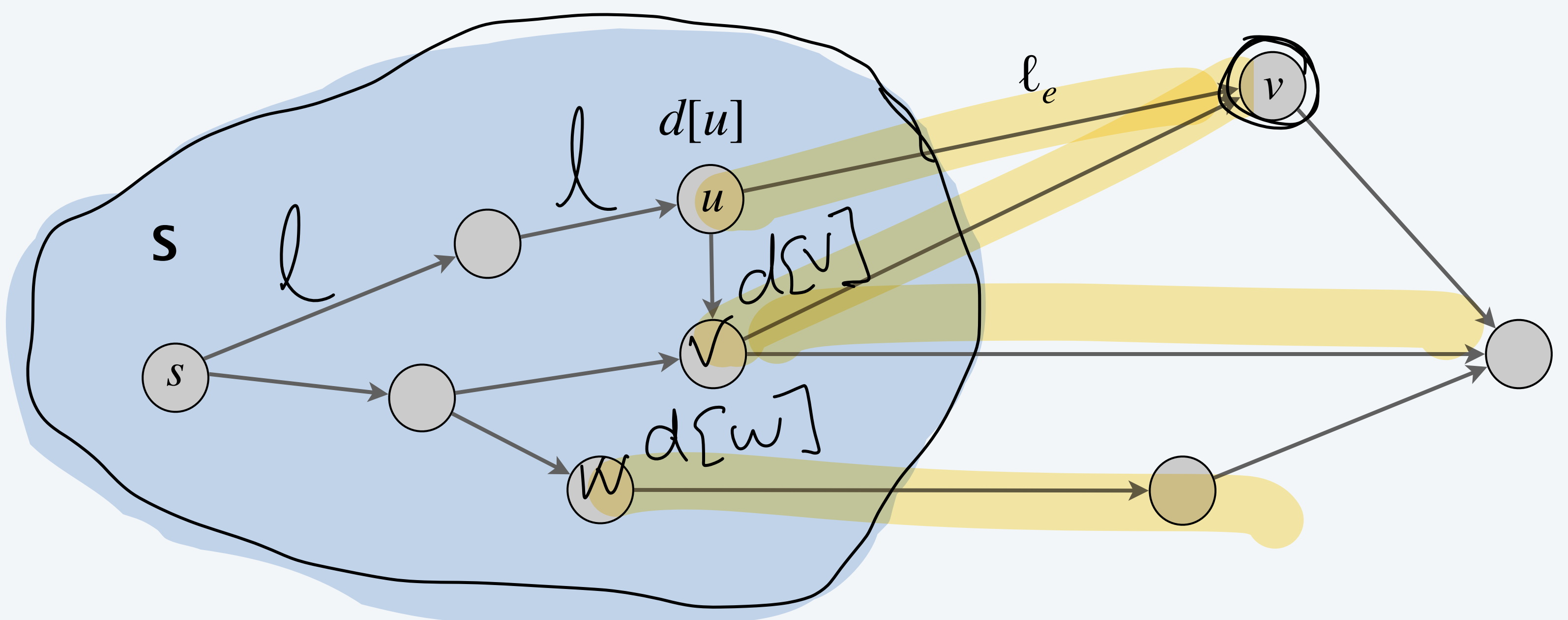


# Dijkstra's algorithm (for single-source shortest paths problem)

Greedy

Maintain a set of explored nodes S for which the algorithm has determined d[u] = length of a shortest path to u.

Add to v the unexplored node $v \notin S$ that minimizes $\pi(v) = \min_{e:(u,v),\, u \in S} \left( d[u] + \ell_e \right)$



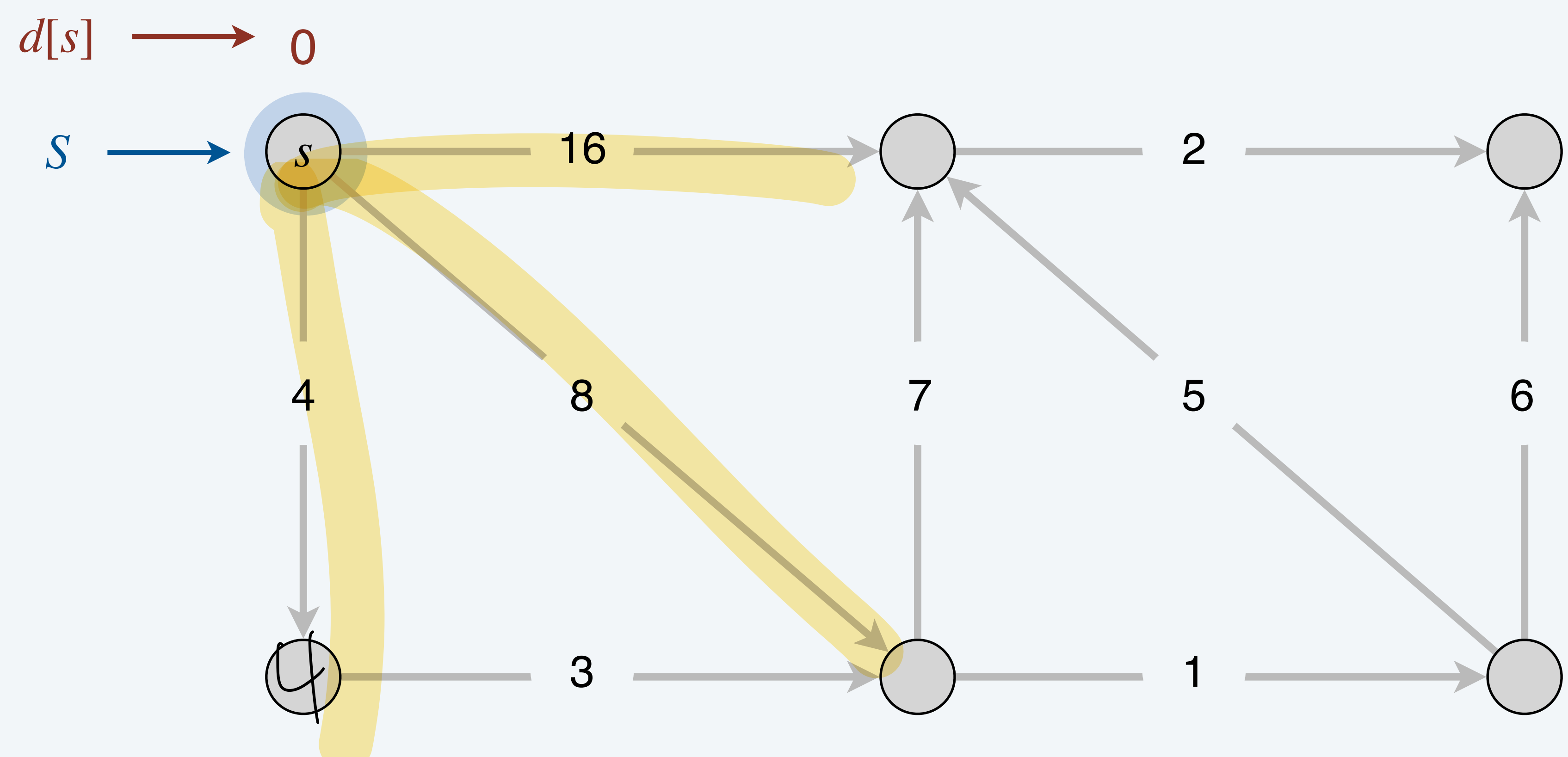
# Dijkstra's algorithm demo

- Initialize $S \leftarrow \{ s \}$ and $d[s] \leftarrow 0$.
- Repeatedly choose unexplored node $v \notin S$ which minimizes

$$\pi(v) \;=\; \min_{e \,=\, (u,v) \;:\; u \in S} d[u] + \ell_e$$

the length of a shortest path from $s$ to some node $u$ in explored part $S$, followed by a single edge $e = (u, v)$

add $v$ to $S$; set $d[v] \leftarrow \pi(v)$ and $pred[v] \leftarrow$ argmin.



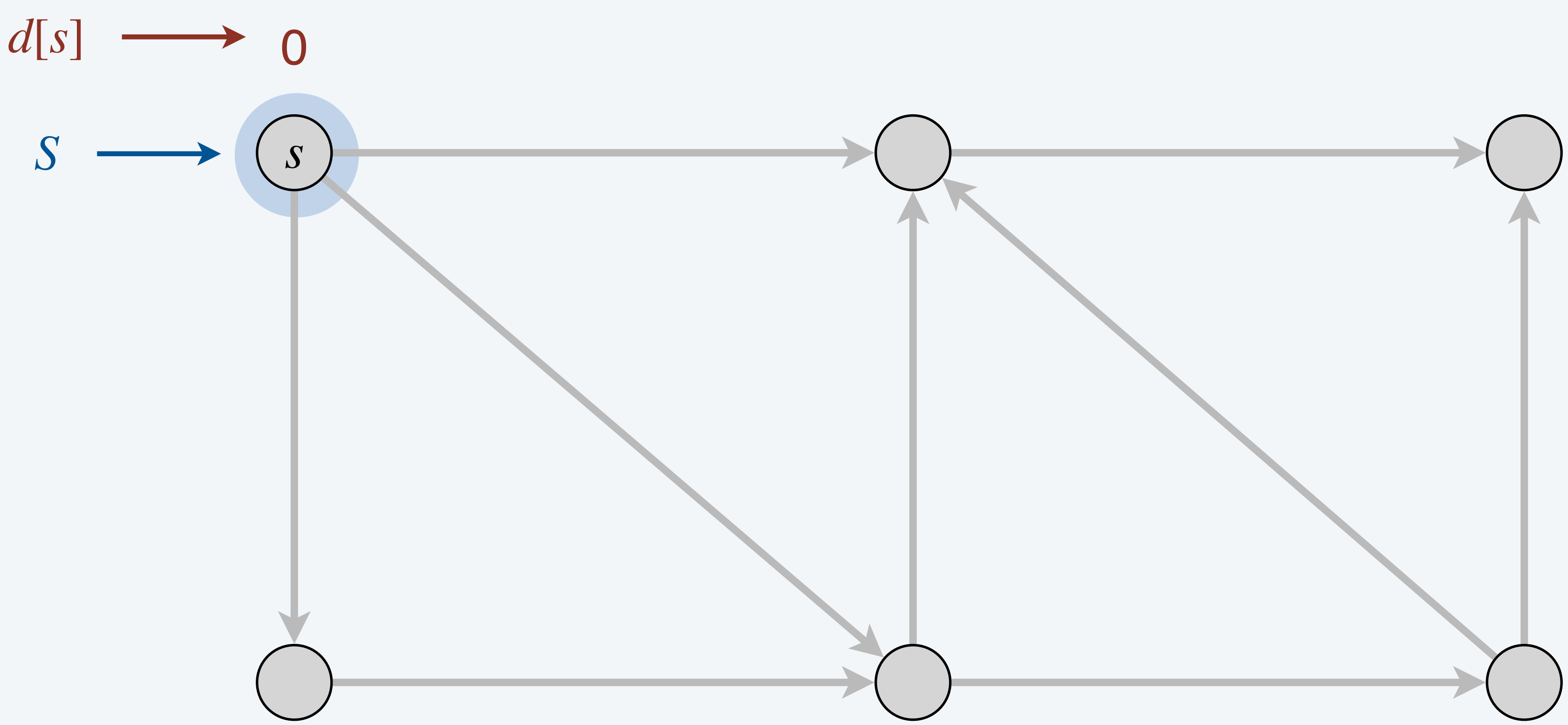
# Dijkstra's algorithm demo

- Initialize $S \leftarrow \{ s \}$ and $d[s] \leftarrow 0$.
- Repeatedly choose unexplored node $v \notin S$ which minimizes

$$\pi(v) \;=\; \min_{e\,=\,(u,v)\;:\;u \in S} d[u] + \ell_e$$

the length of a shortest path from $s$ to some node $u$ in explored part $S$, followed by a single edge $e = (u, v)$

add $v$ to $S$; set $d[v] \leftarrow \pi(v)$ and $pred[v] \leftarrow$ argmin.



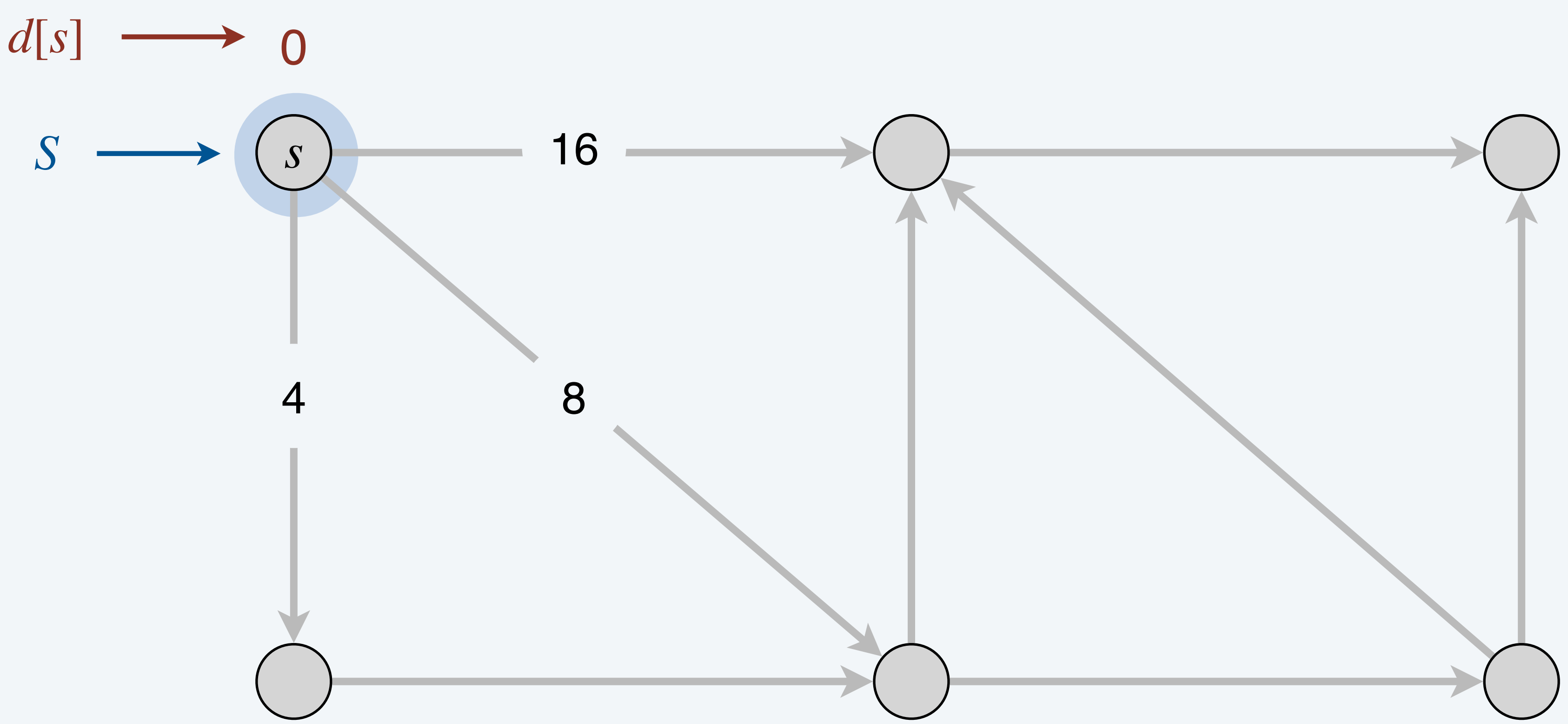
# Dijkstra's algorithm demo

- Initialize $S \leftarrow \{ s \}$ and $d[s] \leftarrow 0$.
- Repeatedly choose unexplored node $v \notin S$ which minimizes

$$\pi(v) \;=\; \min_{e \,=\, (u,v)\,:\, u \in S} d[u] + \ell_e$$

the length of a shortest path from $s$ to some node $u$ in explored part $S$, followed by a single edge $e = (u, v)$

add $v$ to $S$; set $d[v] \leftarrow \pi(v)$ and $pred[v] \leftarrow$ argmin.



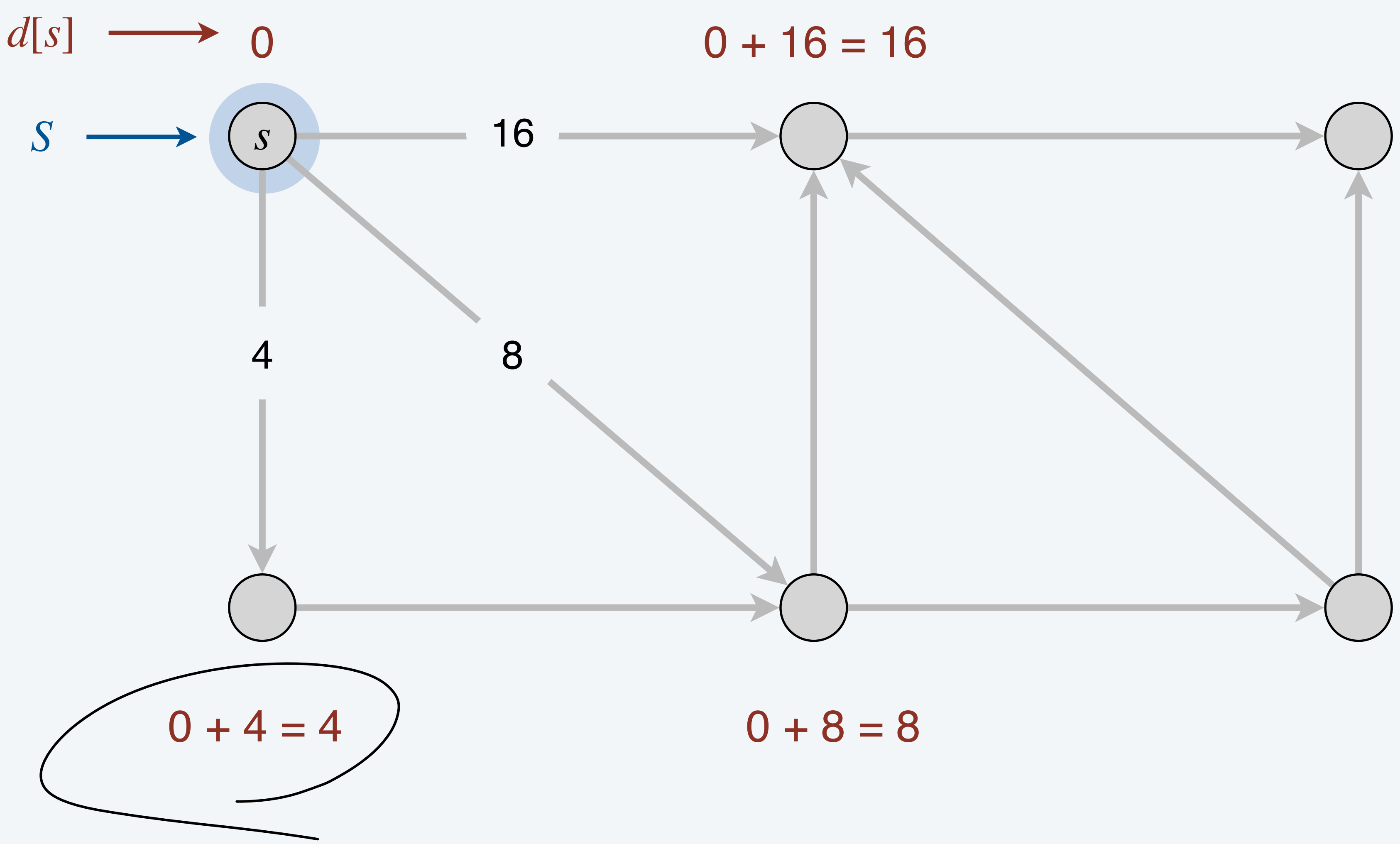
# Dijkstra's algorithm demo

- Initialize $S \leftarrow \{ s \}$ and $d[s] \leftarrow 0$.
- Repeatedly choose unexplored node $v \notin S$ which minimizes

$$\pi(v) \;=\; \min_{e = (u,v)\,:\, u \in S} d[u] + \ell_e$$

the length of a shortest path from $s$ to some node $u$ in explored part $S$, followed by a single edge $e = (u, v)$

add $v$ to $S$; set $d[v] \leftarrow \pi(v)$ and $pred[v] \leftarrow$ argmin.



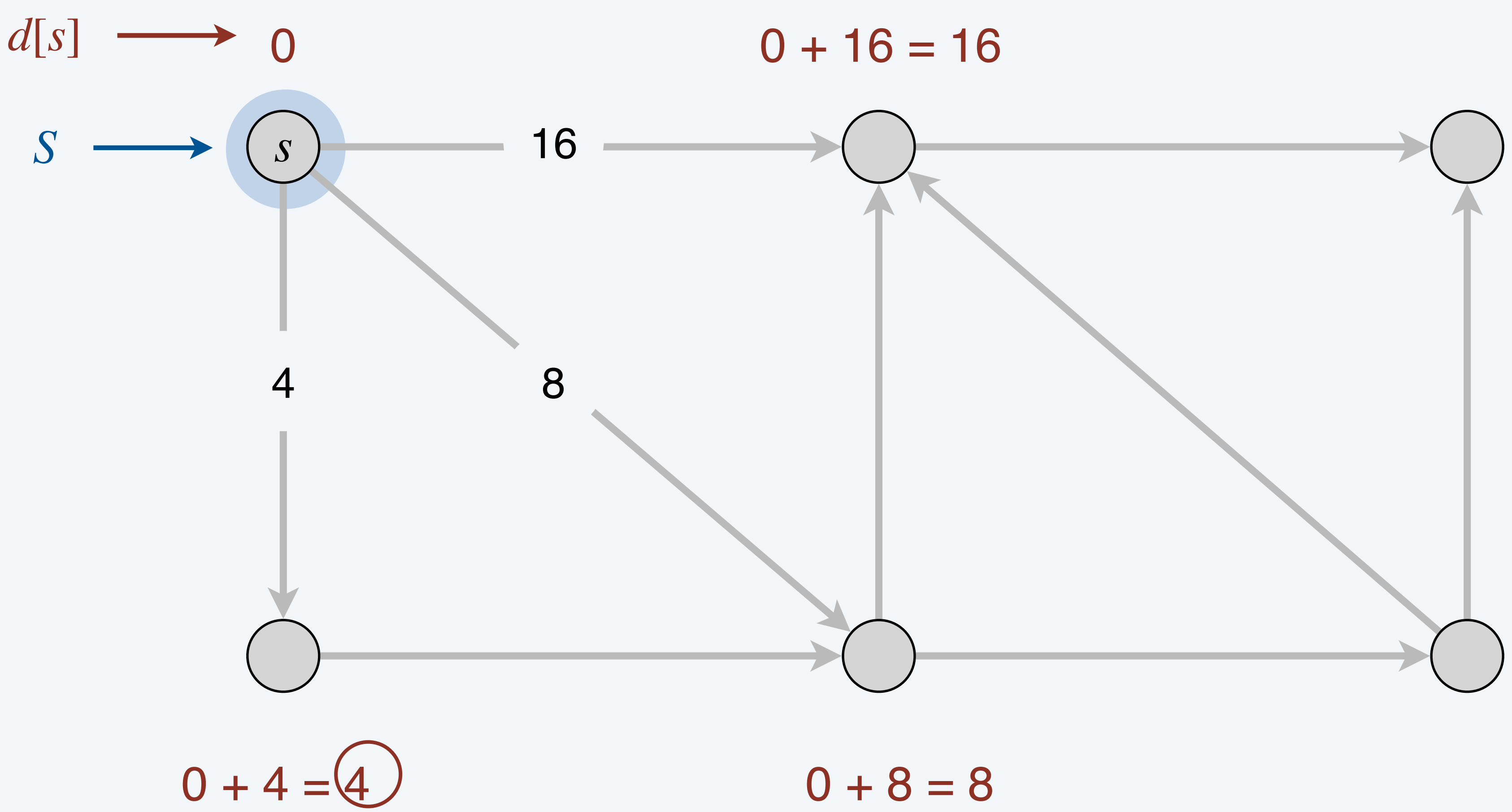
# Dijkstra's algorithm demo

- Initialize $S \leftarrow \{ s \}$ and $d[s] \leftarrow 0$.
- Repeatedly choose unexplored node $v \notin S$ which minimizes

$$\pi(v) = \min_{e = (u,v) \,:\, u \in S} d[u] + \ell_e$$

the length of a shortest path from $s$ to some node $u$ in explored part $S$, followed by a single edge $e = (u, v)$

add $v$ to $S$; set $d[v] \leftarrow \pi(v)$ and $pred[v] \leftarrow$ argmin.



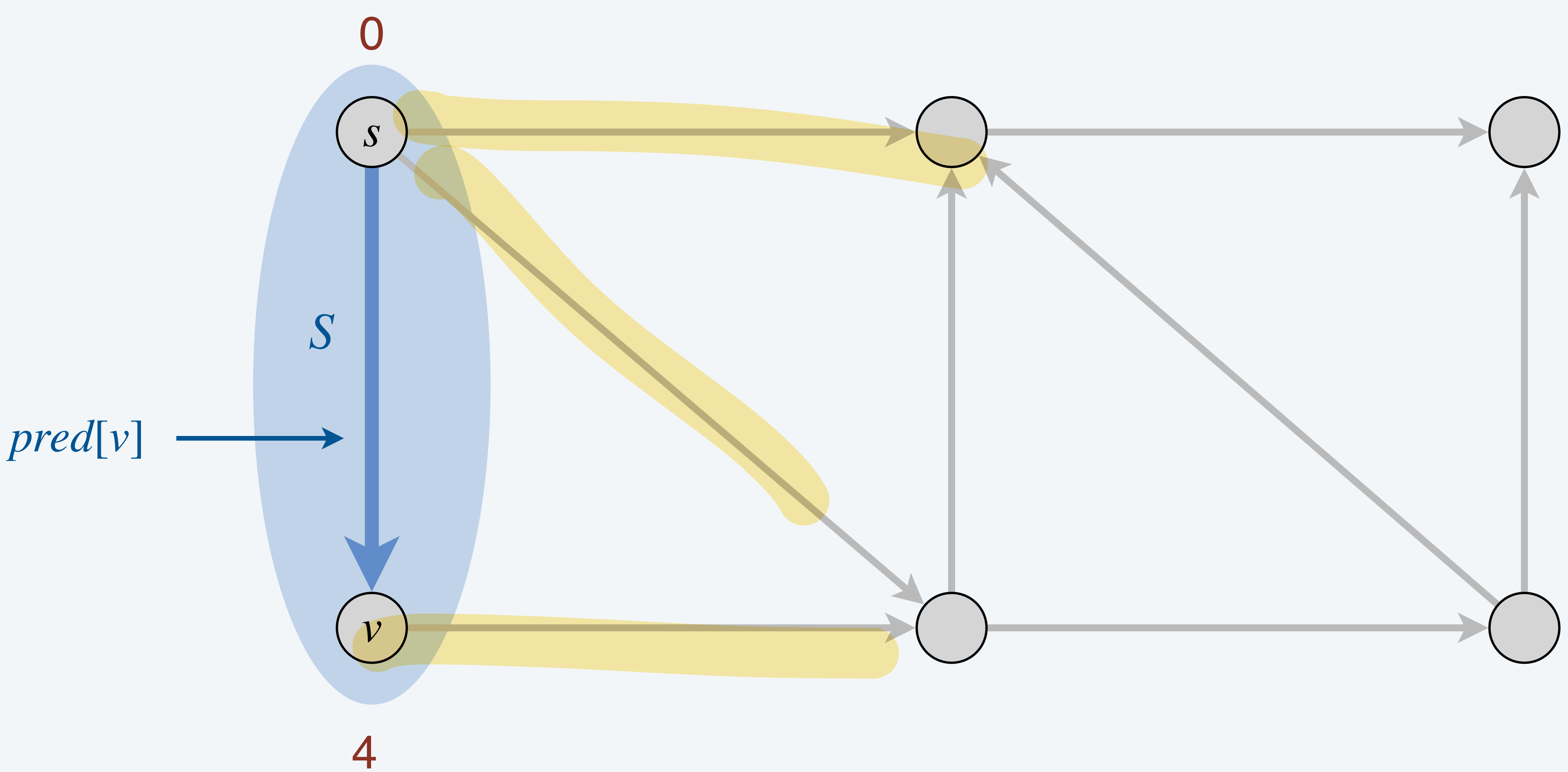
# Dijkstra's algorithm demo

- Initialize $S \leftarrow \{ s \}$ and $d[s] \leftarrow 0$.
- Repeatedly choose unexplored node $v \notin S$ which minimizes

$$\pi(v) = \min_{e = (u,v) \,:\, u \in S} d[u] + \ell_e$$

the length of a shortest path from $s$ to some node $u$ in explored part $S$, followed by a single edge $e = (u, v)$

add $v$ to $S$; set $d[v] \leftarrow \pi(v)$ and $pred[v] \leftarrow$ argmin.



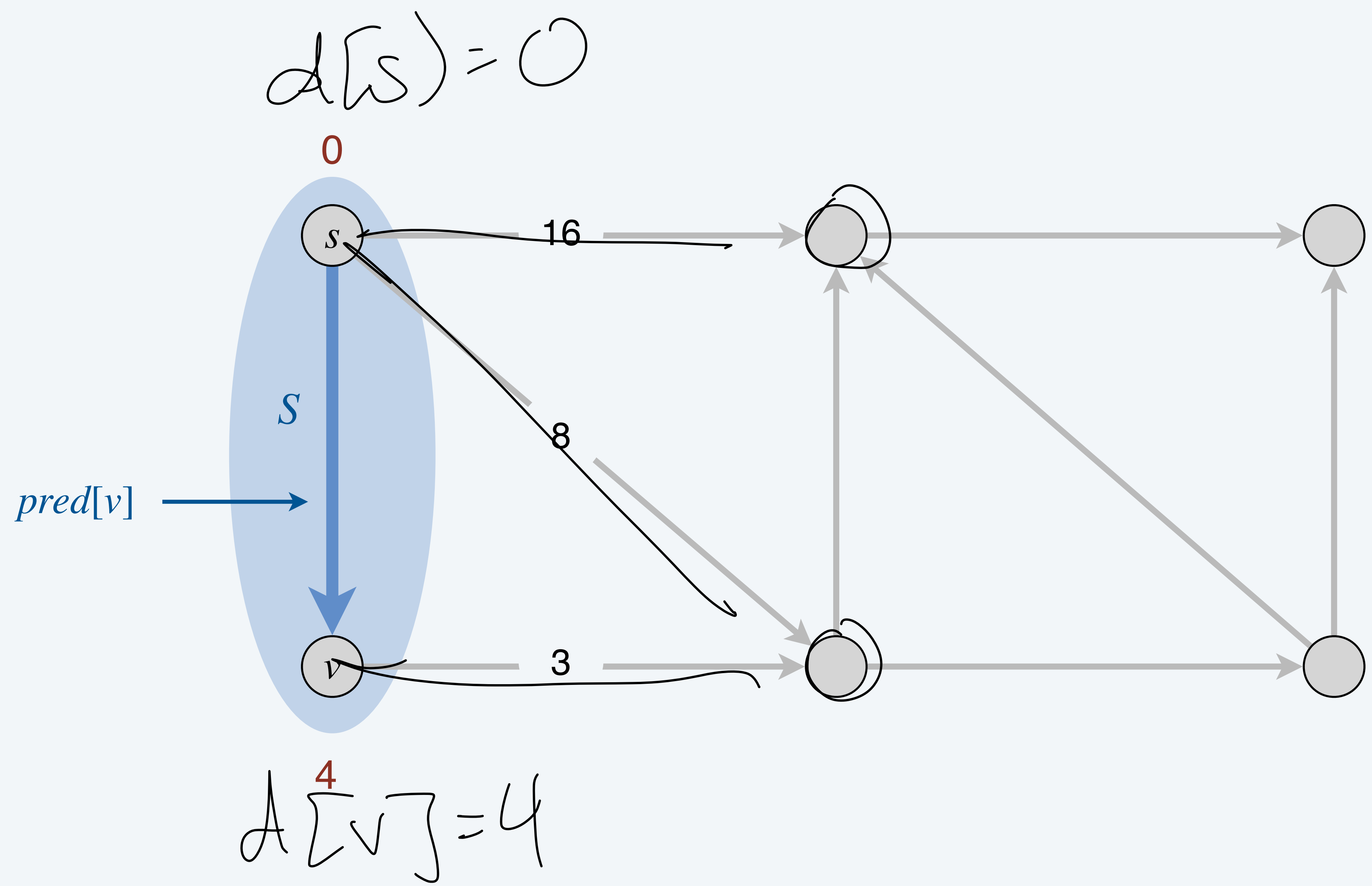
# Dijkstra's algorithm demo

- Initialize $S \leftarrow \{ s \}$ and $d[s] \leftarrow 0$.
- Repeatedly choose unexplored node $v \notin S$ which minimizes

$$\pi(v) \;=\; \min_{e = (u,v)\,:\,u \in S} d[u] + \ell_e$$

the length of a shortest path from $s$ to some node $u$ in explored part $S$, followed by a single edge $e = (u, v)$

add $v$ to $S$; set $d[v] \leftarrow \pi(v)$ and $pred[v] \leftarrow$ argmin.



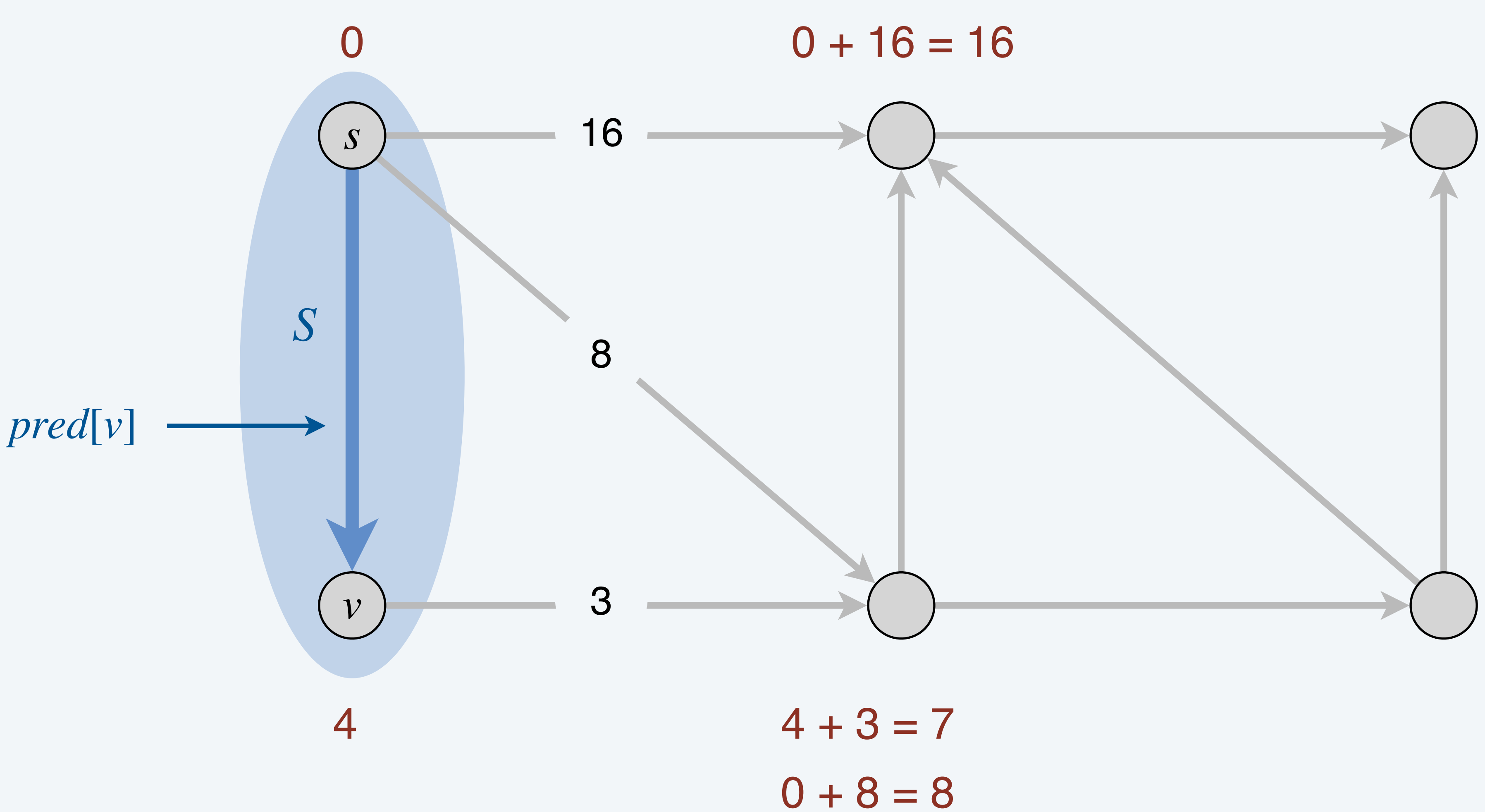
# Dijkstra's algorithm demo

- Initialize $S \leftarrow \{ s \}$ and $d[s] \leftarrow 0$.
- Repeatedly choose unexplored node $v \notin S$ which minimizes

$$\pi(v) = \min_{e = (u,v) \,:\, u \in S} d[u] + \ell_e$$

the length of a shortest path from $s$ to some node $u$ in explored part $S$, followed by a single edge $e = (u, v)$

add $v$ to $S$; set $d[v] \leftarrow \pi(v)$ and $pred[v] \leftarrow$ argmin.



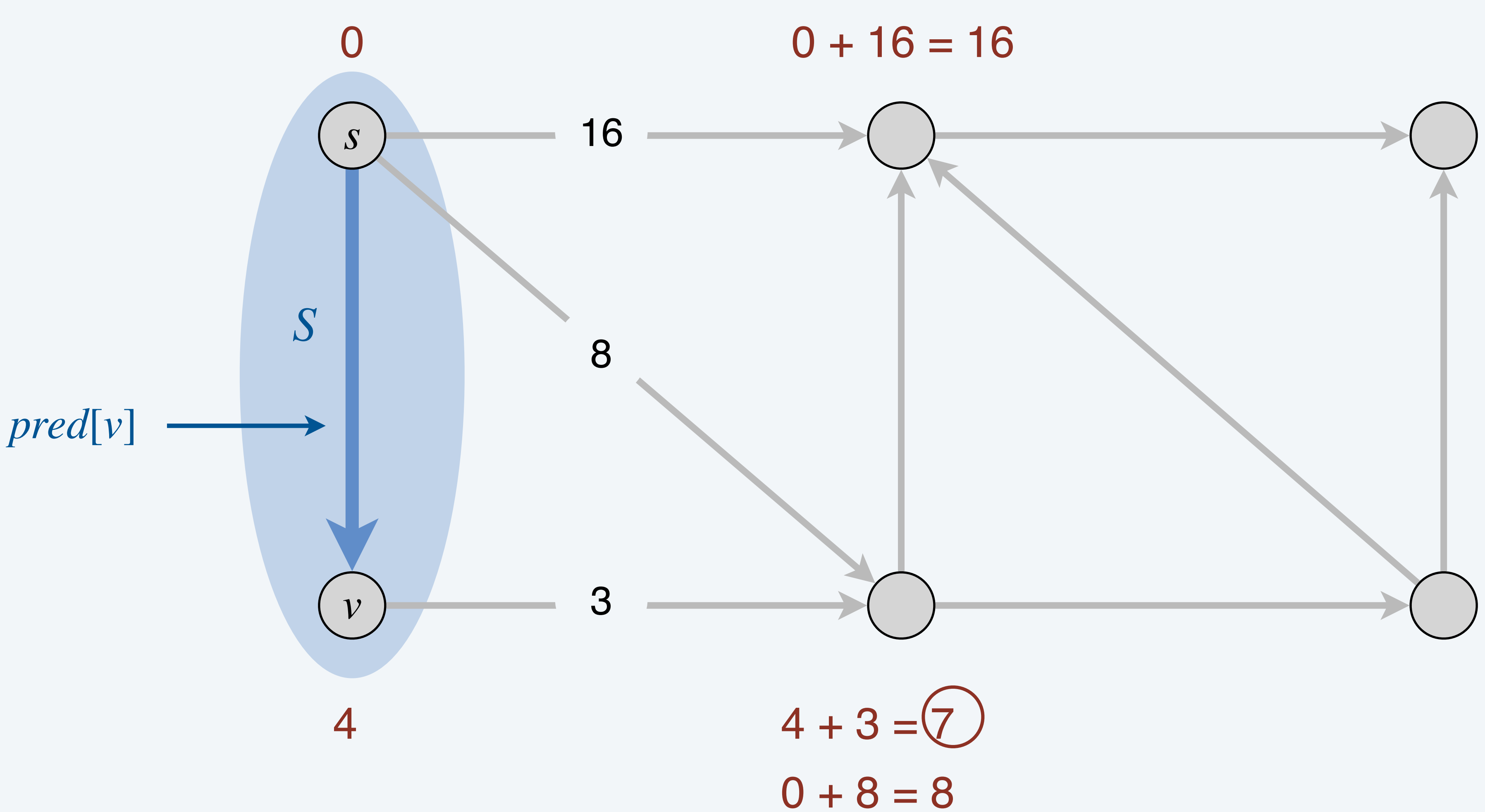
# Dijkstra's algorithm demo

- Initialize $S \leftarrow \{ s \}$ and $d[s] \leftarrow 0$.
- Repeatedly choose unexplored node $v \notin S$ which minimizes

$$\pi(v) = \min_{e = (u,v) \,:\, u \in S} d[u] + \ell_e$$

the length of a shortest path from $s$ to some node $u$ in explored part $S$, followed by a single edge $e = (u, v)$

add $v$ to $S$; set $d[v] \leftarrow \pi(v)$ and $pred[v] \leftarrow \text{argmin}$.



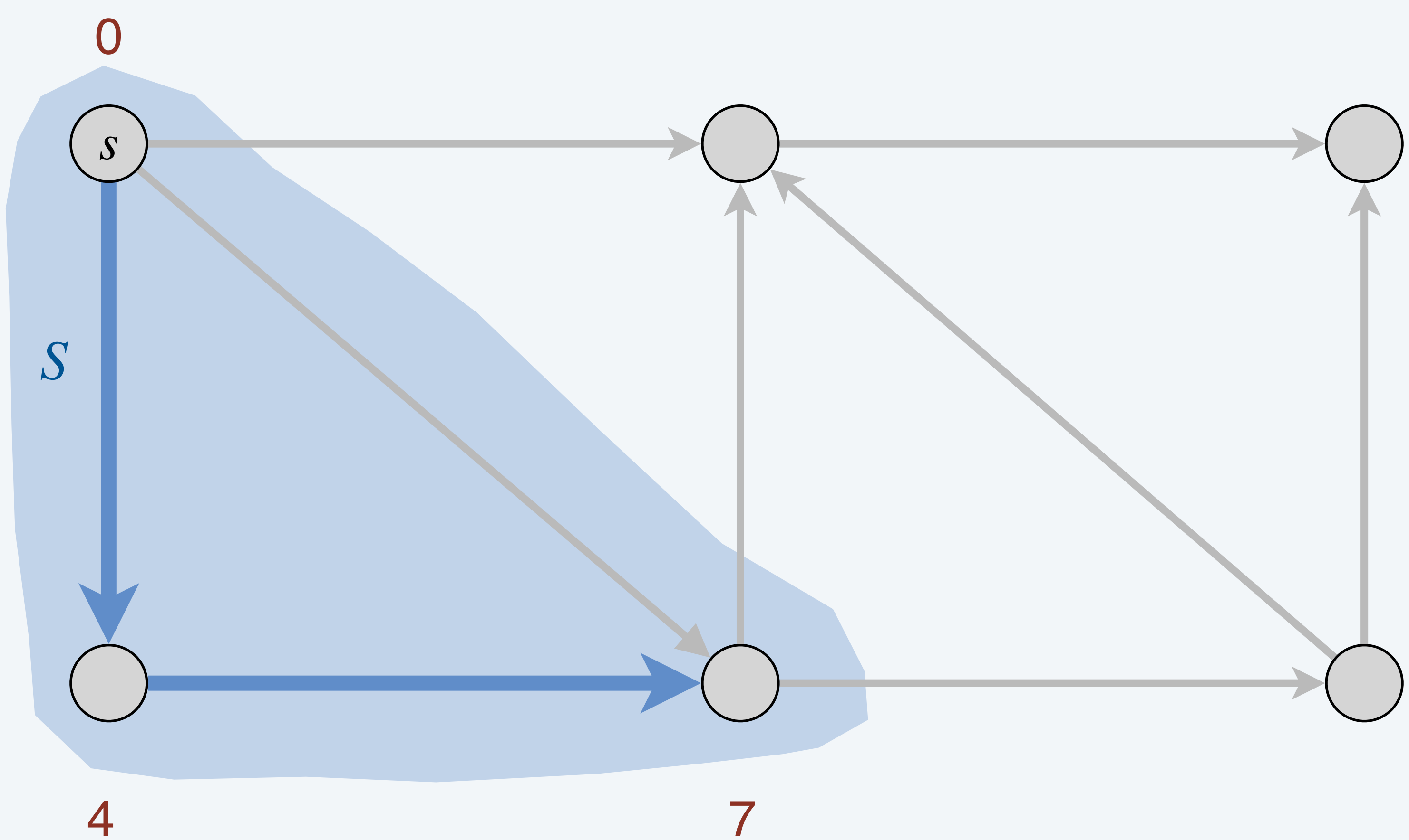
# Dijkstra's algorithm demo

- Initialize $S \leftarrow \{ s \}$ and $d[s] \leftarrow 0$.
- Repeatedly choose unexplored node $v \notin S$ which minimizes

$$\pi(v) = \min_{e = (u,v) \, : \, u \in S} d[u] + \ell_e$$

the length of a shortest path from $s$ to some node $u$ in explored part $S$, followed by a single edge $e = (u, v)$

add $v$ to $S$; set $d[v] \leftarrow \pi(v)$ and $pred[v] \leftarrow$ argmin.



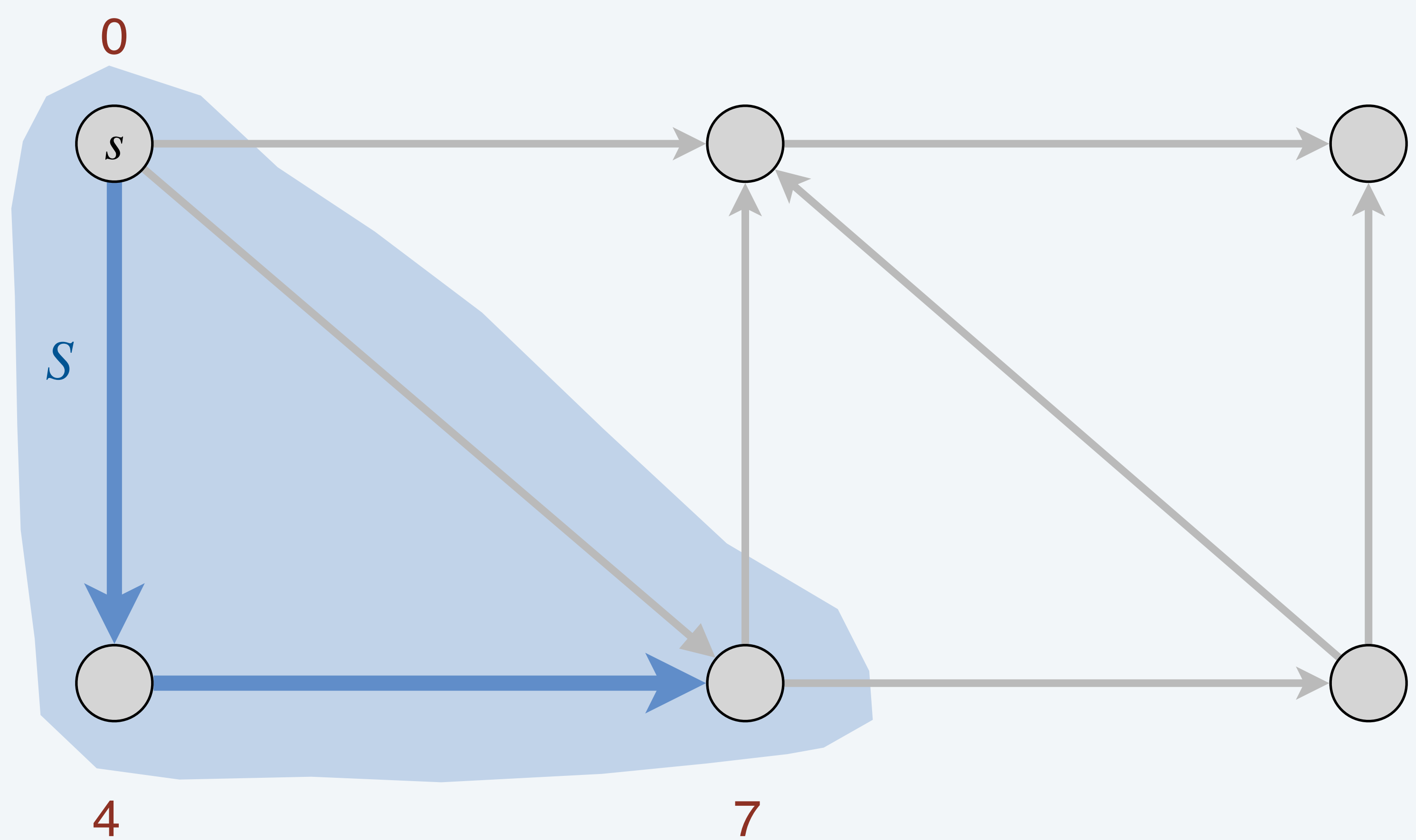
# Dijkstra's algorithm demo ~~demo~~ you do the rest!

- Initialize $S \leftarrow \{ s \}$ and $d[s] \leftarrow 0$.
- Repeatedly choose unexplored node $v \notin S$ which minimizes

$$\pi(v) = \min_{e = (u,v) \,:\, u \in S} d[u] + \ell_e$$

the length of a shortest path from $s$ to some node $u$ in explored part $S$, followed by a single edge $e = (u, v)$

add $v$ to $S$; set $d[v] \leftarrow \pi(v)$ and $pred[v] \leftarrow$ argmin.



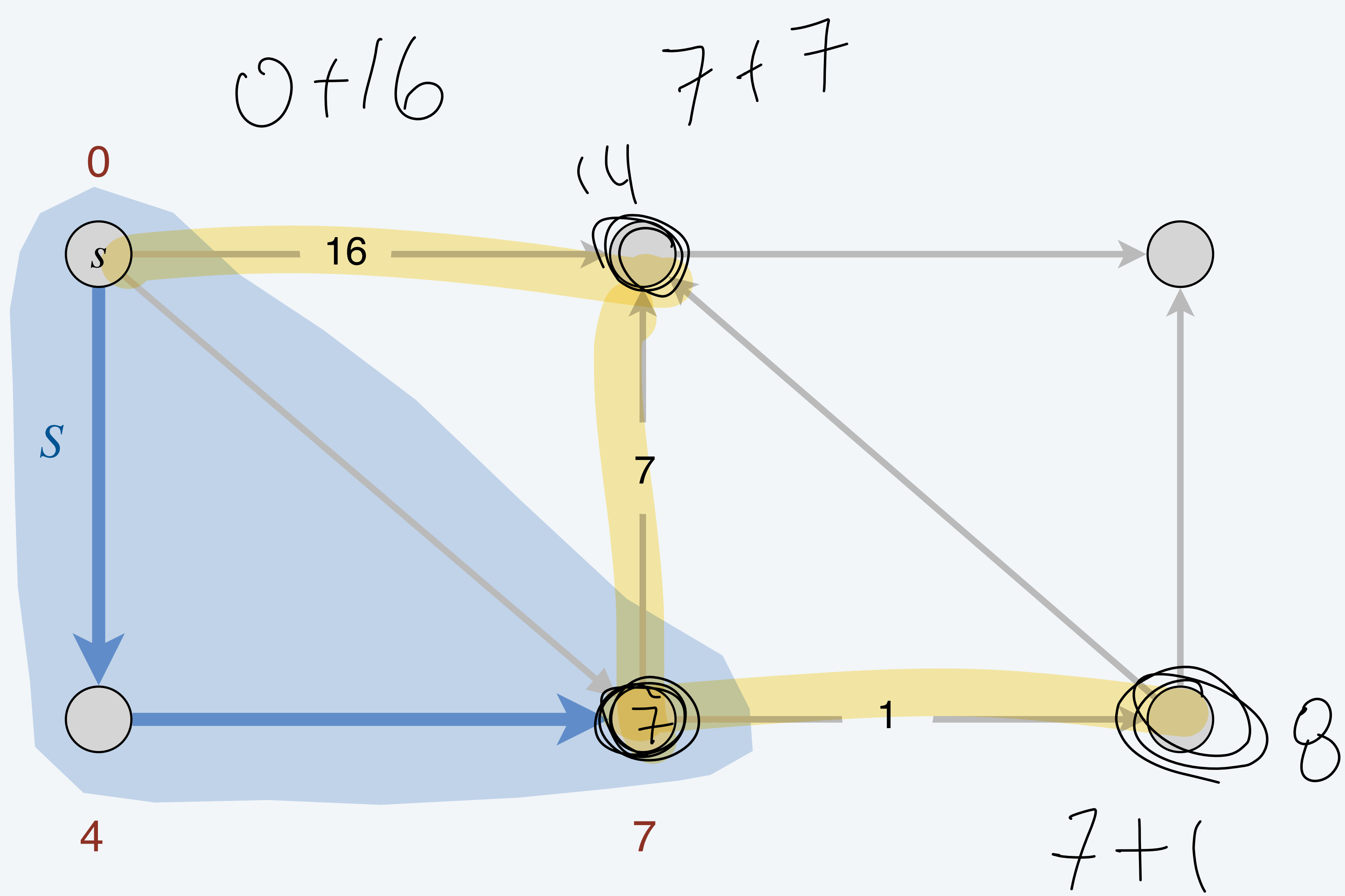
# Dijkstra's algorithm demo

- Initialize $S \leftarrow \{ s \}$ and $d[s] \leftarrow 0$.
- Repeatedly choose unexplored node $v \notin S$ which minimizes

$$\pi(v) = \min_{e = (u,v) \,:\, u \in S} d[u] + \ell_e$$

the length of a shortest path from $s$ to some node $u$ in explored part $S$, followed by a single edge $e = (u, v)$

add $v$ to $S$; set $d[v] \leftarrow \pi(v)$ and $pred[v] \leftarrow$ argmin.



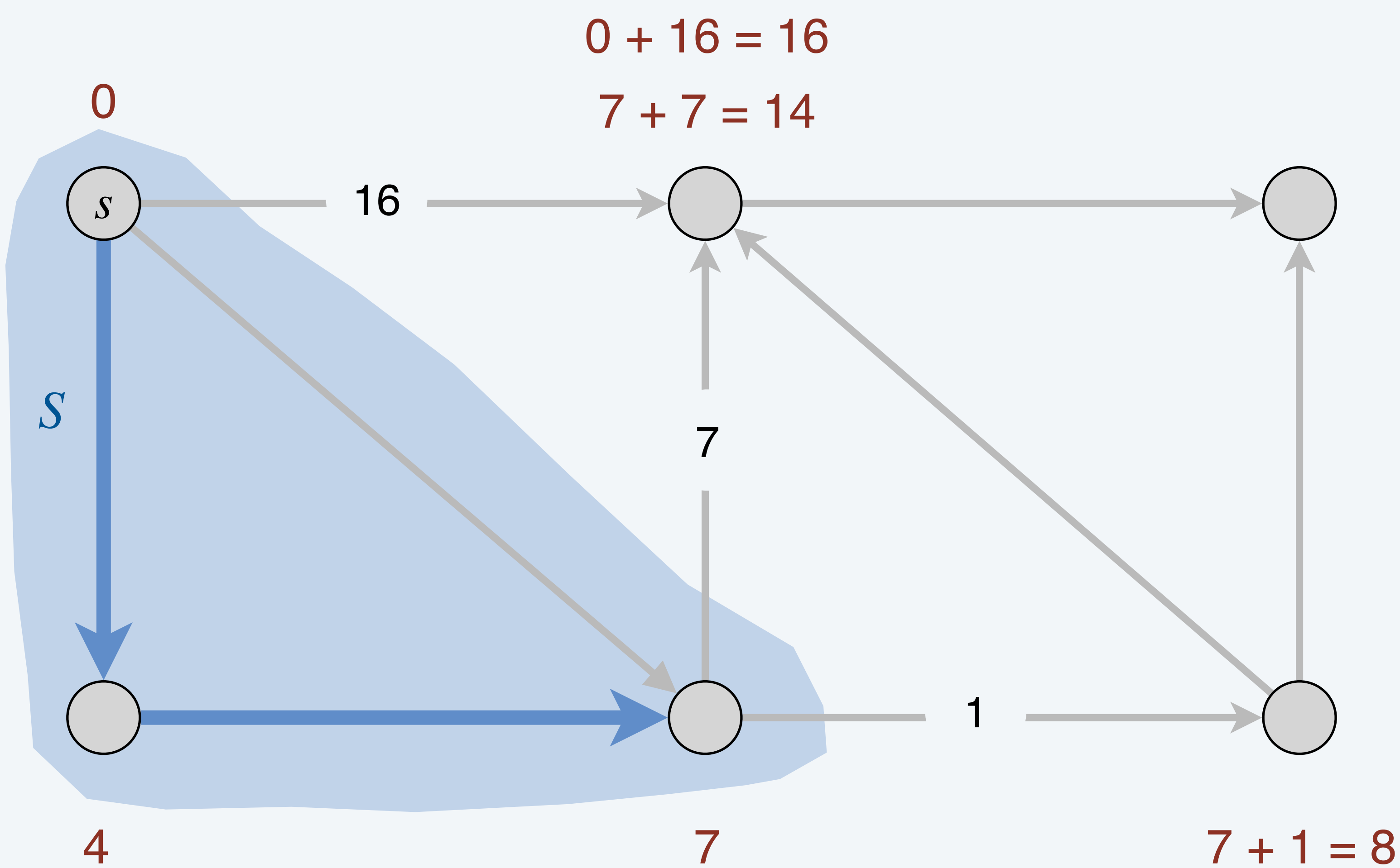
# Dijkstra's algorithm demo

- Initialize $S \leftarrow \{ s \}$ and $d[s] \leftarrow 0$.
- Repeatedly choose unexplored node $v \notin S$ which minimizes

$$\pi(v) \;=\; \min_{e\,=\,(u,v)\;:\;u \in S} d[u] + \ell_e$$

the length of a shortest path from $s$ to some node $u$ in explored part $S$, followed by a single edge $e = (u, v)$

add $v$ to $S$; set $d[v] \leftarrow \pi(v)$ and $pred[v] \leftarrow$ argmin.



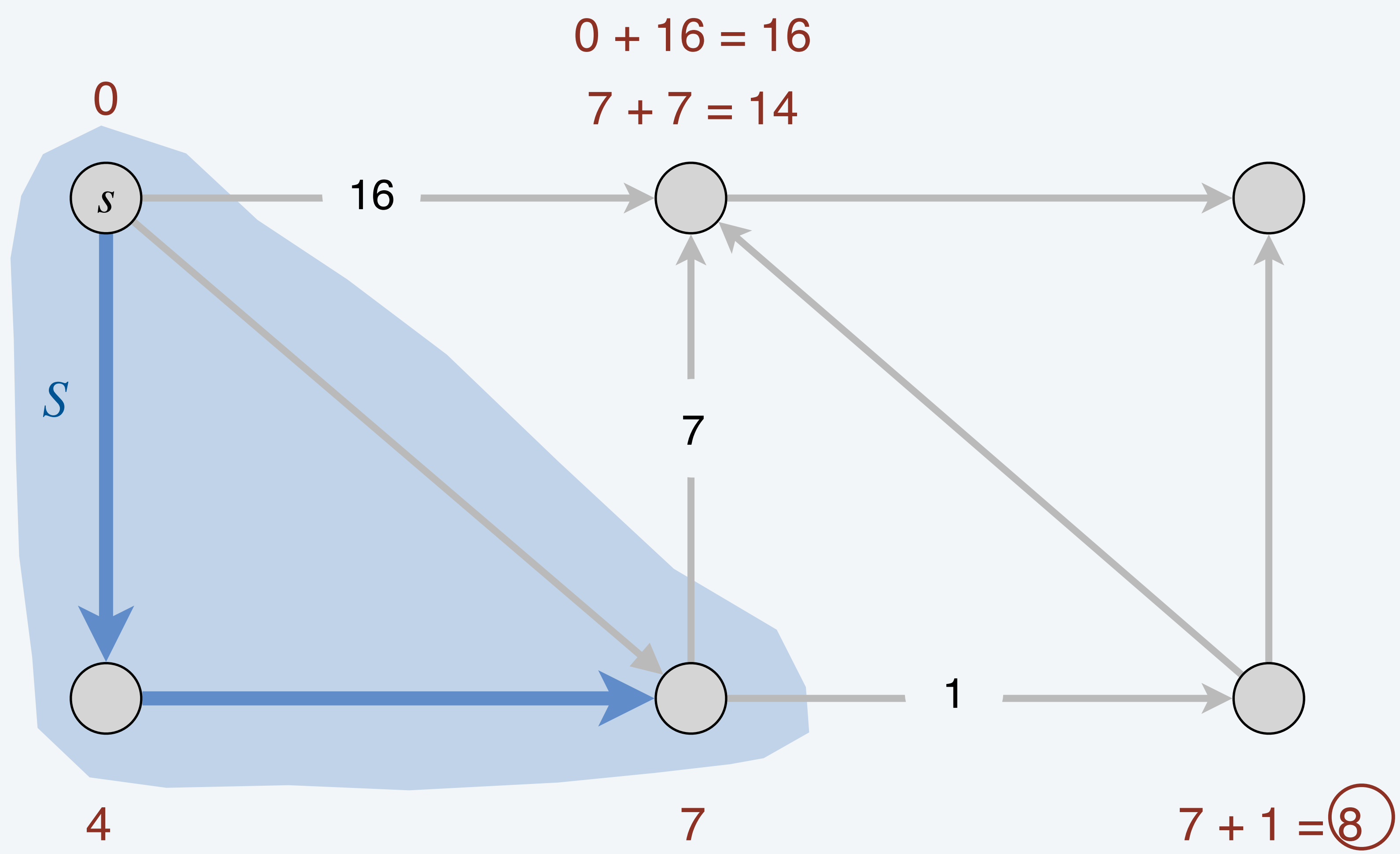
# Dijkstra's algorithm demo

- Initialize $S \leftarrow \{ s \}$ and $d[s] \leftarrow 0$.
- Repeatedly choose unexplored node $v \notin S$ which minimizes

$$\pi(v) = \min_{e = (u,v) \,:\, u \in S} d[u] + \ell_e$$

the length of a shortest path from $s$ to some node $u$ in explored part $S$, followed by a single edge $e = (u, v)$

add $v$ to $S$; set $d[v] \leftarrow \pi(v)$ and $pred[v] \leftarrow \text{argmin}$.



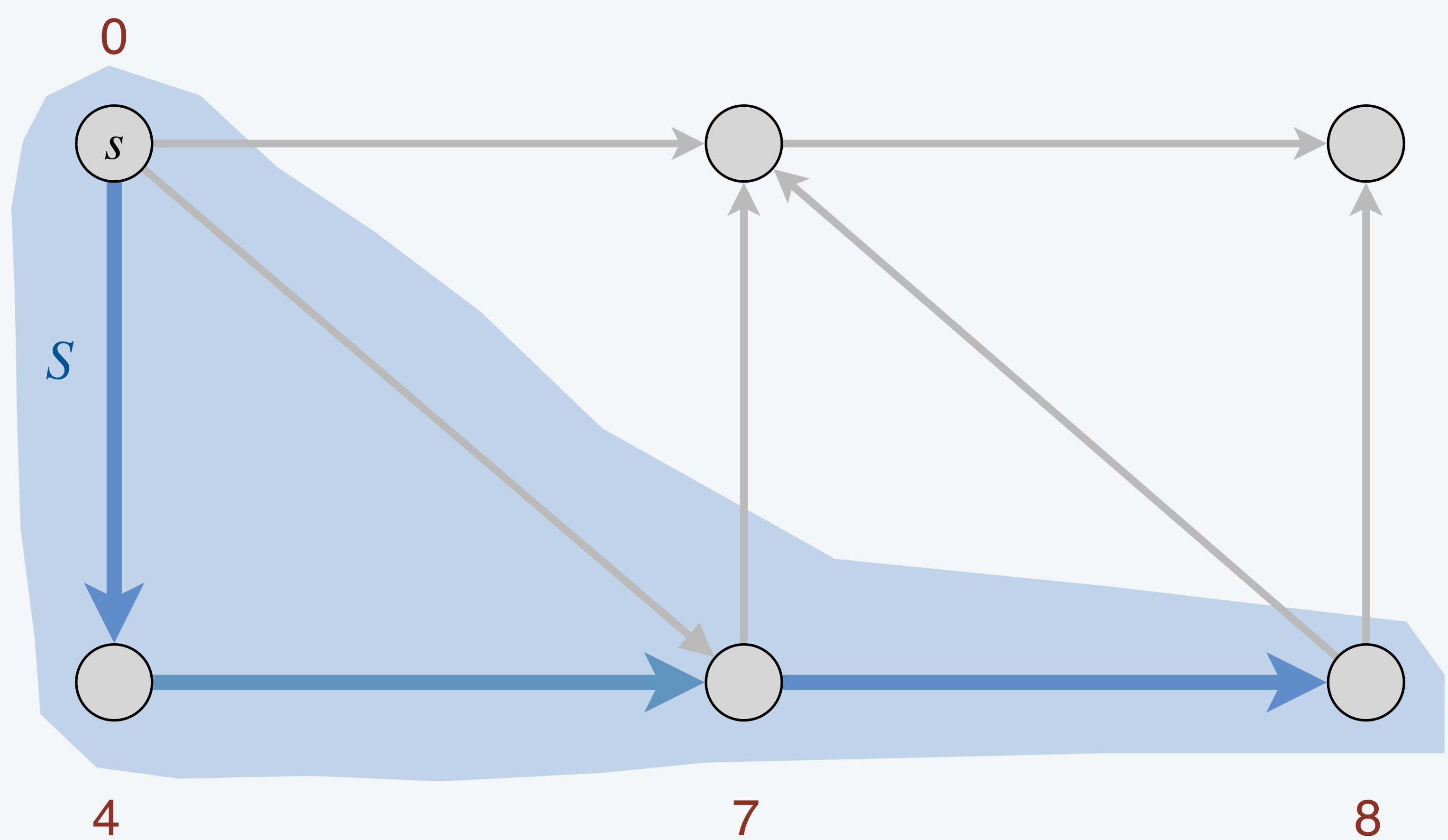
# Dijkstra's algorithm demo

- Initialize $S \leftarrow \{ s \}$ and $d[s] \leftarrow 0$.
- Repeatedly choose unexplored node $v \notin S$ which minimizes

$$\pi(v) = \min_{e = (u,v) \,:\, u \in S} d[u] + \ell_e$$

the length of a shortest path from $s$ to some node $u$ in explored part $S$, followed by a single edge $e = (u, v)$

add $v$ to $S$; set $d[v] \leftarrow \pi(v)$ and $pred[v] \leftarrow$ argmin.



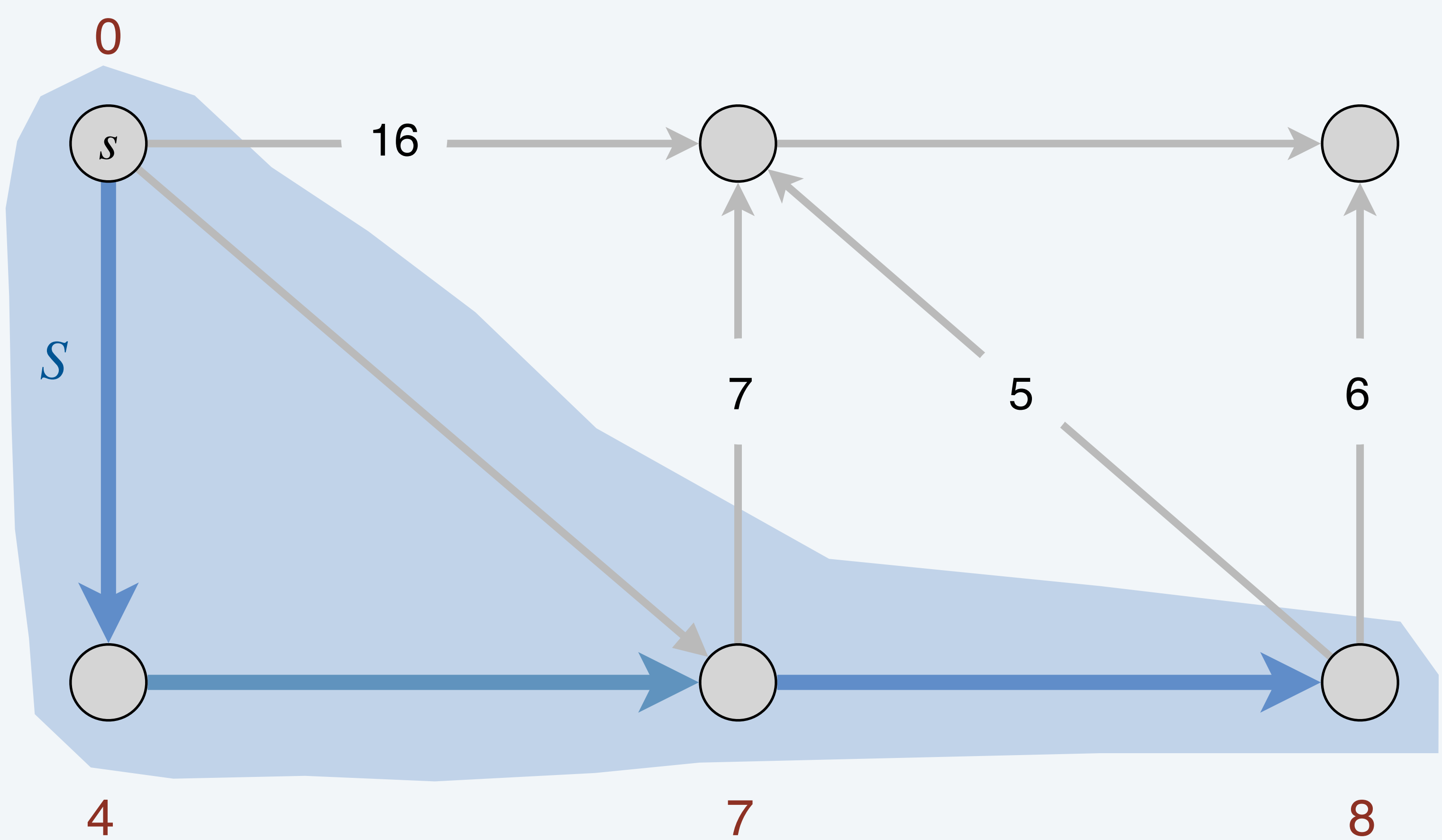
# Dijkstra's algorithm demo

- Initialize $S \leftarrow \{ s \}$ and $d[s] \leftarrow 0$.
- Repeatedly choose unexplored node $v \notin S$ which minimizes

$$\pi(v) = \min_{e = (u,v) \,:\, u \in S} d[u] + \ell_e$$

the length of a shortest path from $s$ to some node $u$ in explored part $S$, followed by a single edge $e = (u, v)$

add $v$ to $S$; set $d[v] \leftarrow \pi(v)$ and $pred[v] \leftarrow$ argmin.



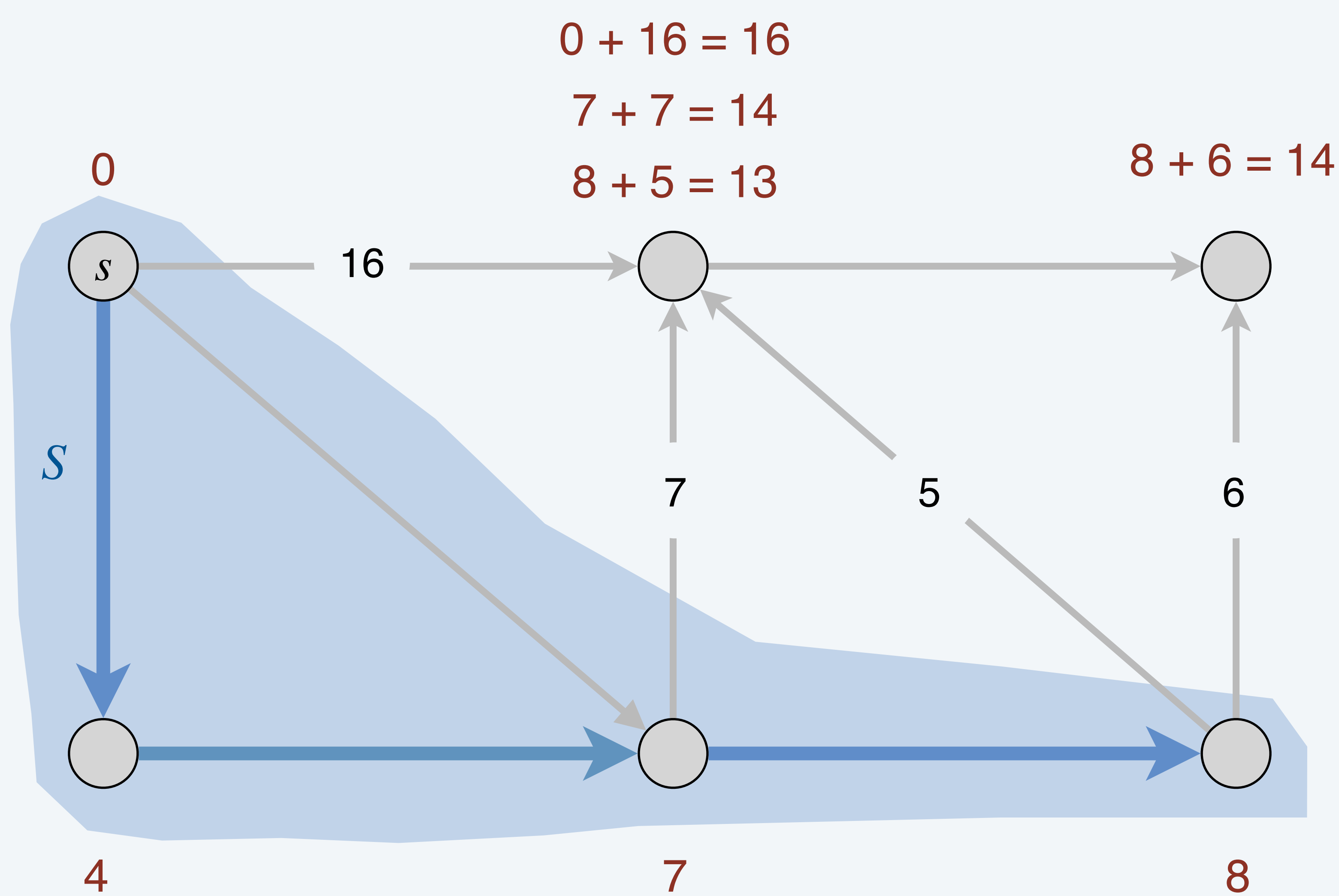
# Dijkstra's algorithm demo

- Initialize $S \leftarrow \{ s \}$ and $d[s] \leftarrow 0$.
- Repeatedly choose unexplored node $v \notin S$ which minimizes

$$\pi(v) = \min_{e = (u,v) \,:\, u \in S} d[u] + \ell_e$$

the length of a shortest path from $s$ to some node $u$ in explored part $S$, followed by a single edge $e = (u, v)$

add $v$ to $S$; set $d[v] \leftarrow \pi(v)$ and $pred[v] \leftarrow$ argmin.



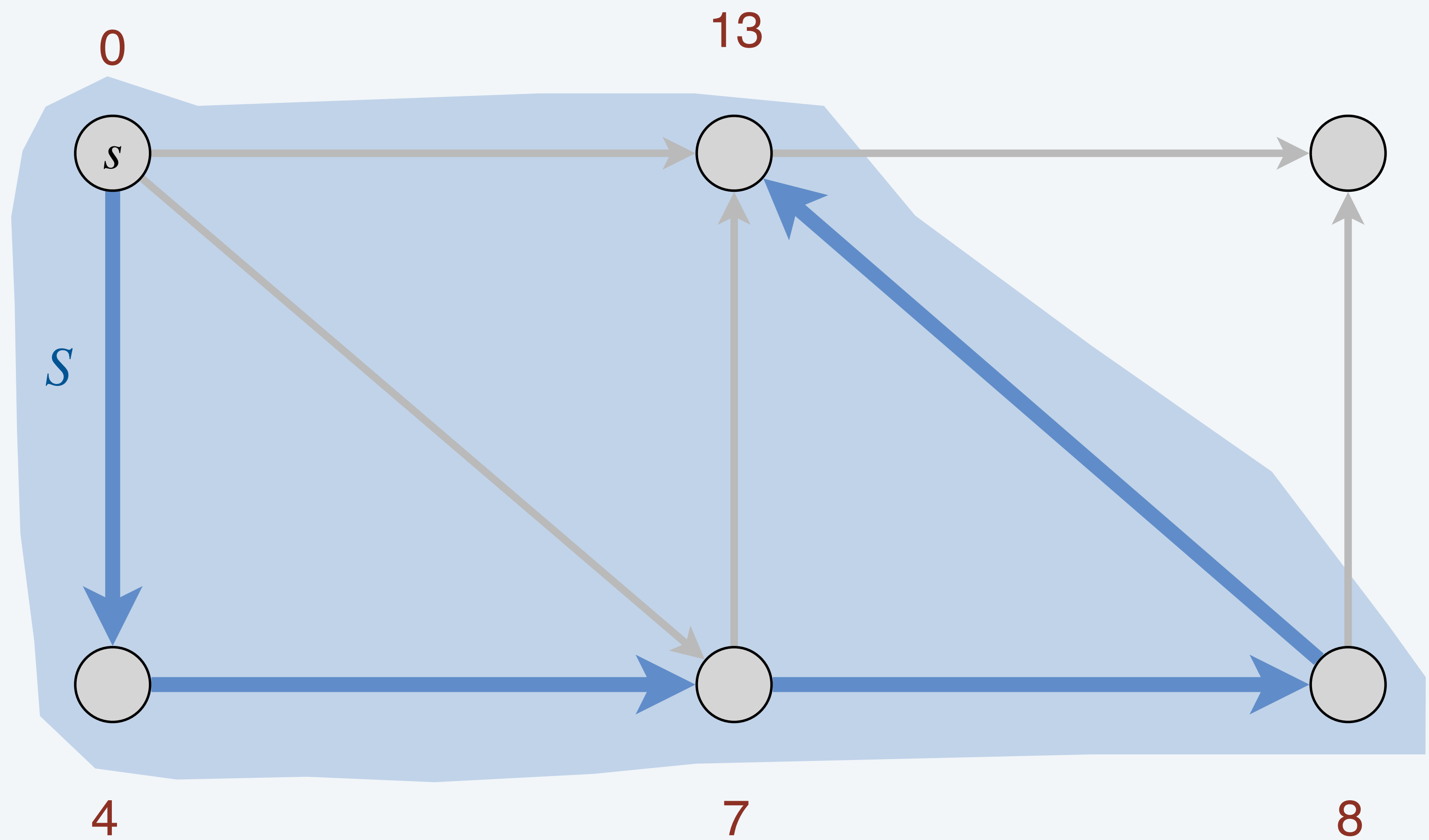
# Dijkstra's algorithm demo

- Initialize $S \leftarrow \{ s \}$ and $d[s] \leftarrow 0$.
- Repeatedly choose unexplored node $v \notin S$ which minimizes

$$\pi(v) \;=\; \min_{e\,=\,(u,v)\;:\;u \in S} d[u] + \ell_e$$

the length of a shortest path from $s$ to some node $u$ in explored part $S$, followed by a single edge $e = (u, v)$

add $v$ to $S$; set $d[v] \leftarrow \pi(v)$ and $pred[v] \leftarrow$ argmin.



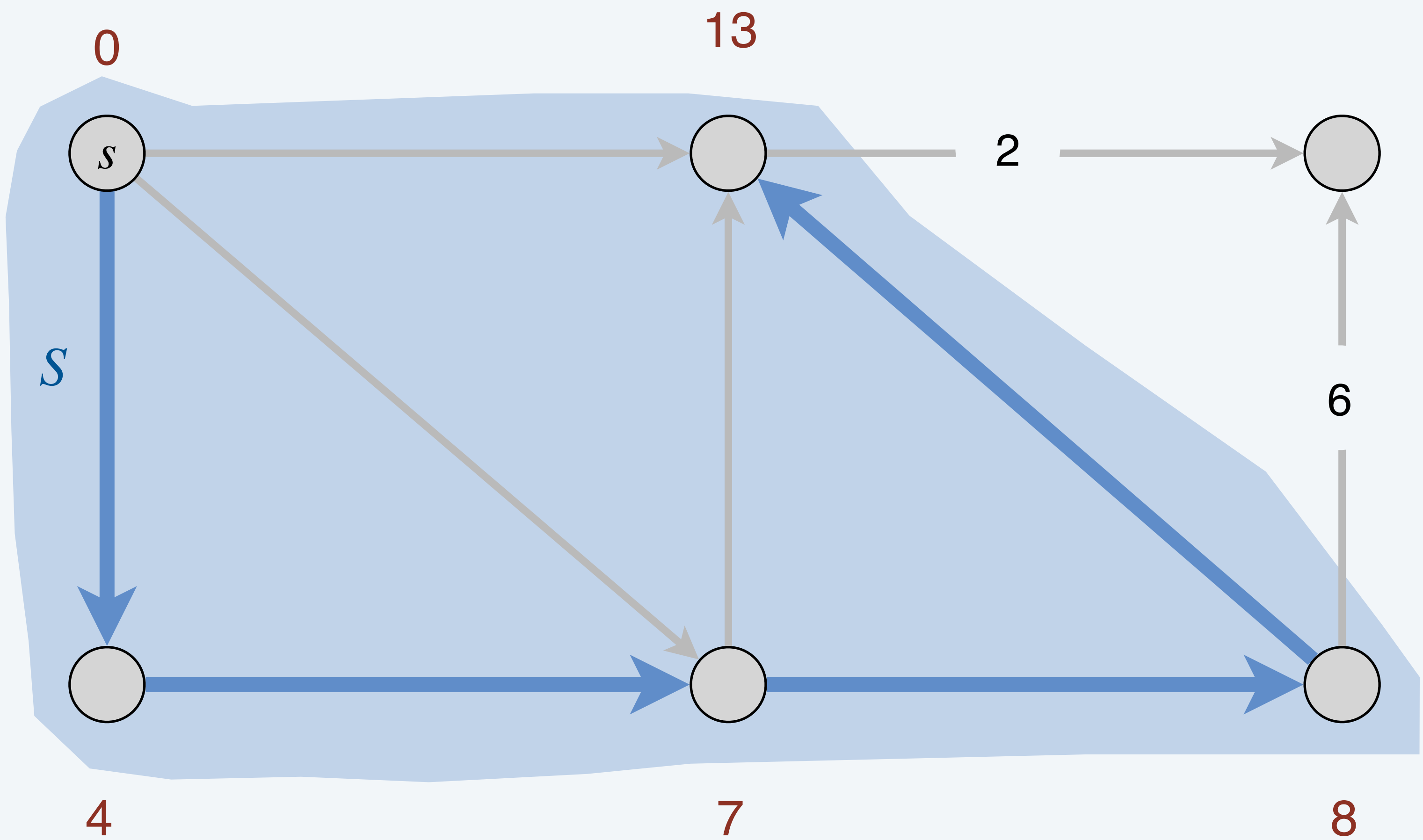
# Dijkstra's algorithm demo

- Initialize $S \leftarrow \{ s \}$ and $d[s] \leftarrow 0$.
- Repeatedly choose unexplored node $v \notin S$ which minimizes

$$\pi(v) \;=\; \min_{e\,=\,(u,v)\;:\;u\in S} d[u] + \ell_e$$

the length of a shortest path from $s$ to some node $u$ in explored part $S$, followed by a single edge $e = (u, v)$

add $v$ to $S$; set $d[v] \leftarrow \pi(v)$ and $pred[v] \leftarrow$ argmin.



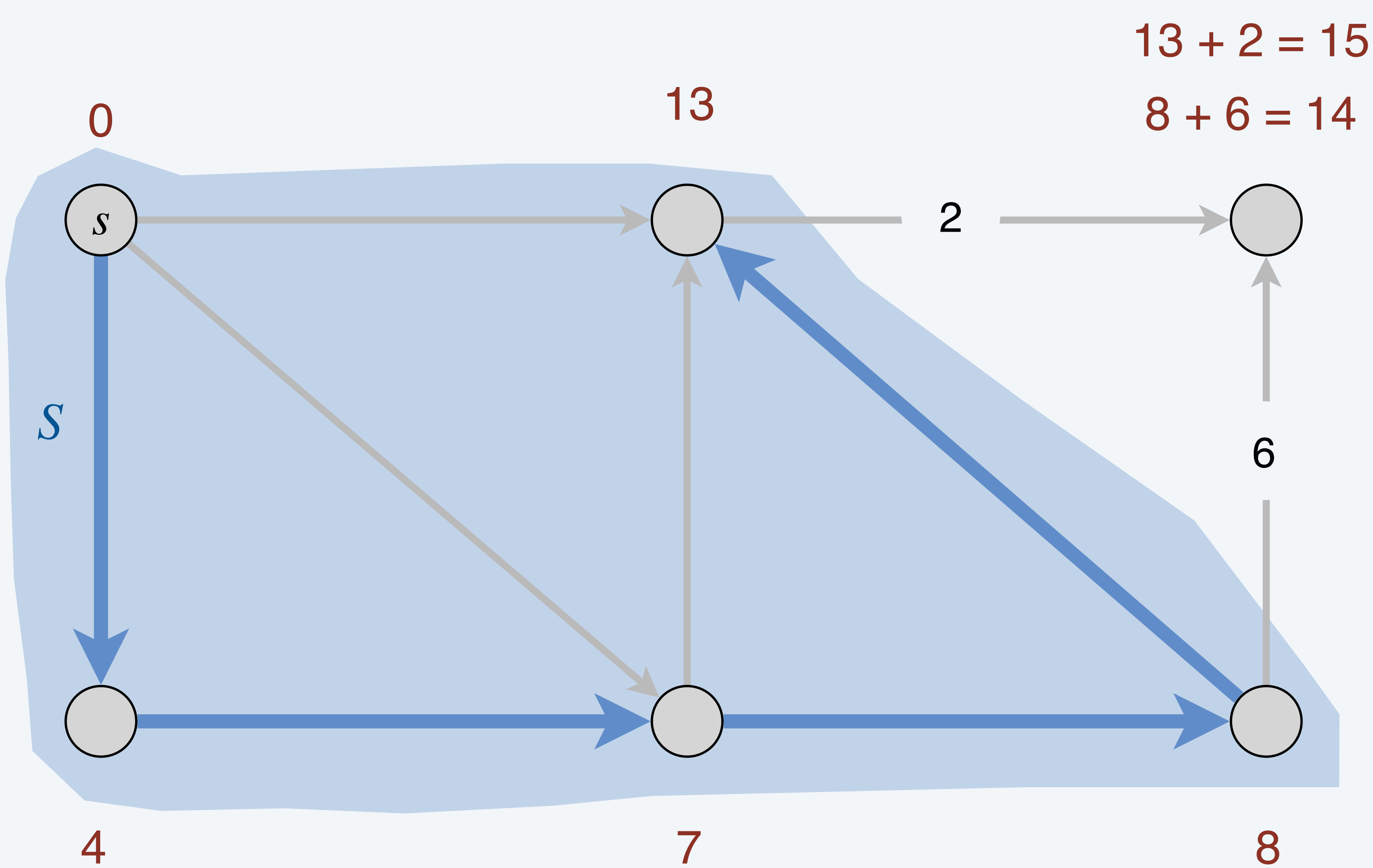
# Dijkstra's algorithm demo

- Initialize $S \leftarrow \{ s \}$ and $d[s] \leftarrow 0$.
- Repeatedly choose unexplored node $v \notin S$ which minimizes

$$\pi(v) = \min_{e = (u,v) \,:\, u \in S} d[u] + \ell_e$$

add $v$ to $S$; set $d[v] \leftarrow \pi(v)$ and $pred[v] \leftarrow$ argmin.

the length of a shortest path from $s$ to some node $u$ in explored part $S$, followed by a single edge $e = (u, v)$



# Dijkstra's algorithm demo

- Initialize $S \leftarrow \{ s \}$ and $d[s] \leftarrow 0$.
- Repeatedly choose unexplored node $v \notin S$ which minimizes

$$\pi(v) \;=\; \min_{e\,=\,(u,v)\;:\;u \in S} \; d[u] + \ell_e$$

add $v$ to $S$; set $d[v] \leftarrow \pi(v)$ and $pred[v] \leftarrow$ argmin.