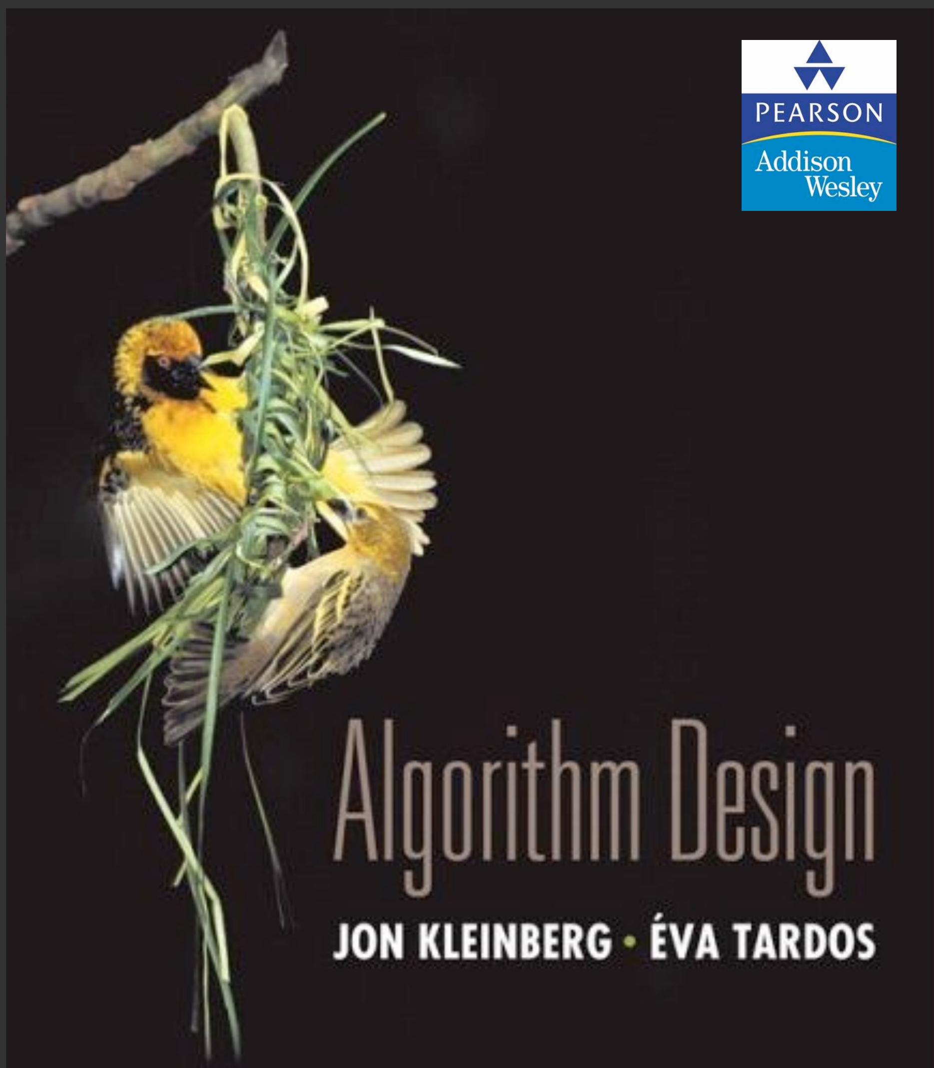


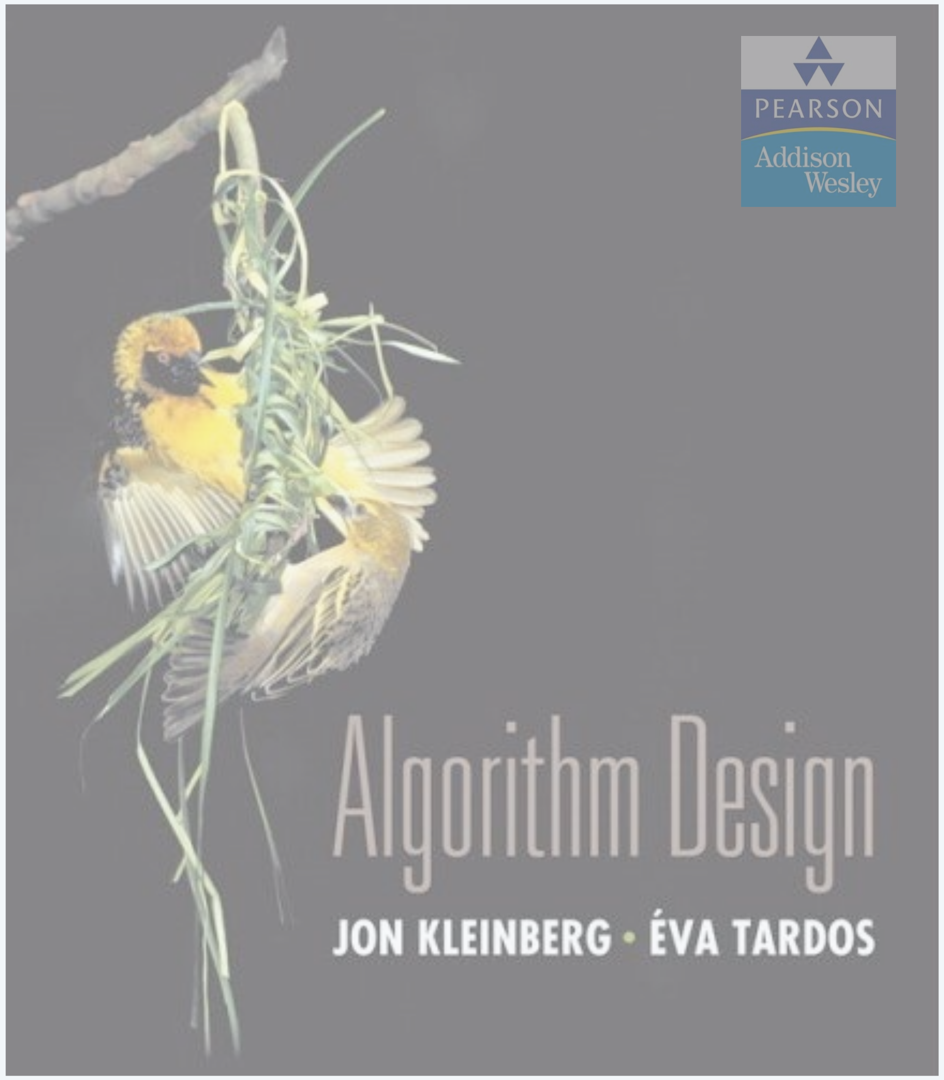
4. Greedy Algorithms II

- *Dijkstra's algorithm demo*
- *Dijkstra's algorithm demo (efficient implementation)*

Lecture slides by Kevin Wayne

http://www.cs.princeton.edu/~wayne/kleinberg-tardos

Last updated on 11/15/17 10:10 AM



4. Greedy Algorithms II

- *Dijkstra's algorithm demo*
- *Dijkstra's algorithm demo (efficient implementation)*

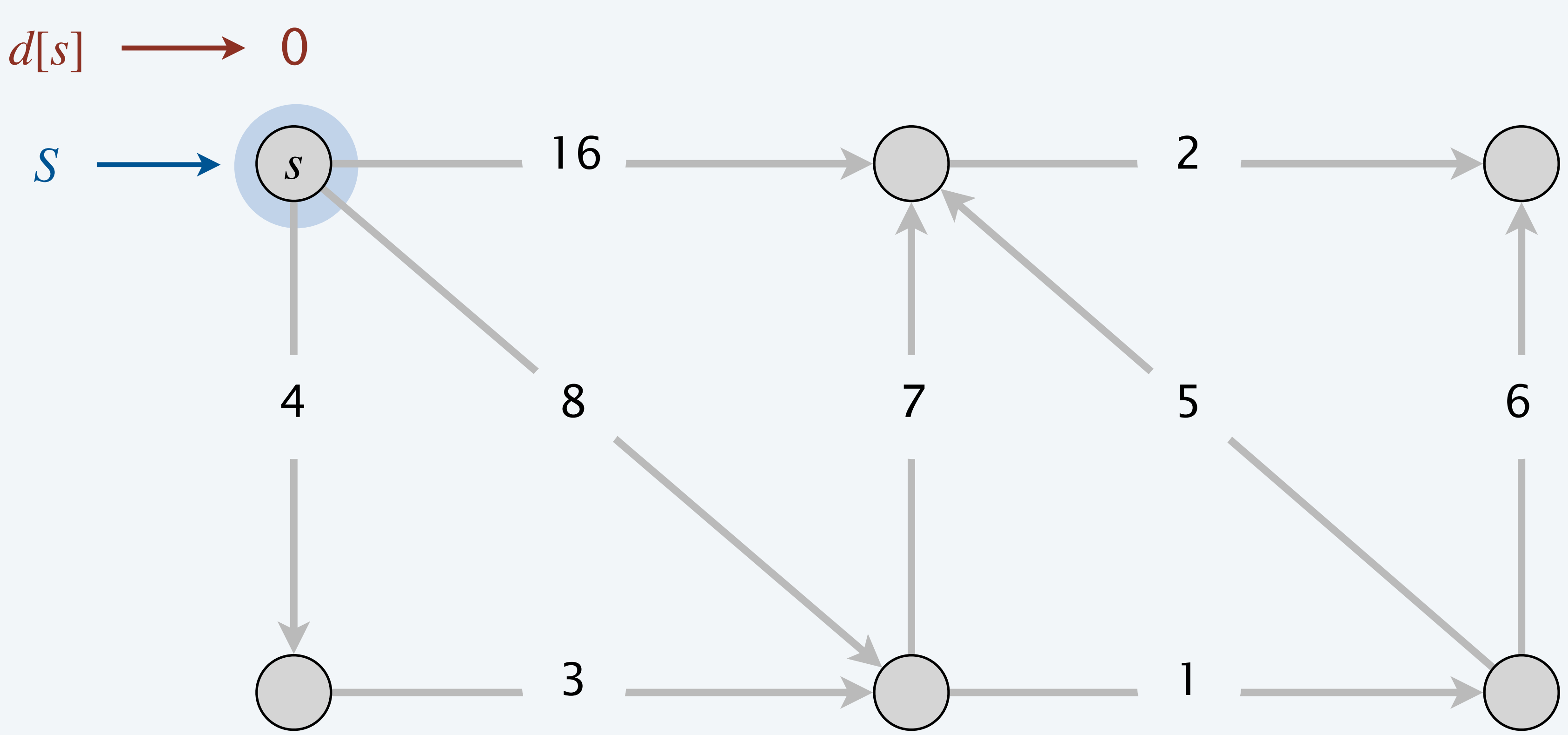
Dijkstra's algorithm demo

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

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

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 $pred[v] \leftarrow \text{argmin}$.



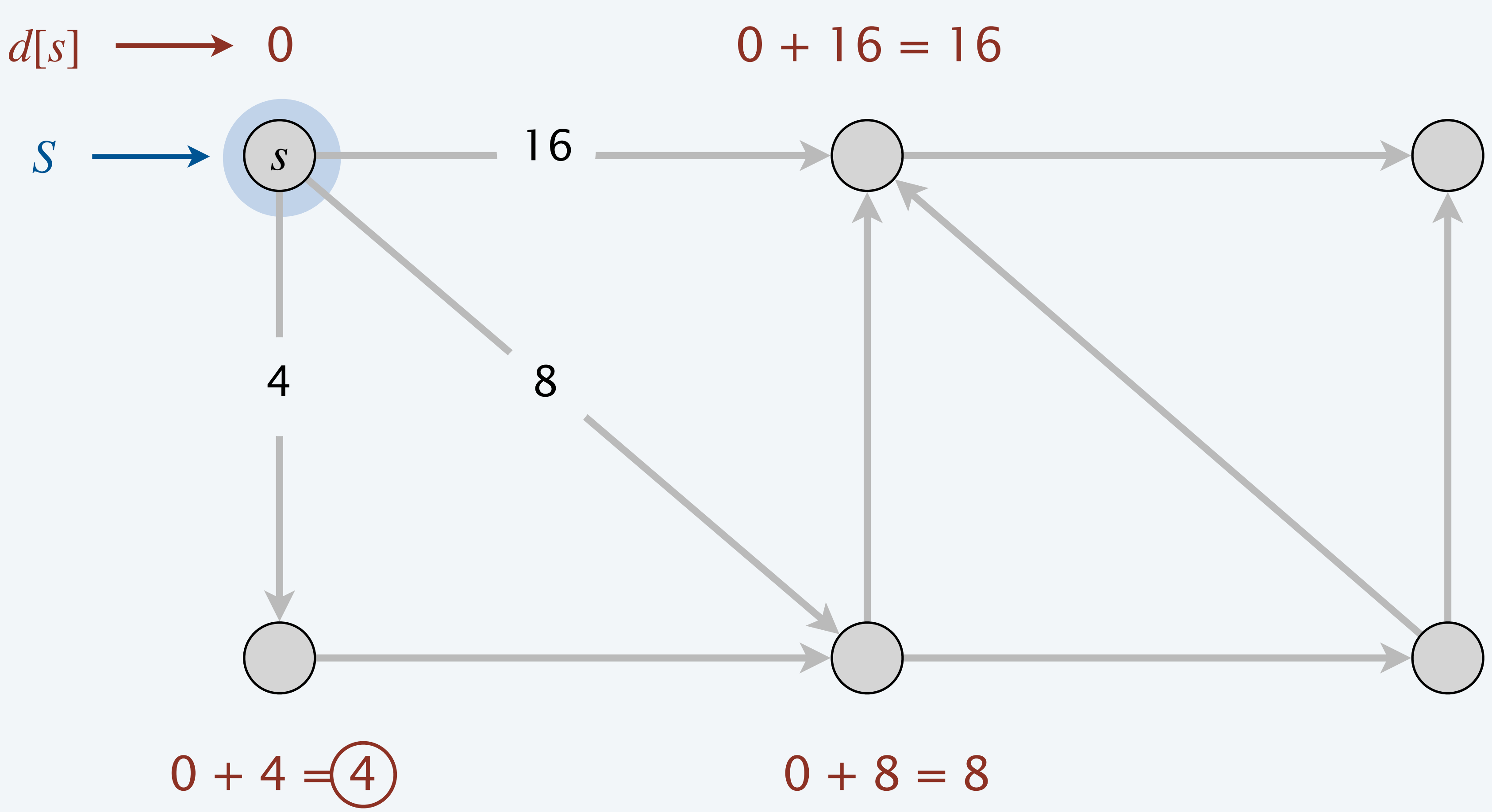
Dijkstra's algorithm demo

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

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

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.



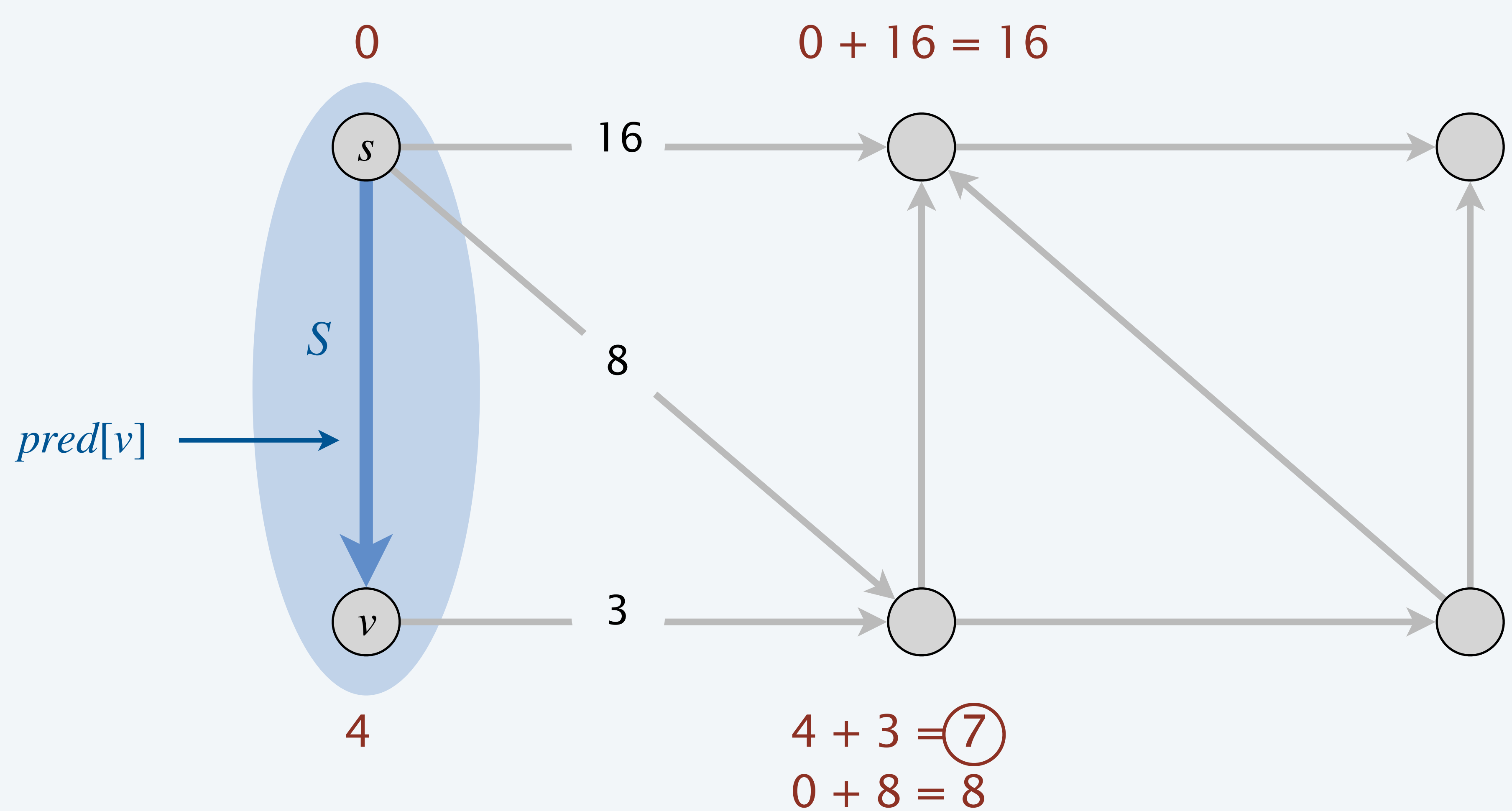
Dijkstra's algorithm demo

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

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

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.



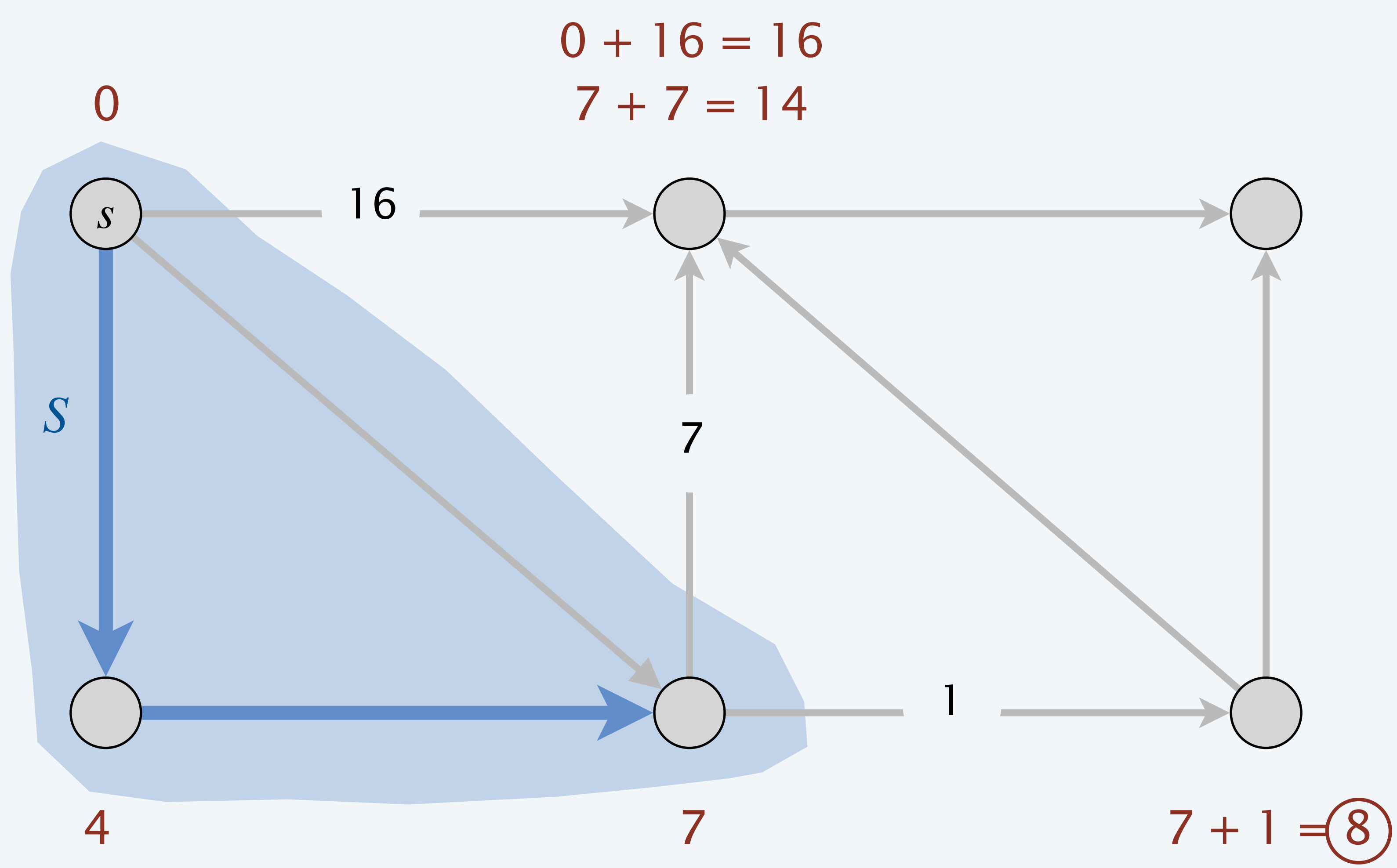
Dijkstra's algorithm demo

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

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

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.



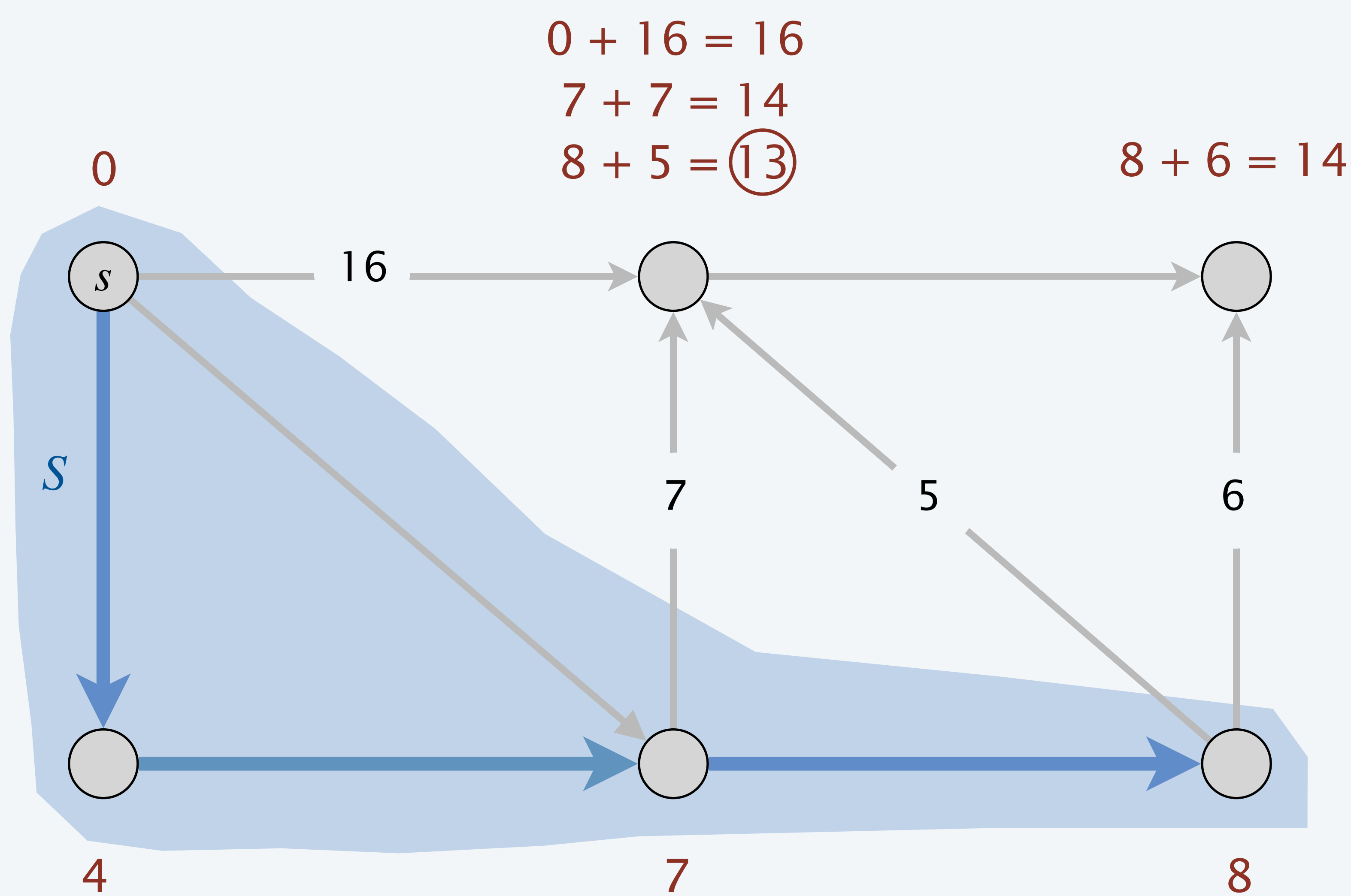
Dijkstra's algorithm demo

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

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

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.



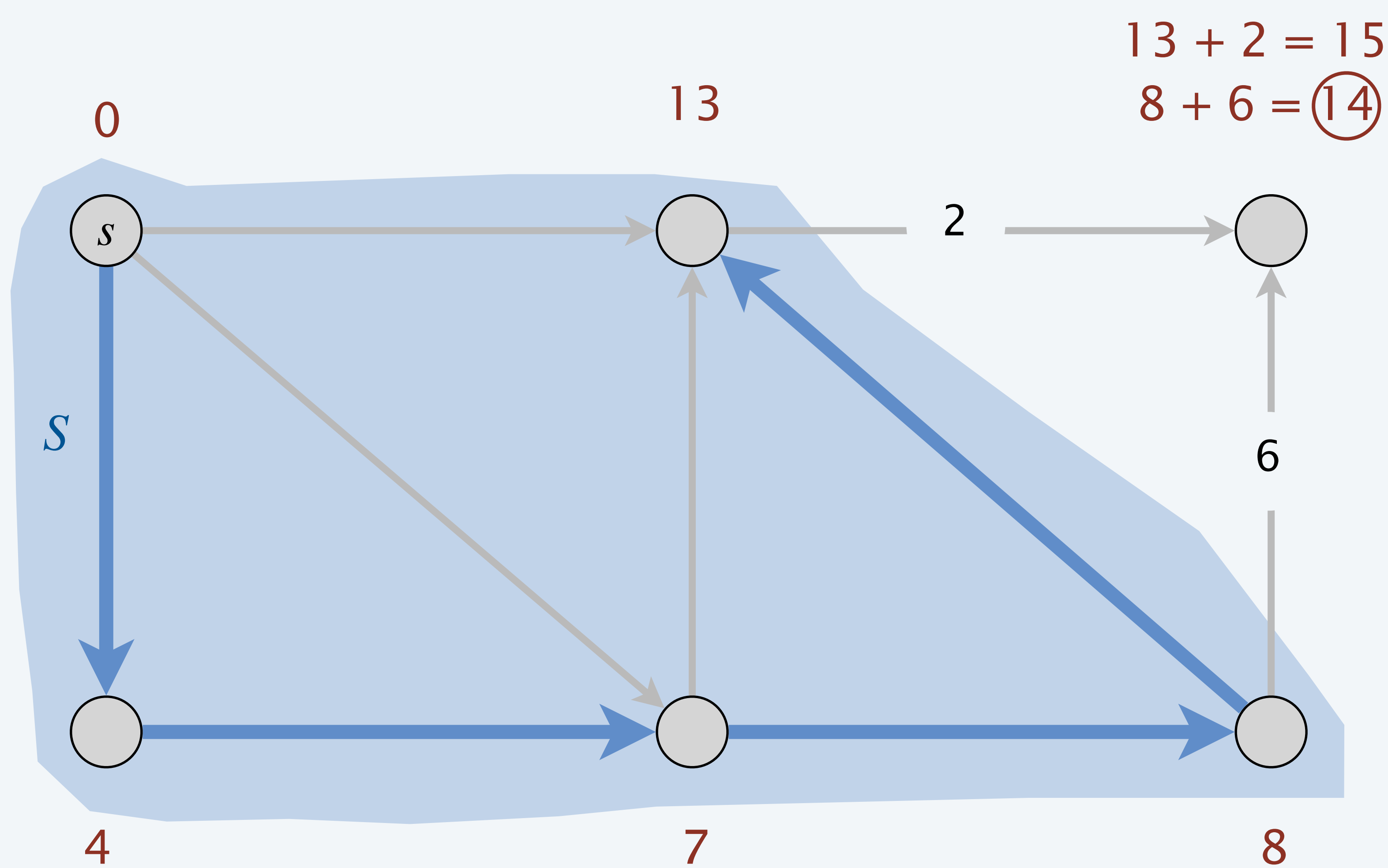
Dijkstra's algorithm demo

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

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

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.

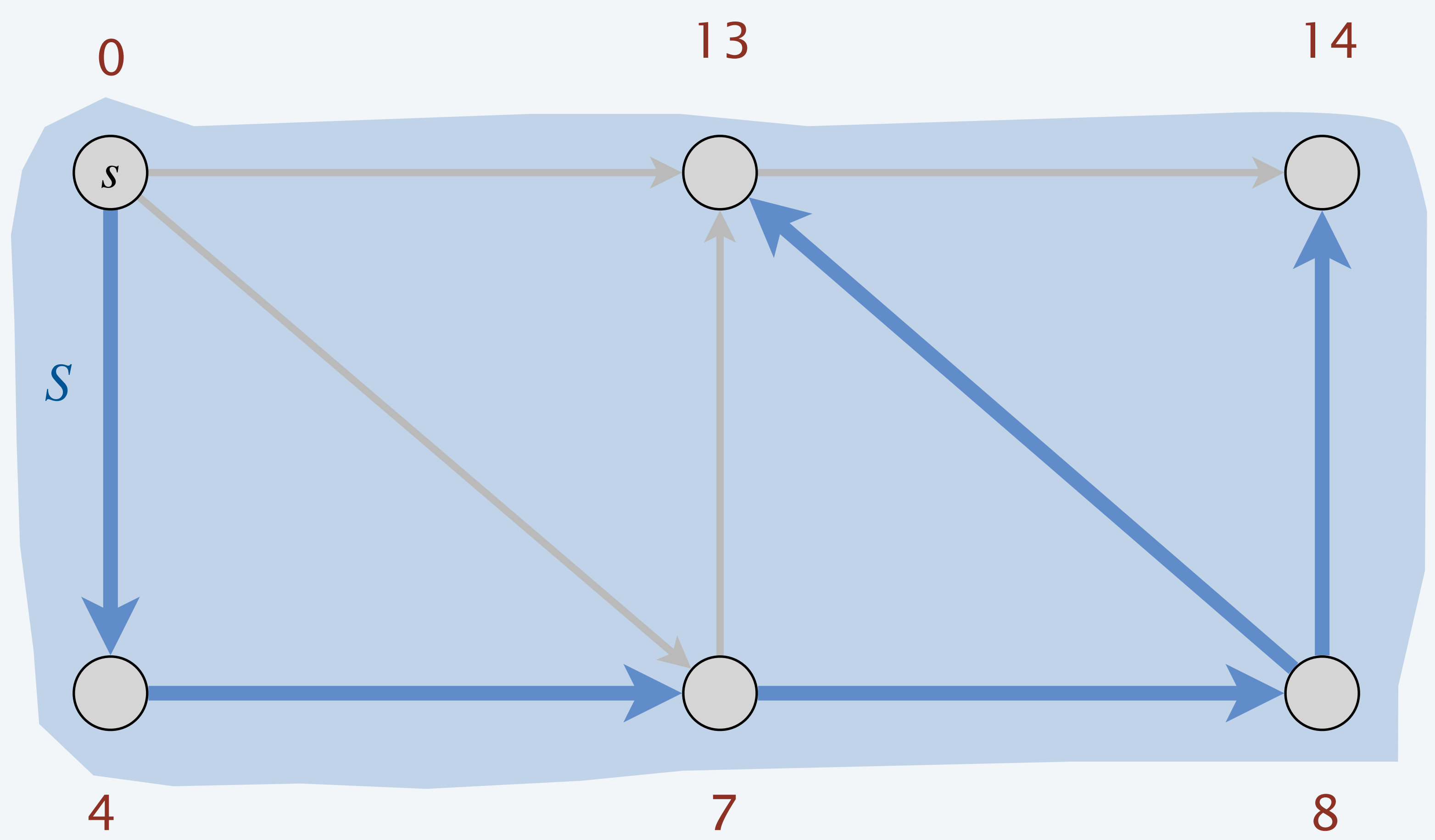


Dijkstra's algorithm demo

- 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 $pred[v] \leftarrow \text{argmin}$.

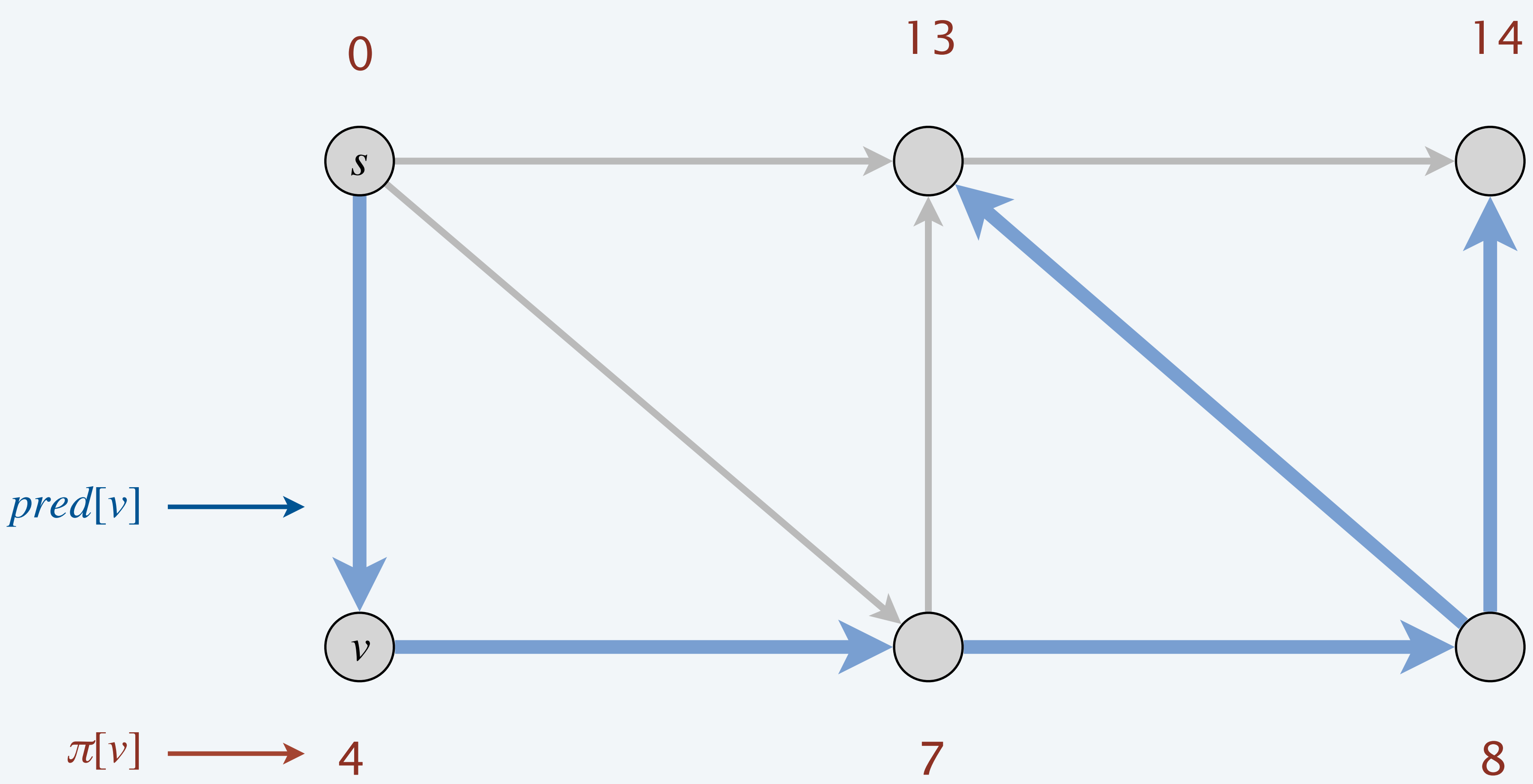


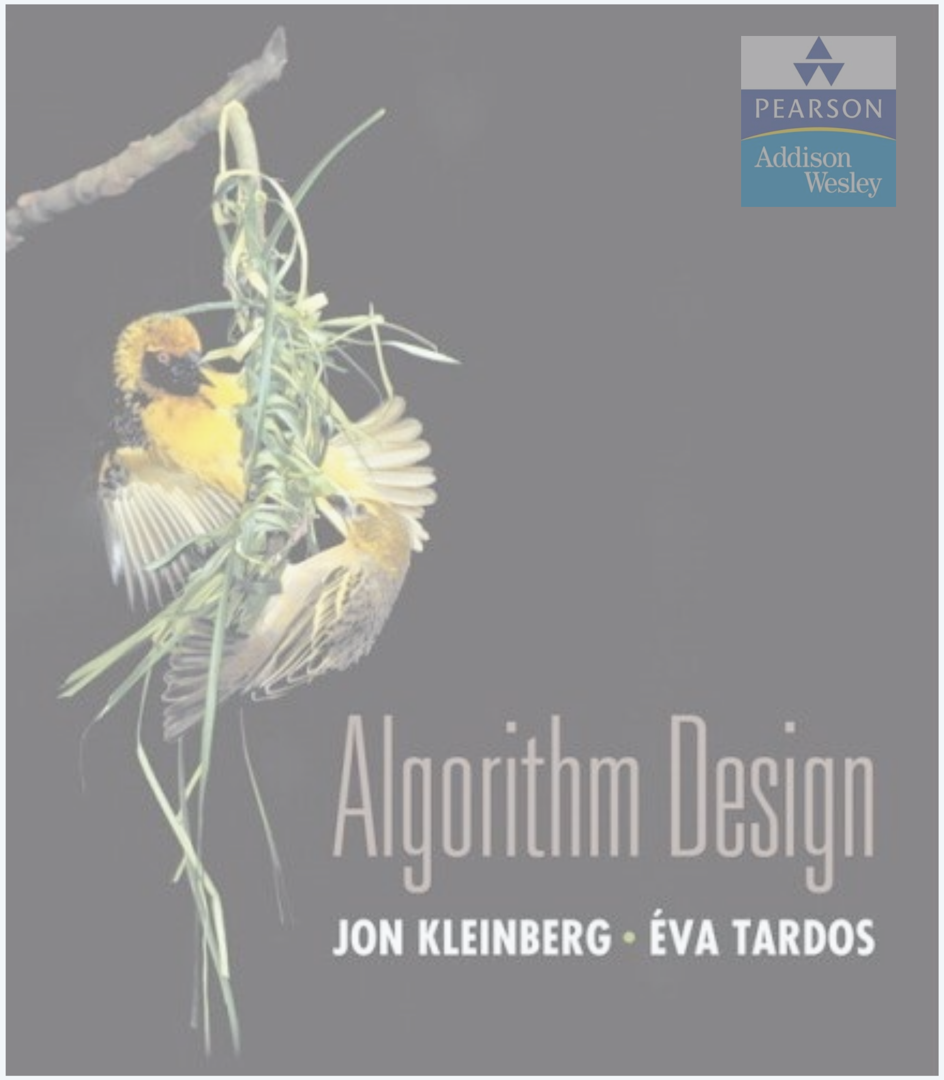
Dijkstra's algorithm demo

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

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

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





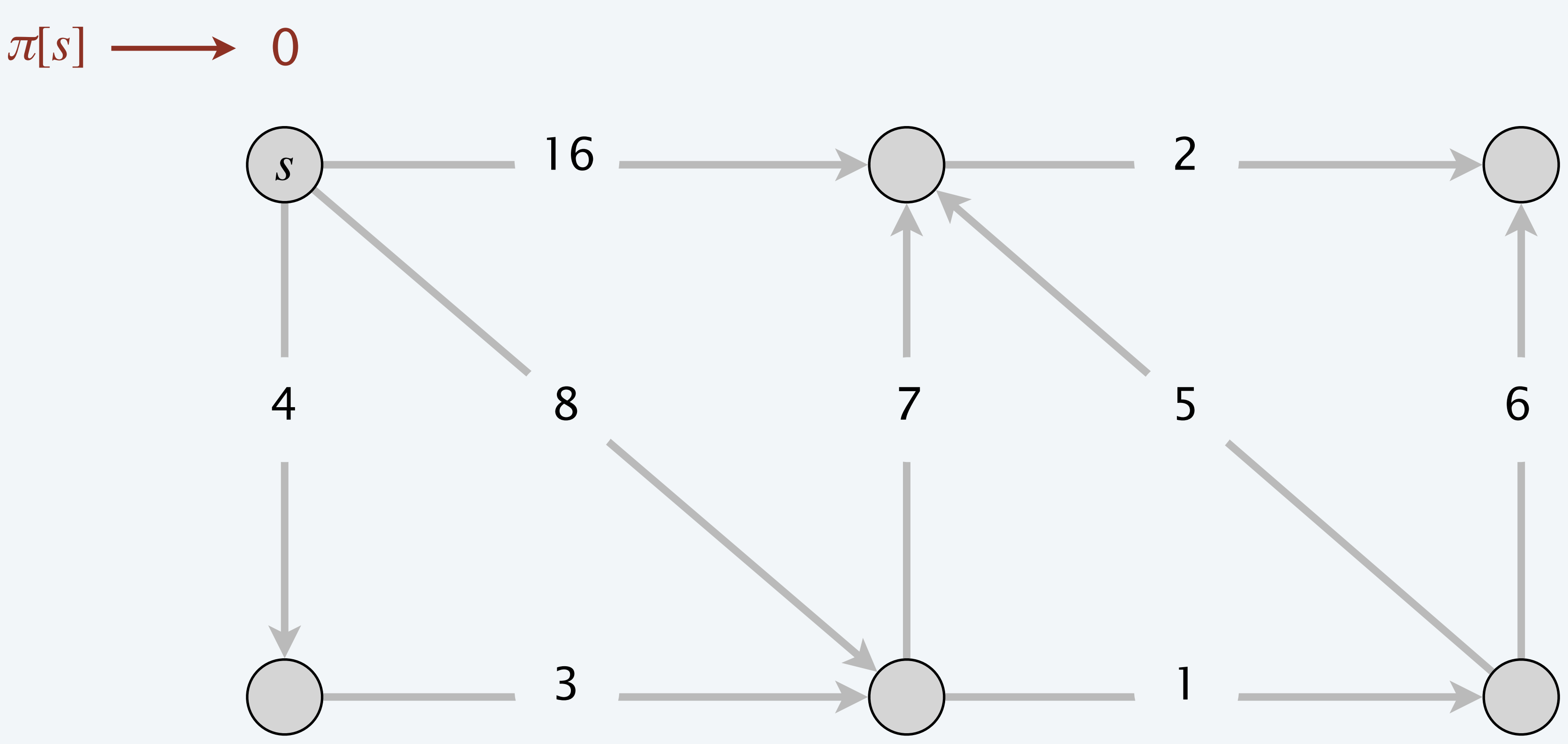
4. Greedy Algorithms II

- *Dijkstra's algorithm demo*
- *Dijkstra's algorithm demo (efficient implementation)*

Dijkstra's algorithm demo (efficient implementation)

Initialization.

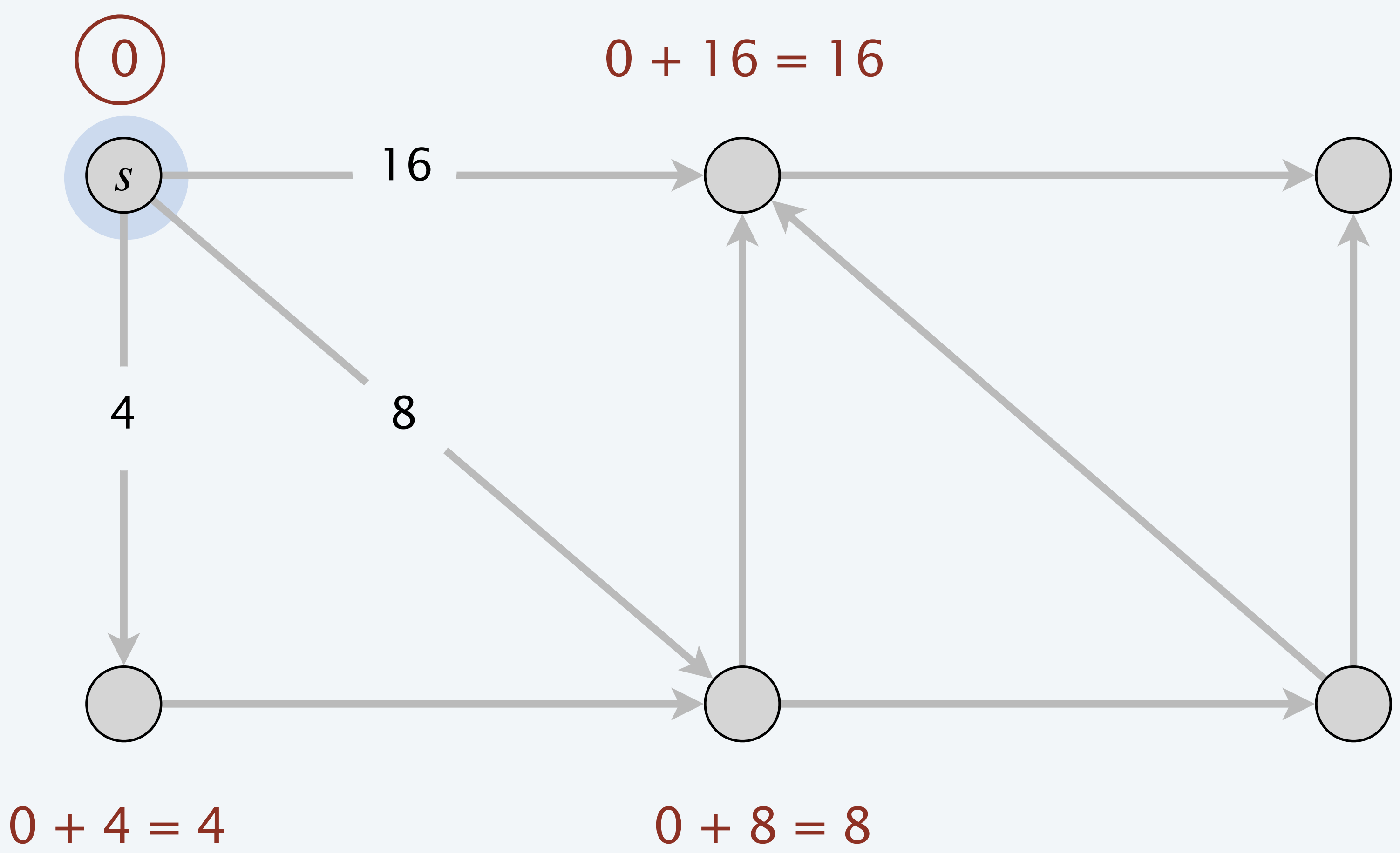
- For all $v \neq s$: $\pi[v] \leftarrow \infty$.
- For all $v \neq s$: $pred[v] \leftarrow null$.
- $S \leftarrow \varnothing$ and $\pi[s] \leftarrow 0$.



Dijkstra's algorithm demo (efficient implementation)

Basic step. Choose unexplored node $u \notin S$ with minimum $\pi[u]$.

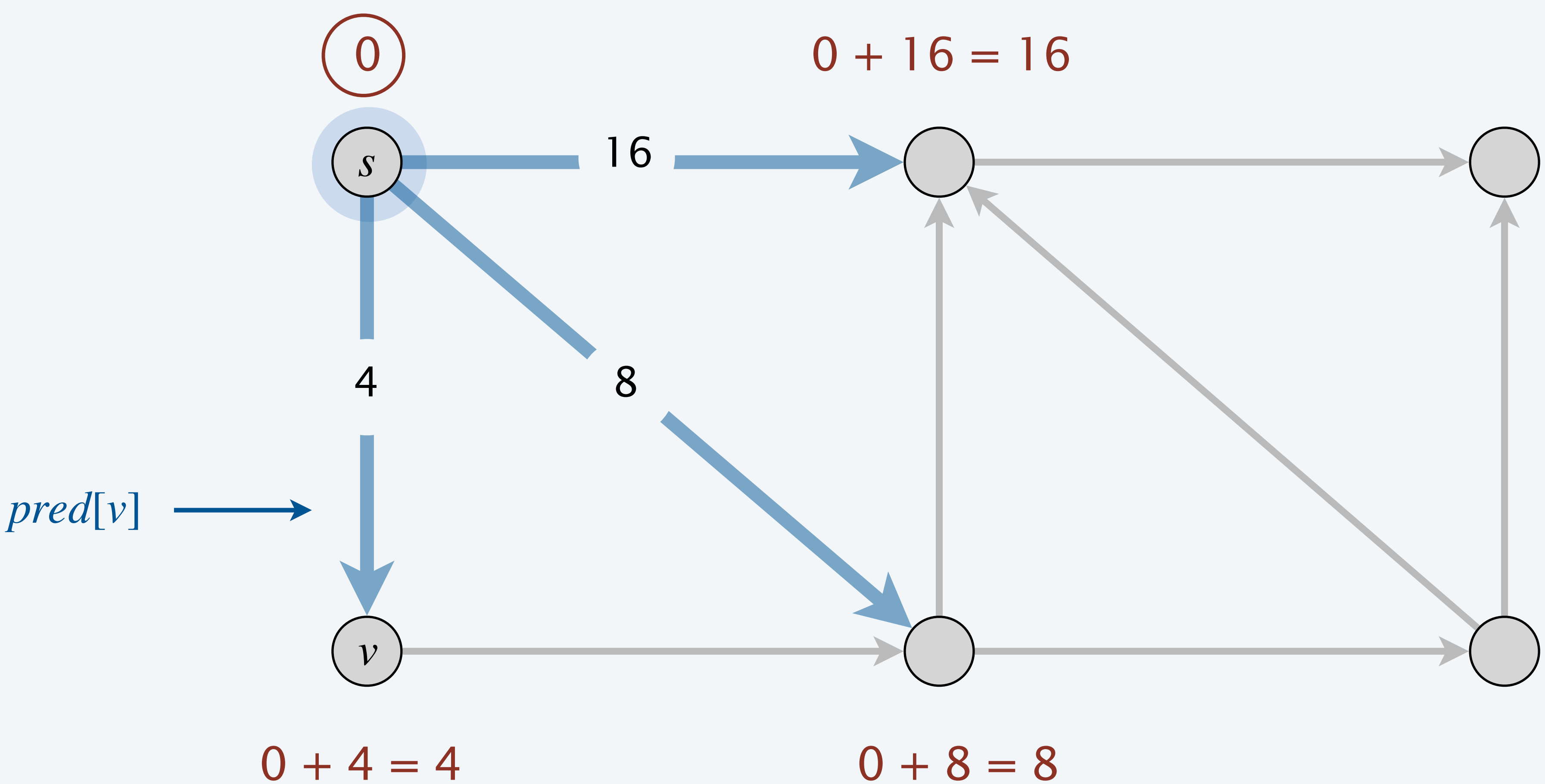
- Add u to S.
- For each edge $e = (u, v)$ leaving u, if $\pi[v] > \pi[u] + \ell_e$ then:
  - $\pi[v] \leftarrow \pi[u] + \ell_e$
  - $pred[v] \leftarrow e$



Dijkstra's algorithm demo (efficient implementation)

Basic step. Choose unexplored node $u \notin S$ with minimum $\pi[u]$.

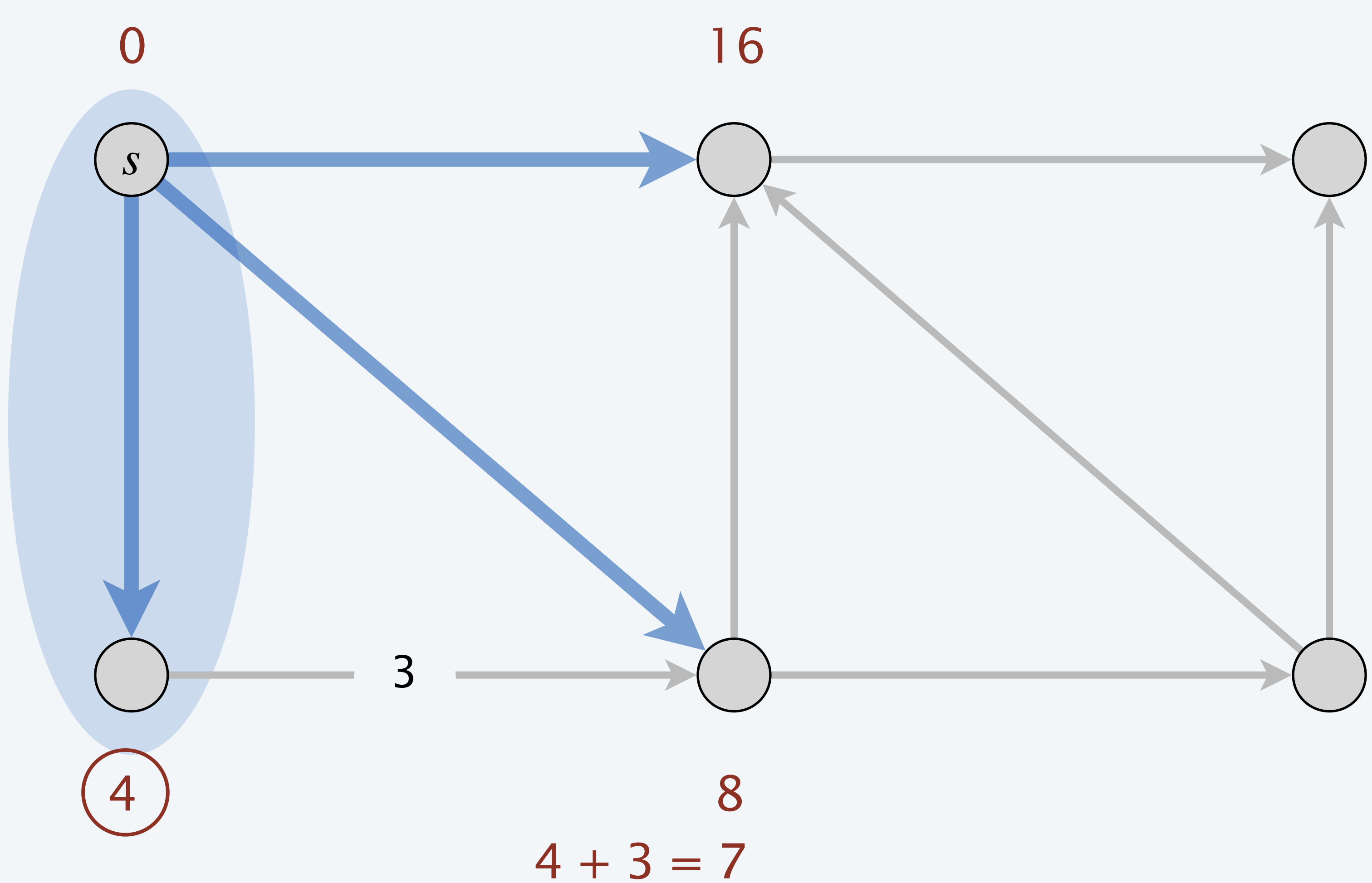
- Add u to S.
- For each edge $e = (u, v)$ leaving u, if $\pi[v] > \pi[u] + \ell_e$ then:
  - $\pi[v] \leftarrow \pi[u] + \ell_e$
  - $pred[v] \leftarrow e$



Dijkstra's algorithm demo (efficient implementation)

Basic step. Choose unexplored node $u \notin S$ with minimum $\pi[u]$.

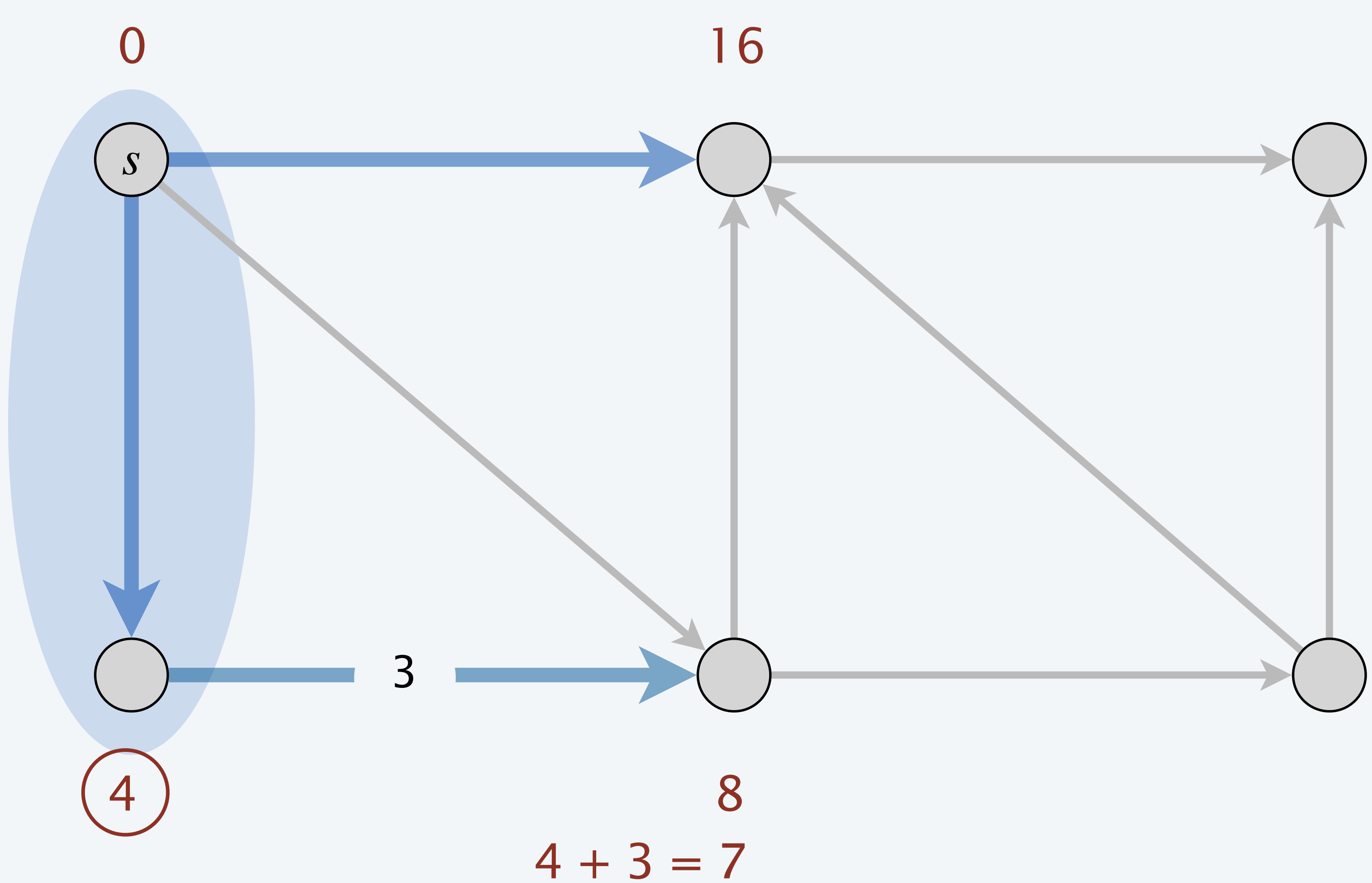
- Add u to S.
- For each edge $e = (u, v)$ leaving u, if $\pi[v] > \pi[u] + \ell_e$ then:
  - $\pi[v] \leftarrow \pi[u] + \ell_e$
  - $pred[v] \leftarrow e$



Dijkstra's algorithm demo (efficient implementation)

Basic step. Choose unexplored node $u \notin S$ with minimum $\pi[u]$.

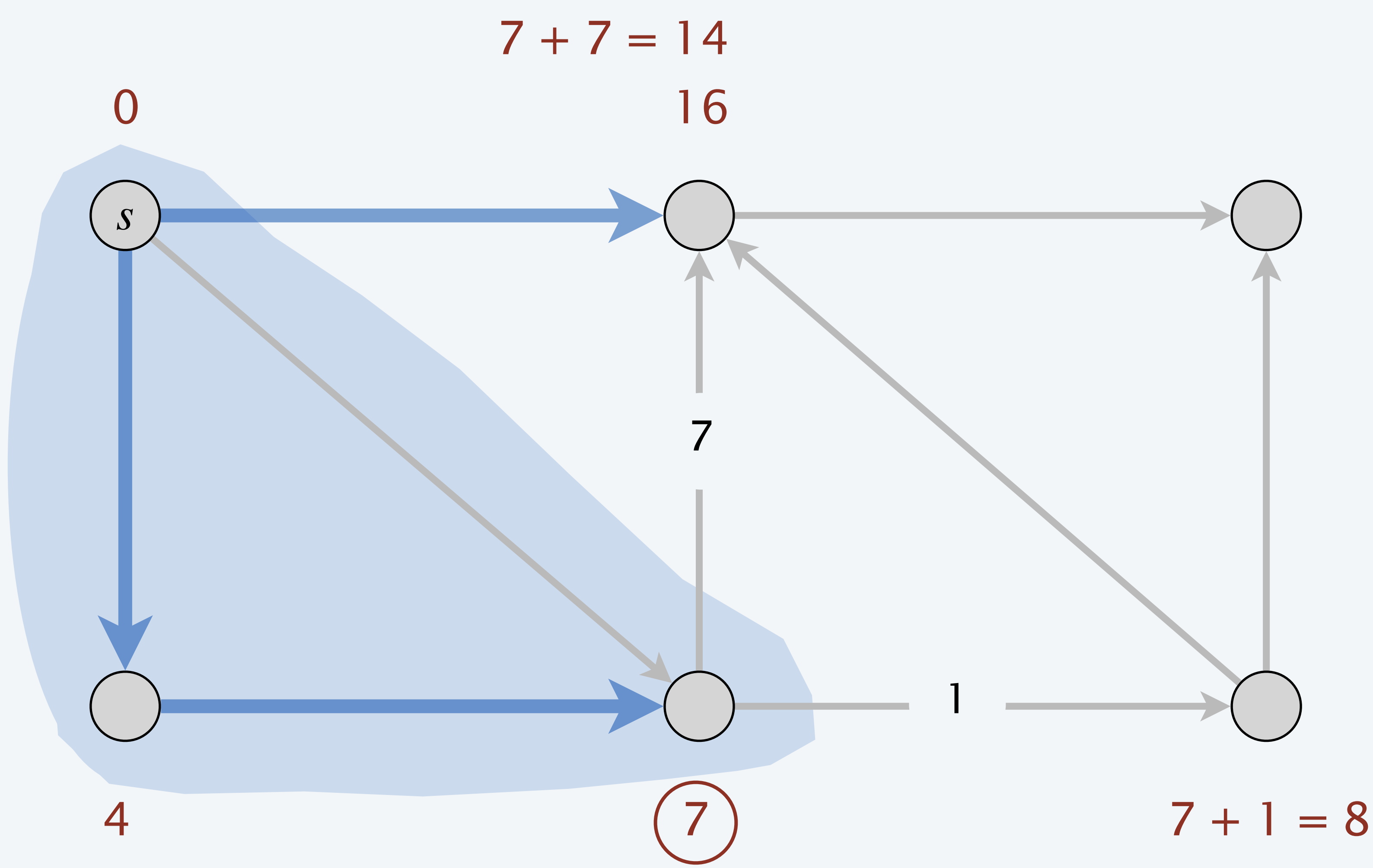
- Add u to S.
- For each edge $e = (u, v)$ leaving u, if $\pi[v] > \pi[u] + \ell_e$ then:
  - $\pi[v] \leftarrow \pi[u] + \ell_e$
  - $pred[v] \leftarrow e$



Dijkstra's algorithm demo (efficient implementation)

Basic step. Choose unexplored node $u \notin S$ with minimum $\pi[u]$.

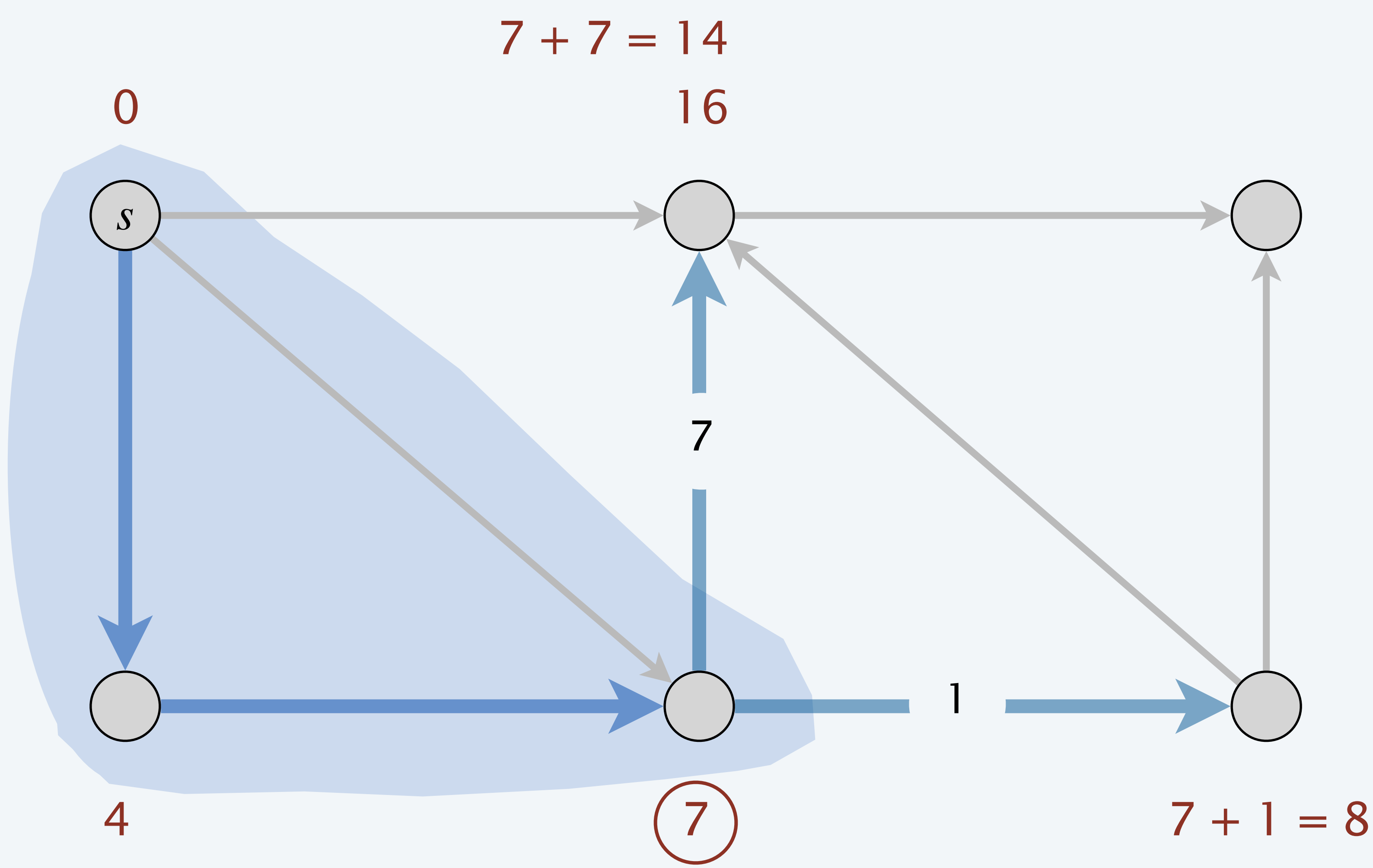
- Add u to S.
- For each edge $e = (u, v)$ leaving u, if $\pi[v] > \pi[u] + \ell_e$ then:
  - $\pi[v] \leftarrow \pi[u] + \ell_e$
  - $pred[v] \leftarrow e$



Dijkstra's algorithm demo (efficient implementation)

Basic step. Choose unexplored node $u \notin S$ with minimum $\pi[u]$.

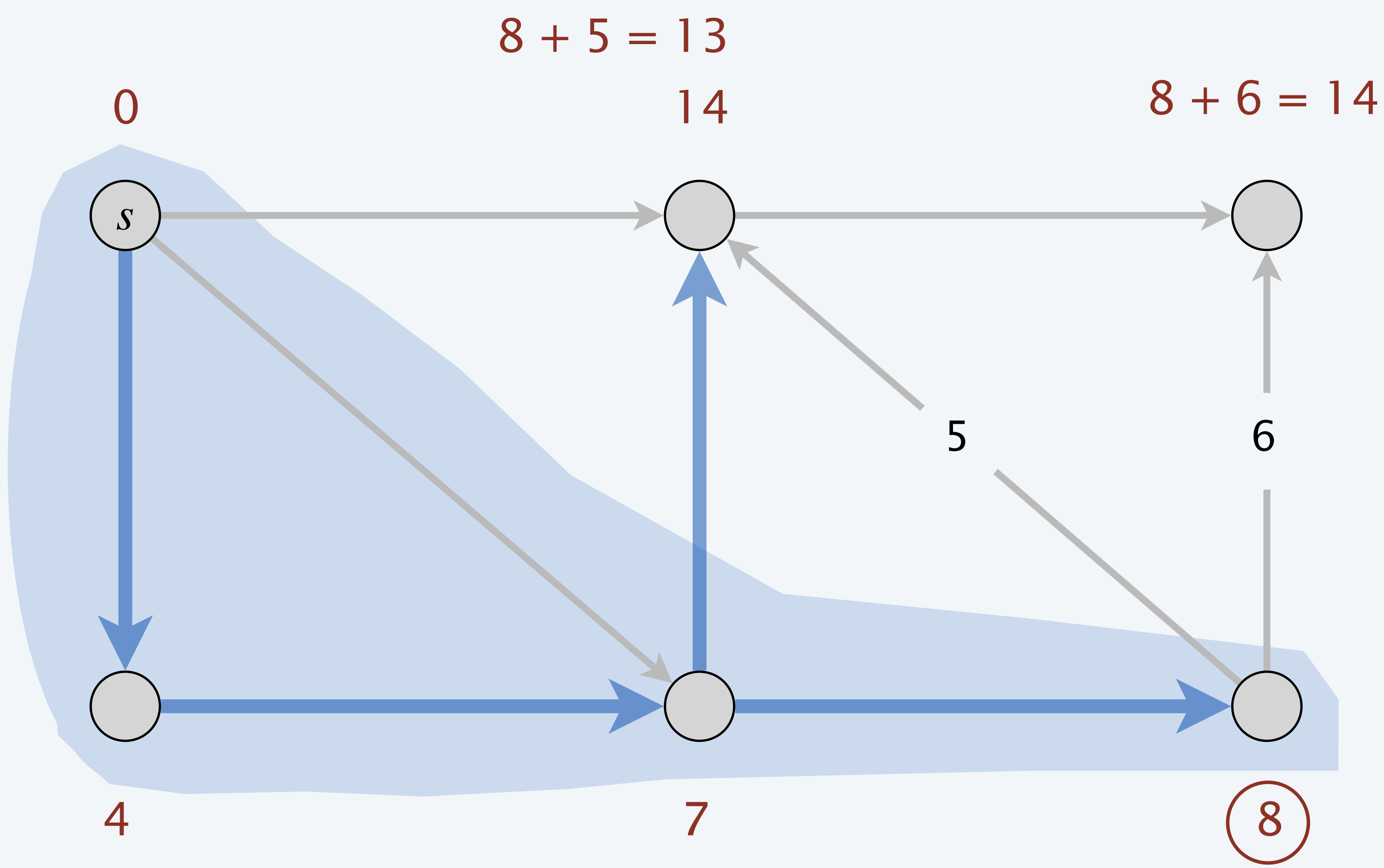
- Add u to S.
- For each edge $e = (u, v)$ leaving u, if $\pi[v] > \pi[u] + \ell_e$ then:
  - $\pi[v] \leftarrow \pi[u] + \ell_e$
  - $pred[v] \leftarrow e$



Dijkstra's algorithm demo (efficient implementation)

Basic step. Choose unexplored node $u \notin S$ with minimum $\pi[u]$.

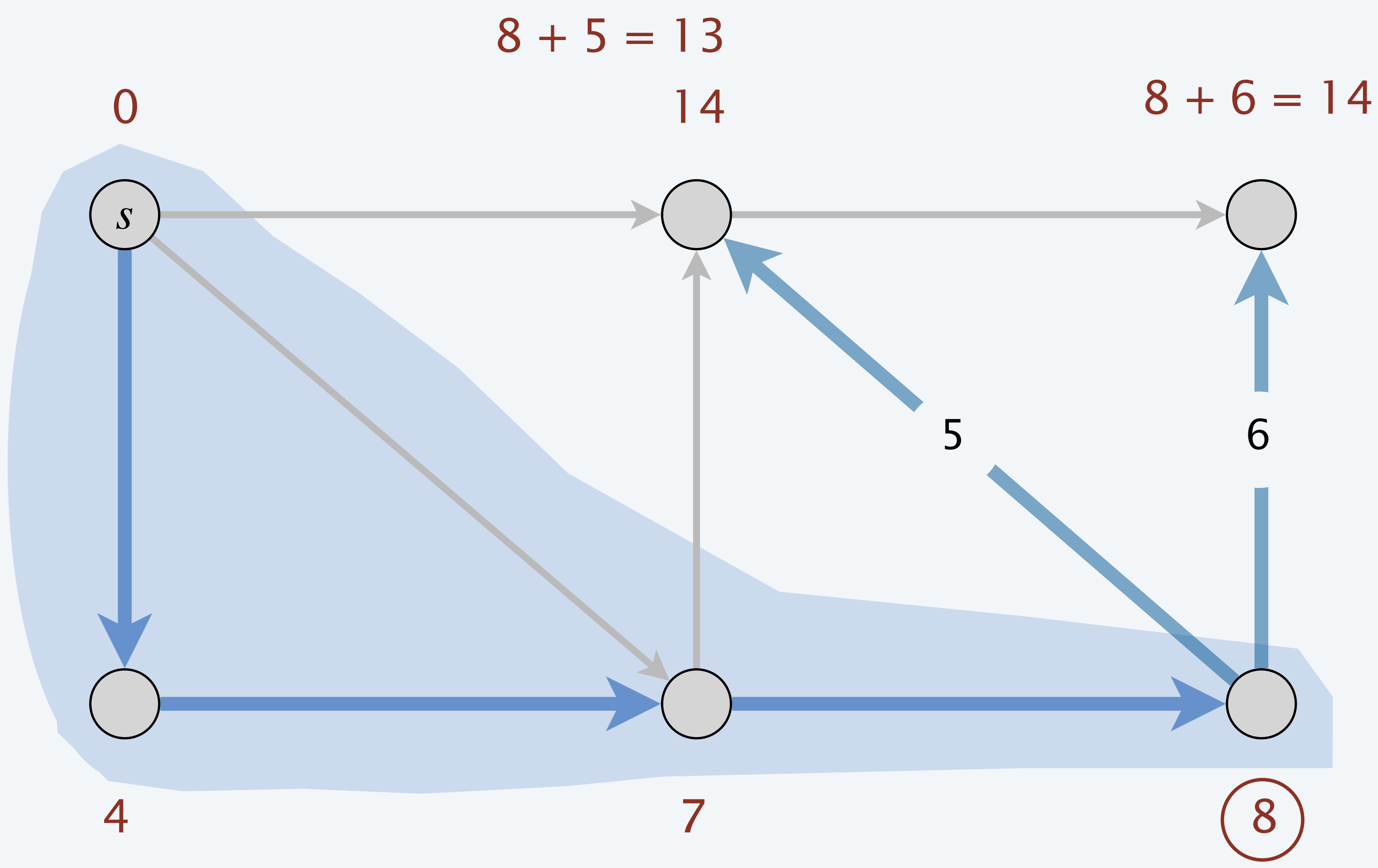
- Add u to S.
- For each edge $e = (u, v)$ leaving u, if $\pi[v] > \pi[u] + \ell_e$ then:
  - $\pi[v] \leftarrow \pi[u] + \ell_e$
  - $pred[v] \leftarrow e$



Dijkstra's algorithm demo (efficient implementation)

Basic step. Choose unexplored node $u \notin S$ with minimum $\pi[u]$.

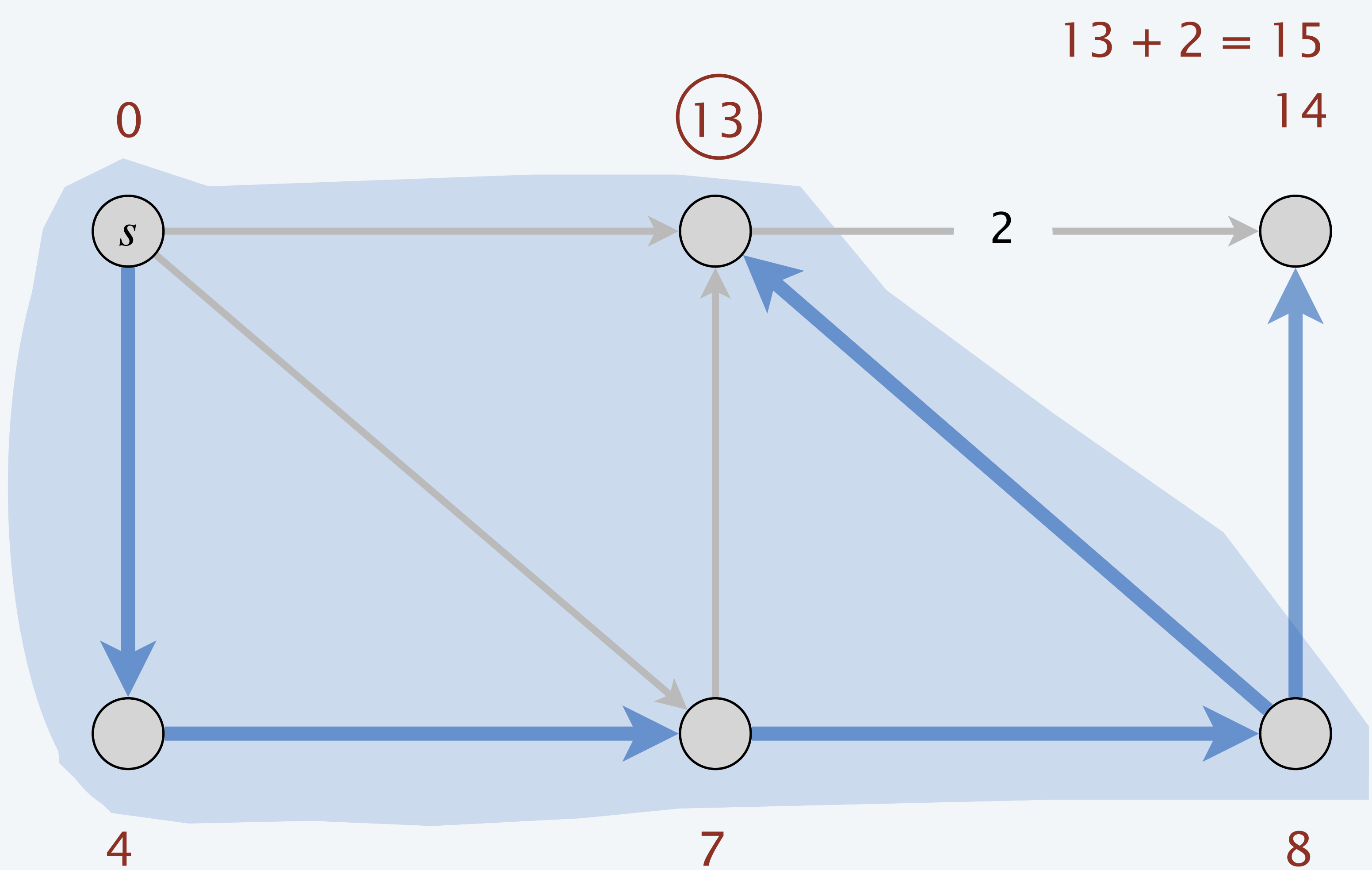
- Add u to S.
- For each edge $e = (u, v)$ leaving u, if $\pi[v] > \pi[u] + \ell_e$ then:
  - $\pi[v] \leftarrow \pi[u] + \ell_e$
  - $pred[v] \leftarrow e$



Dijkstra's algorithm demo (efficient implementation)

Basic step. Choose unexplored node $u \notin S$ with minimum $\pi[u]$.

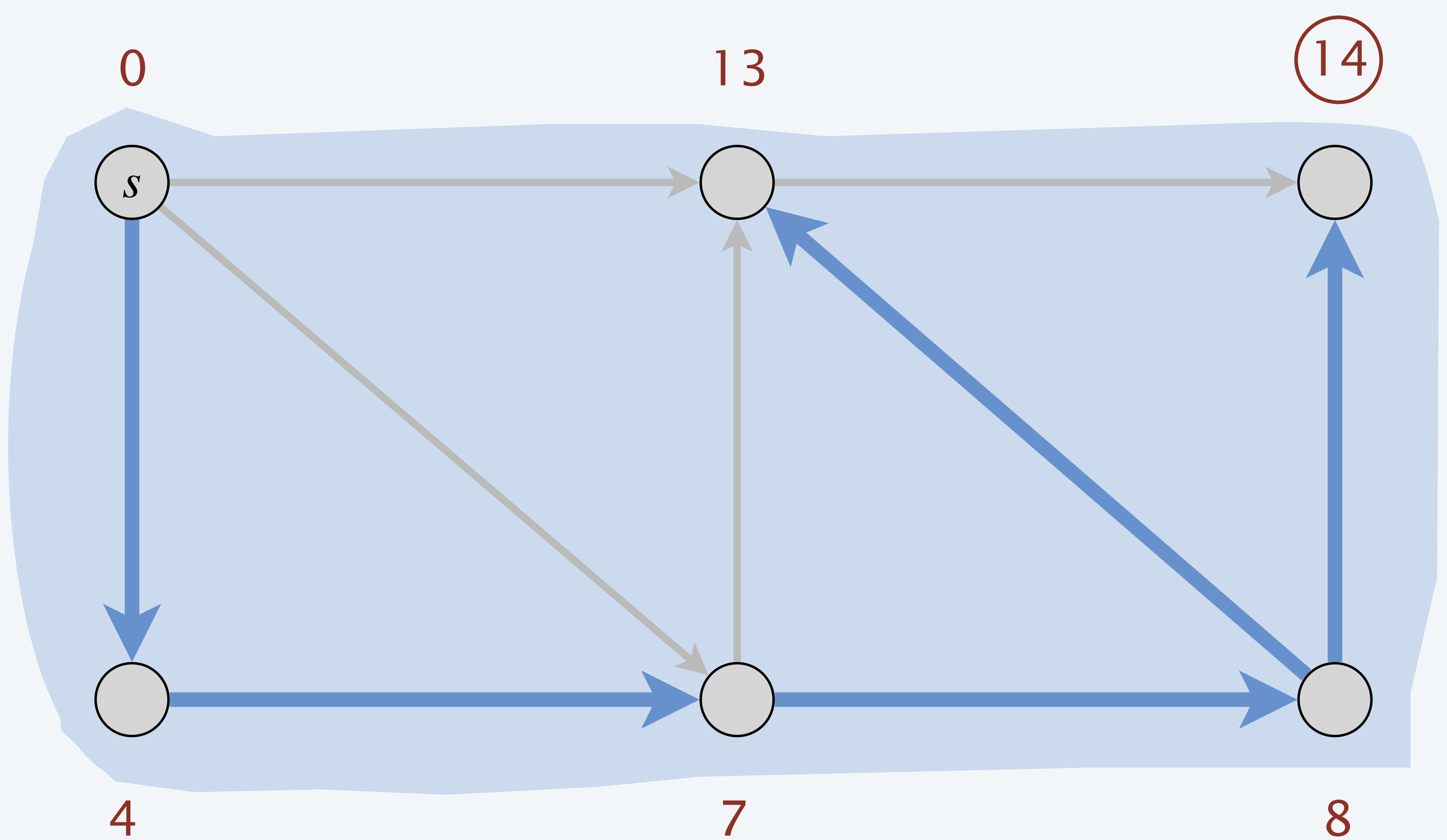
- Add u to S.
- For each edge $e = (u, v)$ leaving u, if $\pi[v] > \pi[u] + \ell_e$ then:
  - $\pi[v] \leftarrow \pi[u] + \ell_e$
  - $pred[v] \leftarrow e$



Dijkstra's algorithm demo (efficient implementation)

Basic step. Choose unexplored node $u \notin S$ with minimum $\pi[u]$.

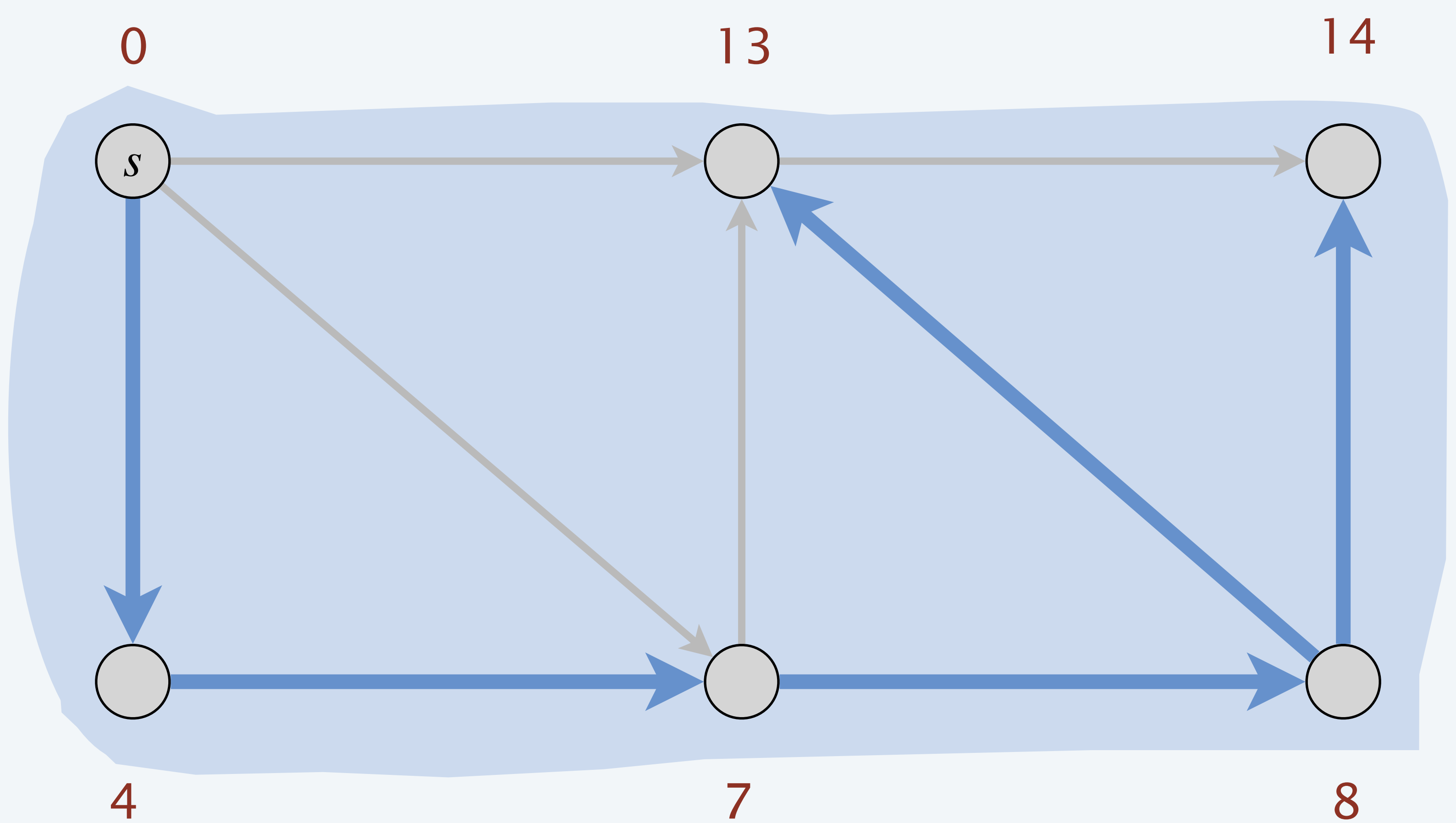
- Add u to S.
- For each edge $e = (u, v)$ leaving u, if $\pi[v] > \pi[u] + \ell_e$ then:
  - $\pi[v] \leftarrow \pi[u] + \ell_e$
  - $pred[v] \leftarrow e$



Dijkstra's algorithm demo (efficient implementation)

Basic step. Choose unexplored node $u \notin S$ with minimum $\pi[u]$.

- Add u to S.
- For each edge $e = (u, v)$ leaving u, if $\pi[v] > \pi[u] + \ell_e$ then:
  - $\pi[v] \leftarrow \pi[u] + \ell_e$
  - $pred[v] \leftarrow e$



Dijkstra's algorithm demo (efficient implementation)

Termination.

- $\pi[v]$ = length of a shortest $s \rightarrow v$ path.
- $pred[v]$ = last edge on a shortest $s \rightarrow v$ path.