Portland State University

W'21 CS 584/684
Algorithm Design & Analysis

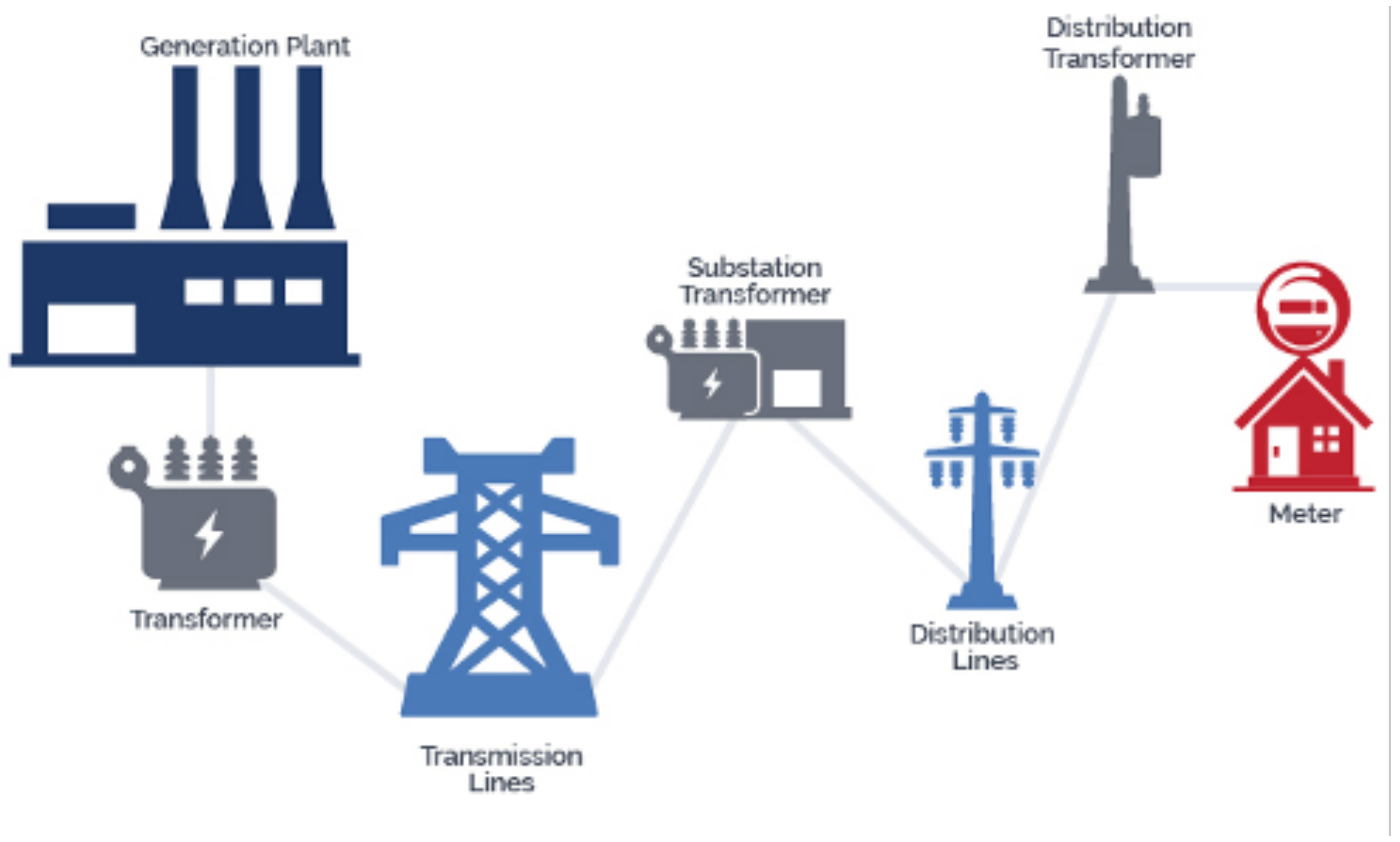
Fang Song

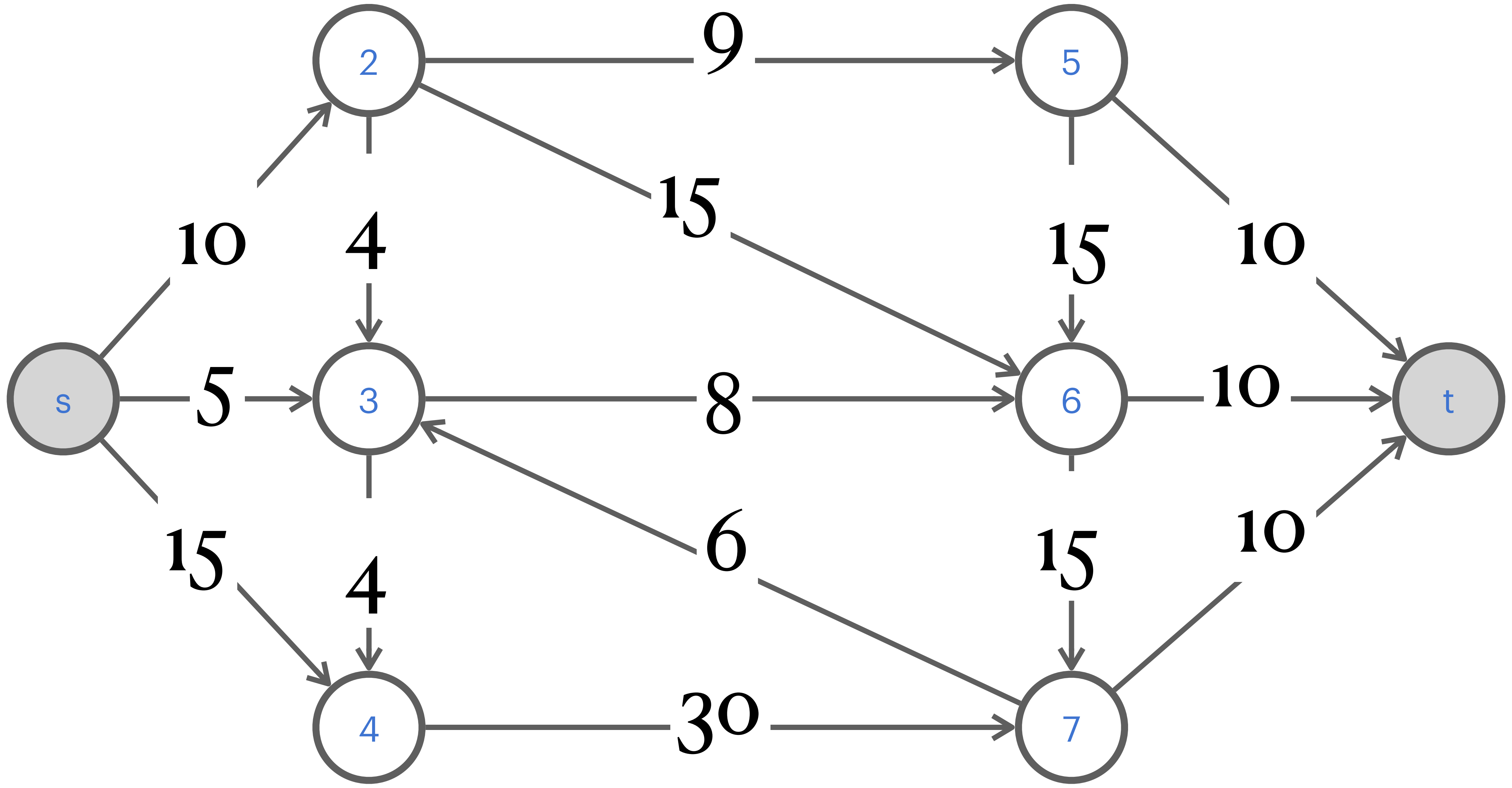
Lecture 13

- Amortized analysis
- Network flow

Recap: flow network

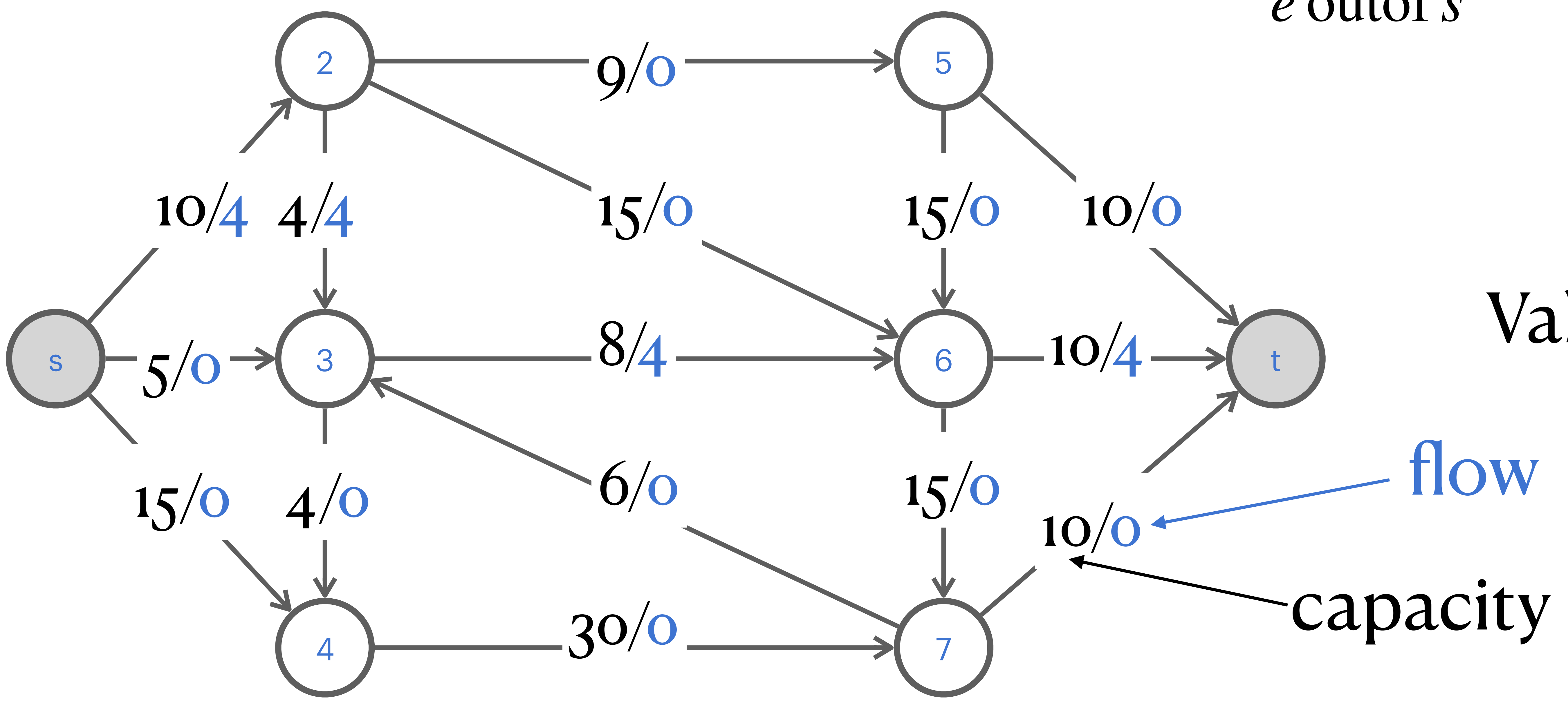
- **Abstraction for material flowing through the edges.**
  - $G = (V, E)$ directed graph, no parallel edges.
  - Two distinguished nodes: $s =$ source, $t =$ sink.
  - $c(e)$: capacity of edge e, $\forall e \in E$.





Flows

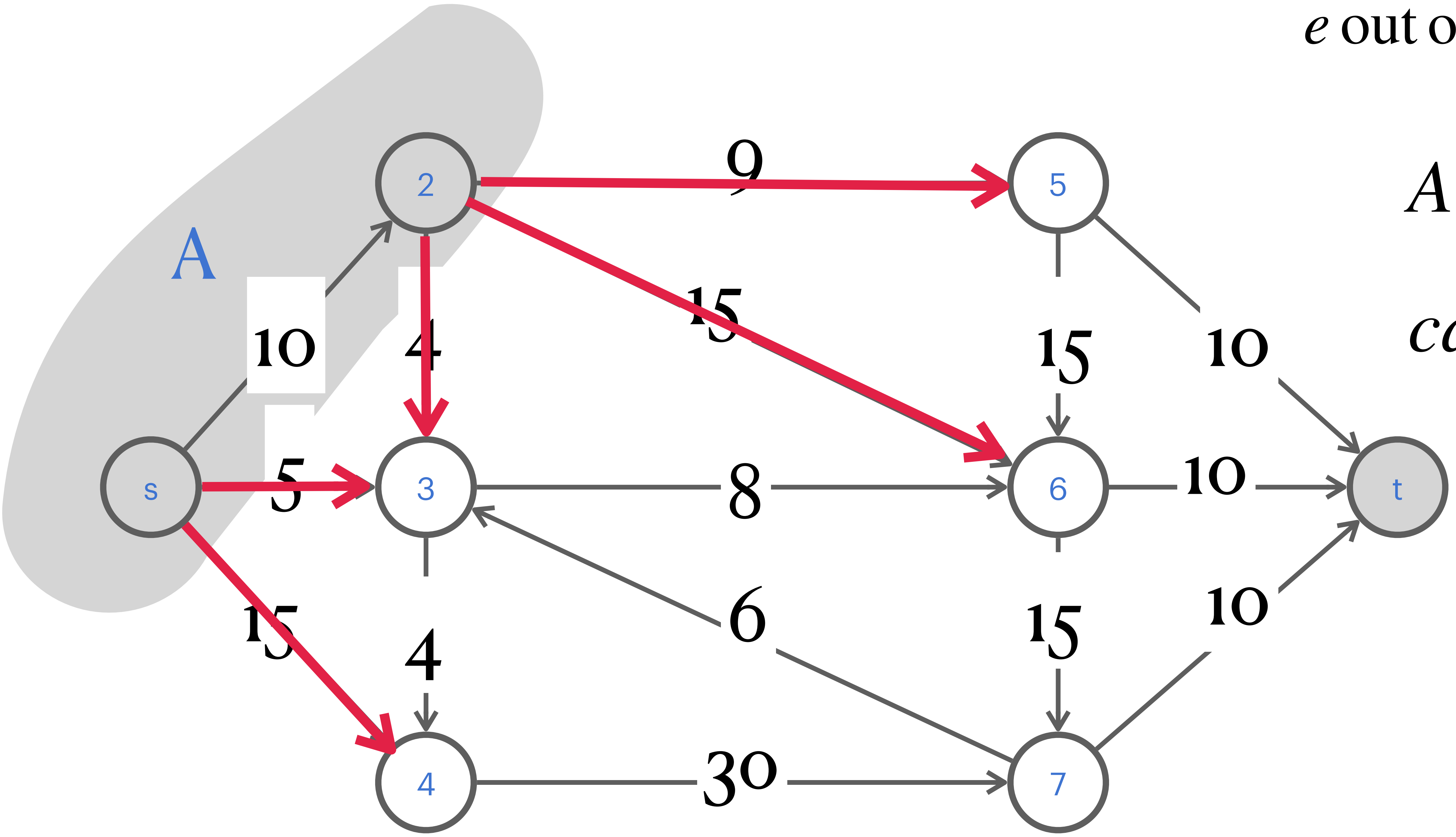
- **Definition. An $s - t$ flow is a function $f : E \to \mathbb{R}^+$ satisfying**
  - [Capacity] $\forall e \in E : 0 \le f(e) \le c(e)$.
  - [Conservation] $\forall v \in V - \{s, t\}$: $\displaystyle\sum_{e \text{ into } v} f(e) = \sum_{e \text{ out of } v} f(e)$
- **Definition. The value of a flow f is $v(f) := \displaystyle\sum_{e \text{ outof } s} f(e)$**



Value $v(f) = 4 + 0 + 0 = 4$

Cuts

- Recall. A cut is a subset of vertices.
- Def. $s - t$ cut: $(A, B = V - A)$ partition of V with $s \in A$ and $t \in B$.
- Def. Capacity of cut (A, B): $cap(A, B) = \sum_{e \text{ out of } A} c(e)$



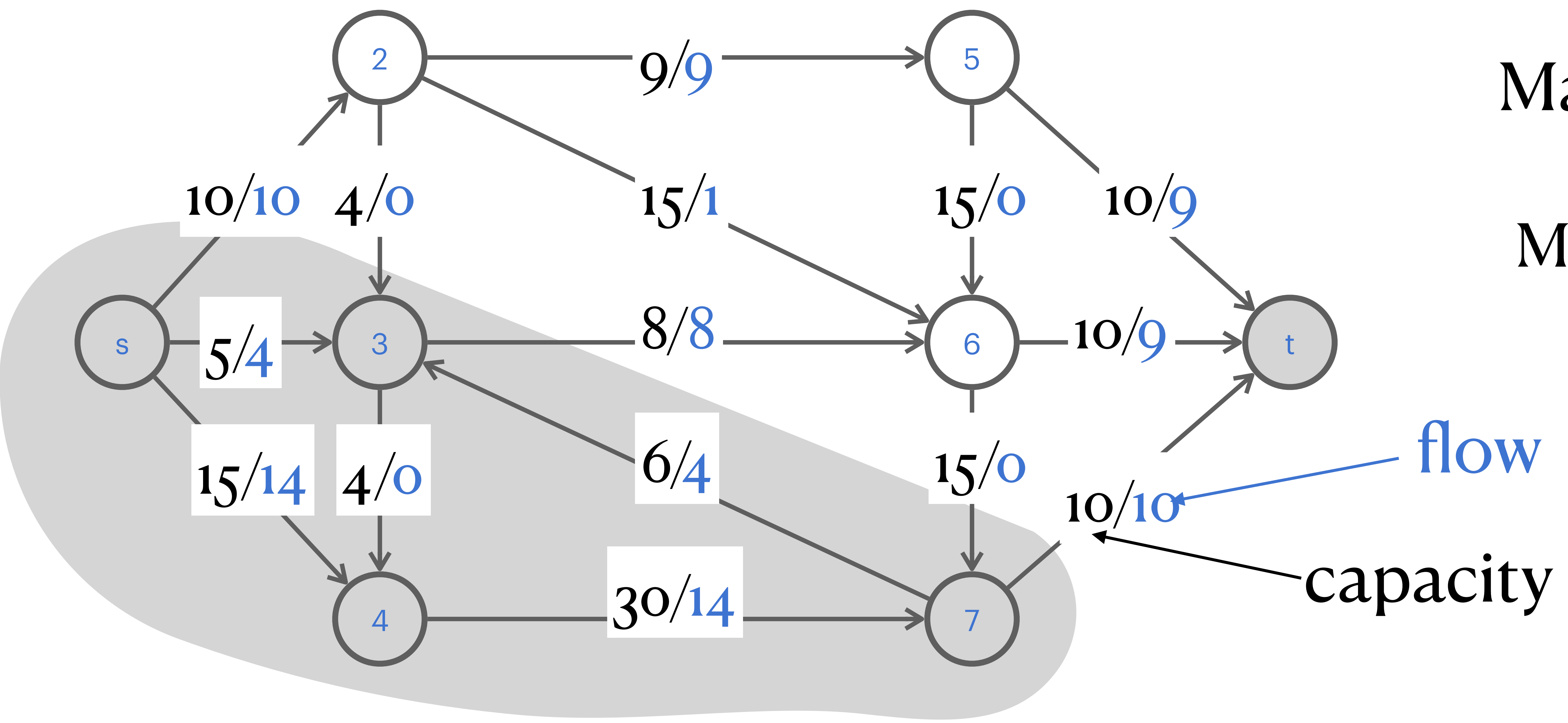
$A = \{s,2\}, B = \{3,4,\ldots,t\}$

$cap(A, B) = 9 + 15 + 4 + 5 + 15 = 48$

How do they relate?

Max flow   Min cut

- Find $s - t$ flow of maximum value.
- Find $s - t$ cut of minimum capacity.



Max flow $v(f) = 28$

Min cut $cap(A, B) = 28$

Useful observations

- **Flow-value lemma**. Let f be any flow, and let (A, B) be any $s - t$ cut. Then the net flow across the cut is equal to the amount leaving s (i.e., value of flow).

$$\sum_{e \text{ out of } A} f(e) - \sum_{e \text{ into } A} f(e) = v(f)$$

- **Weak duality**. Let f be any flow, and let (A, B) be any $s - t$ cut. Then the value of the flow is at most the capacity of the cut.

$$v(f) \leq cap(A, B)$$

- **Corollary of weak duality**. Let f be any flow, and let (A, B) be any $s - t$ cut. If $v(f) = cap(A, B)$, then f is a max flow, and (A, B) a min cut.

Max-flow Min-cut theorem

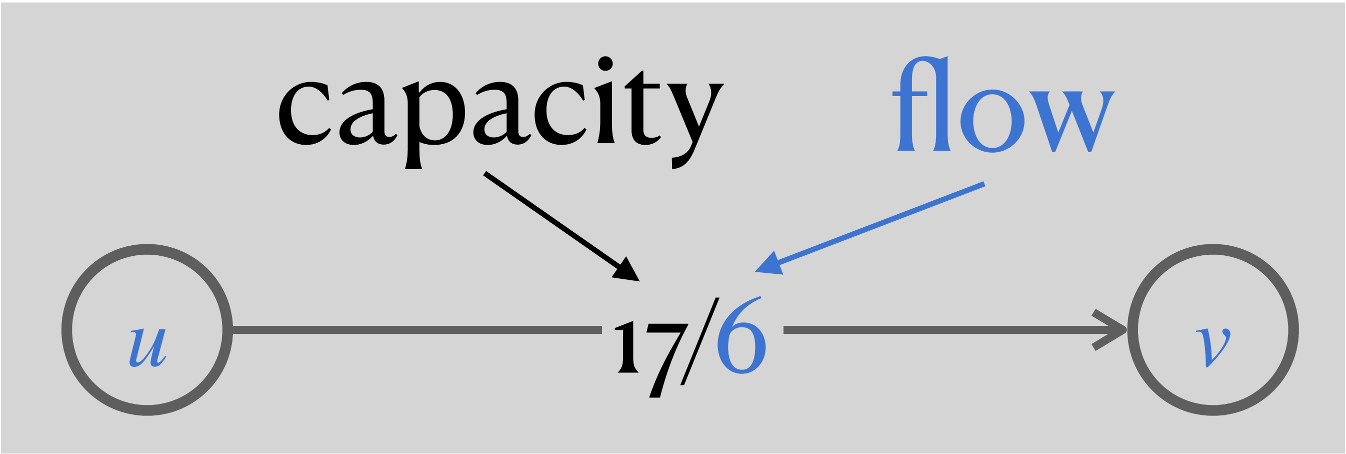
Theorem. Value of max flow = capacity of min cut.

[Strong duality]

A constructive proof: augmenting path

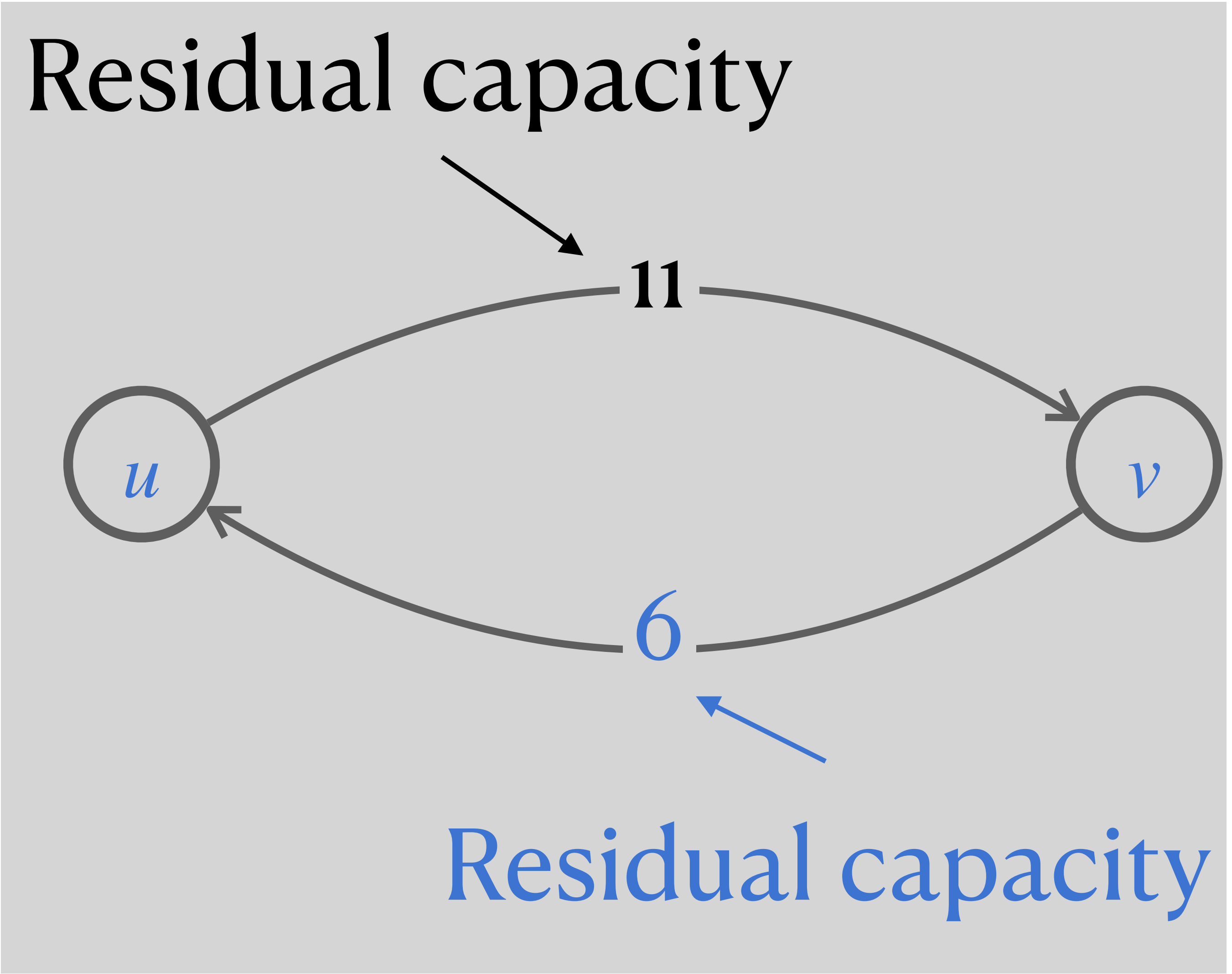
Residual graph

- **Original edge:** $e = (u, v) \in E$
  - Capacity $c(e)$, low $f(e)$.



- **Residual edge: "undo" flow**
  - $e = (u, v)$ and $e^R = (v, u)$
  - Residual capacity with flow f:

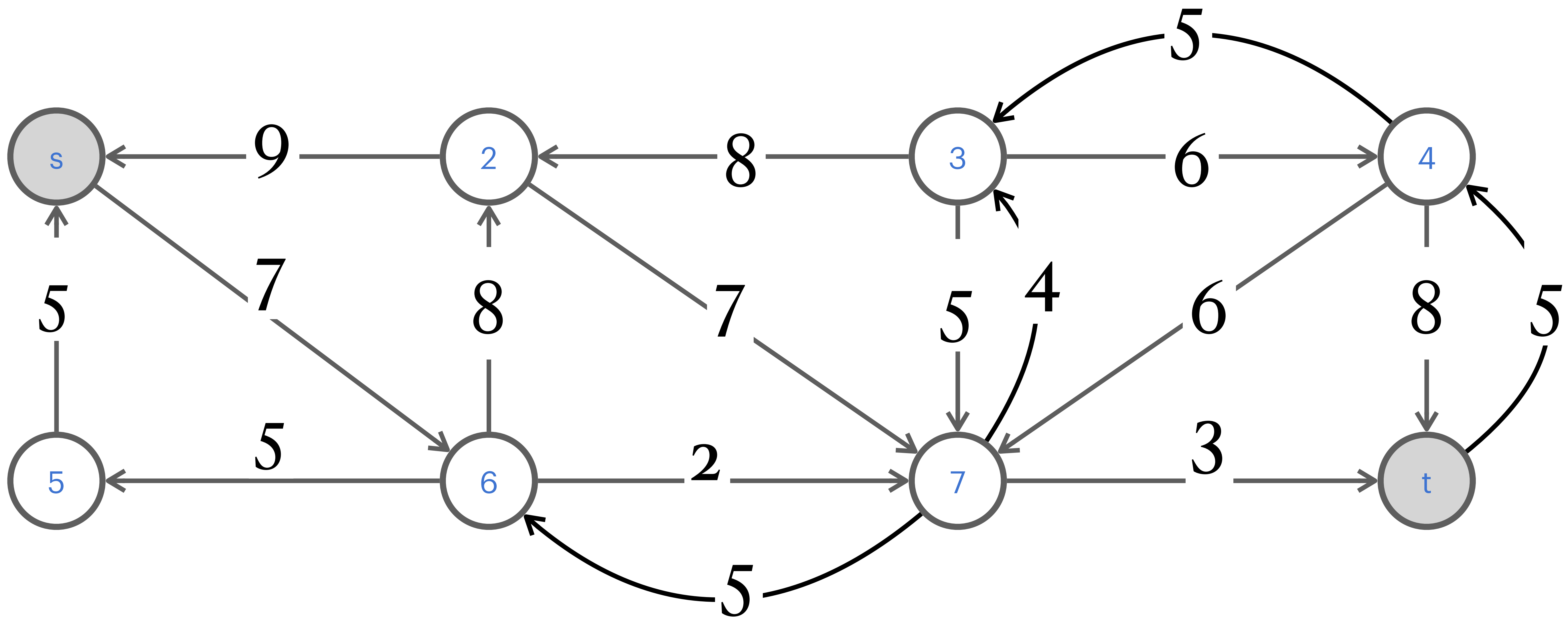
$$C_f(u,v) = \begin{cases} c(u,v) - f(u,v) & \text{if } (u,v) \in E \\ f(v,u) & \text{if } (v,u) \in E \end{cases}$$



- **Residual graph $G_f = (V, E_f)$**
  - Residual edges with positive residual capacity.
  - $E_f = \{e : f(e) < c(e)\} \cup \{e^R : f(e) > 0\}$.

Augmenting path

- Definition. An augmenting path is a simple $s \rightsquigarrow t$ path in residual graph G_f.
- Definition. The bottleneck capacity of an augmenting path P is the minimum residual capacity of any edge in P.



Which augmenting path has the highest bottleneck capacity ?

Augmenting path theorem

- **Theorem.** f is a max flow iff. NO augmenting paths $s \rightsquigarrow t$ in G_f.

  A.k.a. Algorithmic max-flow min-cut theorem.

- **Proof.** We show the following equivalence ($a \Rightarrow b \Rightarrow c \Rightarrow a$)
  a. f is a max flow.
  b. There is no augmenting path (with respect to f).
  c. There exists a cut (A, B) such that $cap(A, B) = v(f)$.

  Corollary of weak duality. Also implies (A, B) a min-cut.

- **N.B.** $a \Leftrightarrow c$ is Max-flow min-cut Theorem: value of max flow = capacity of min cut.

Augmenting path theorem: proof

a. f is a max flow.
b. There is no augmenting path (with respect to f).
c. There exists a cut (A, B) such that $cap(A, B) = v(f)$.

- $a \Rightarrow b$. **We show contrapositive** $\neg b \Rightarrow \neg a$.
  - If $\exists$ augmenting path, we can find a new flow f' with larger flow value below.

$\delta \leftarrow$ bottleneck capacity of augmenting path P.

For each $e \in P, f'(e) := \begin{cases} f(e) + \delta & \text{if } e \in E \\ f(e) - \delta & \text{if } e^R \in E \end{cases}$

  - Exercise. Verify f' is a feasible flow (capacity and conservation hold).
  - $v(f') = v(f) + \delta > v(f)$ because only first edge in P leaves s.

Augmenting path theorem: proof cont'd

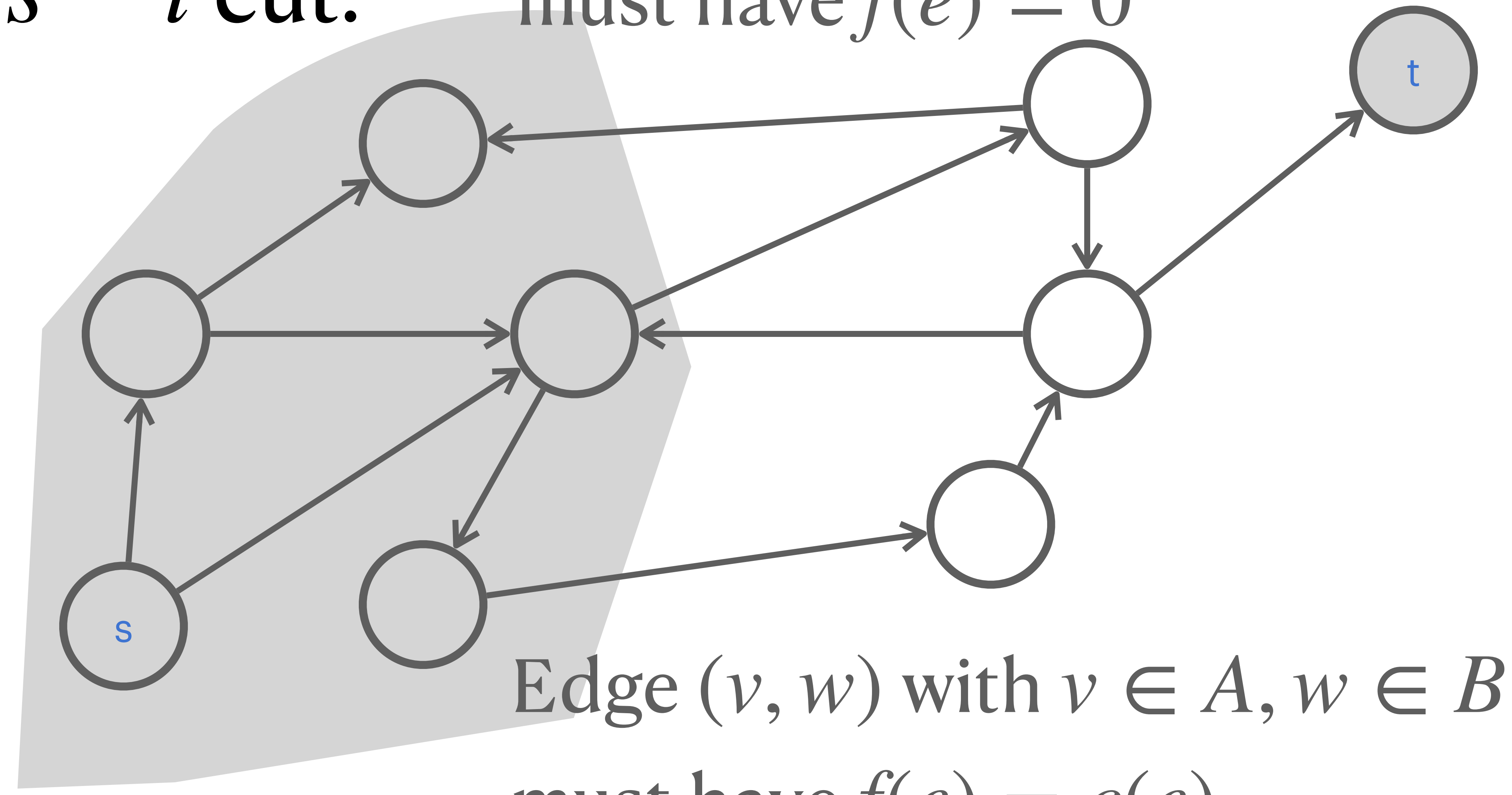
a. f is a max flow.
b. There is no augmenting path (with respect to f).
c. There exists a cut (A, B) such that $cap(A, B) = v(f)$.

- $b \Rightarrow c$. **Assuming G_f has no augmenting path.**
  - Let A be the set of nodes reachable from s in G_f.
  - Clearly $s \in A, t \notin A$. $(A, B = S - A)$ is an $s - t$ cut.
  - Obs. On edges of G go from A to B.

$$v(f) = \sum_{e \text{ out of } A} f(e) - \sum_{e \text{ into } A} f(e)$$

$$= \sum_{e \text{ out of } A} c(e) - 0 = cap(A, B)$$

Edge (v, w) with $v \in B, w \in A$ must have $f(e) = 0$



Edge (v, w) with $v \in A, w \in B$ must have $f(e) = c(e)$

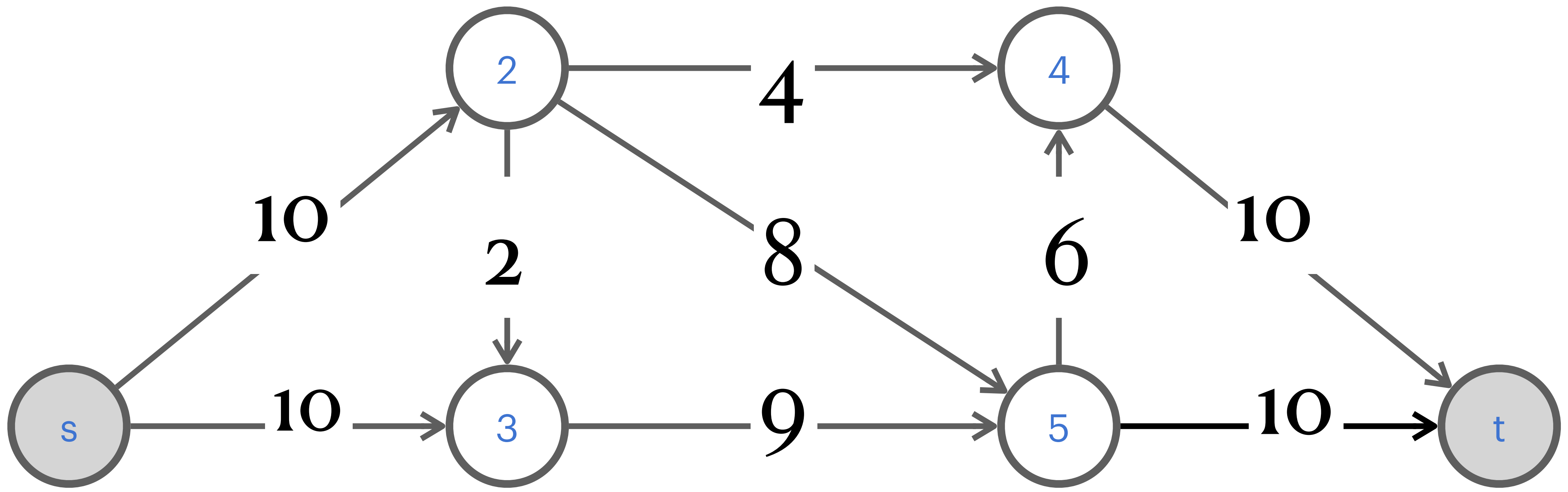
Ford-Fulkerson algorithm

Ford-Fulkerson augmenting path algorithm

Augment(f, c, P)
 $\delta \leftarrow$ bottleneck capacity of augmenting path P.
 For each $e \in P$
  If $e \in E, f'(e) = f(e) + \delta$
  Else $f'(e) = f(e) - \delta$
 Return f'

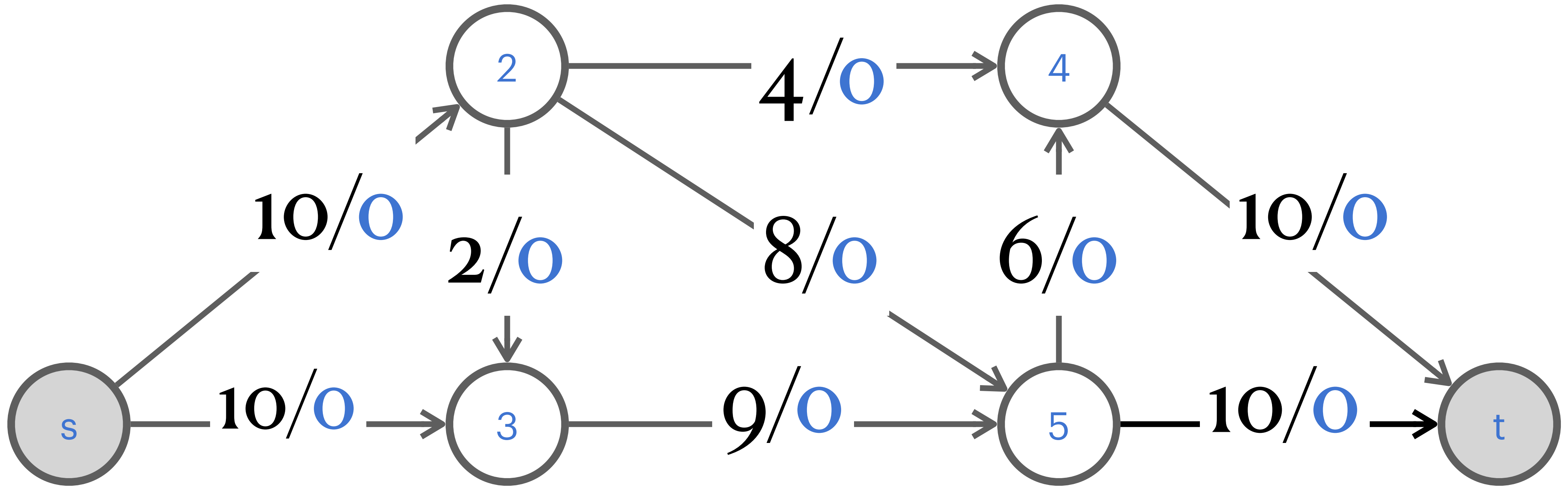
Ford-Fulkerson(G, s, t, c)
 For each $e \in E$
  $f(e) \leftarrow 0$, $G_f \leftarrow$ residual graph
  While there is an augmenting path P in G_f
   $f \leftarrow$ Augment(f, c, P)
   Update G_f
 Return f'

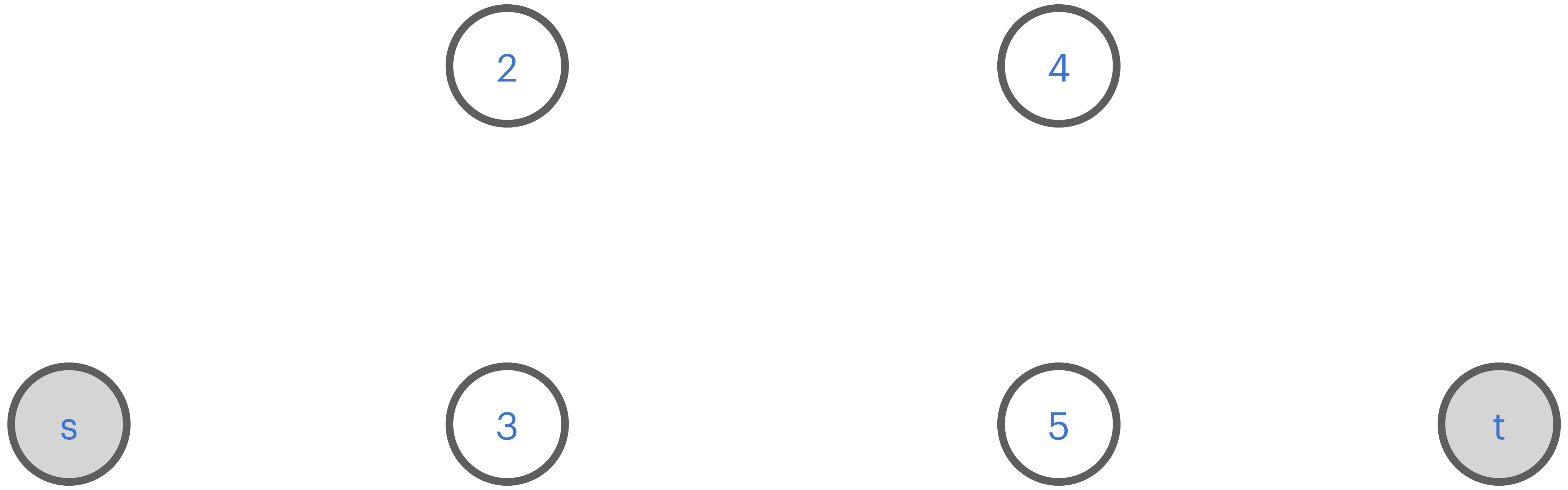
Ford-Fulkerson algorithm: demo0



Ford-Fulkerson algorithm: demo1

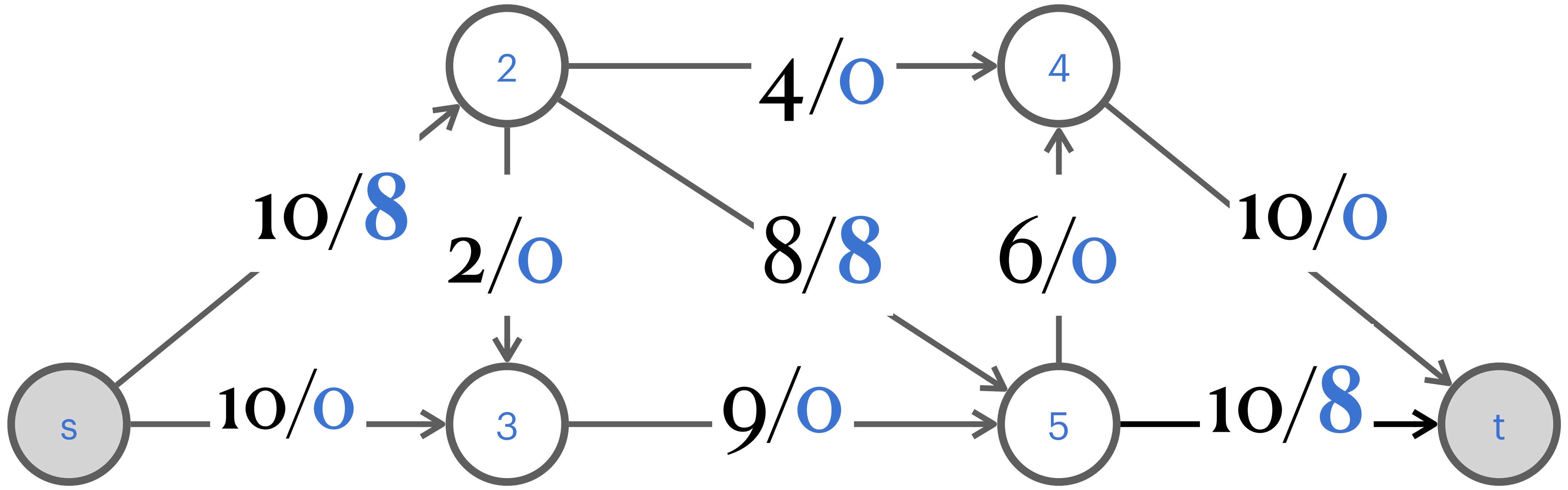
$v(f) = 0$

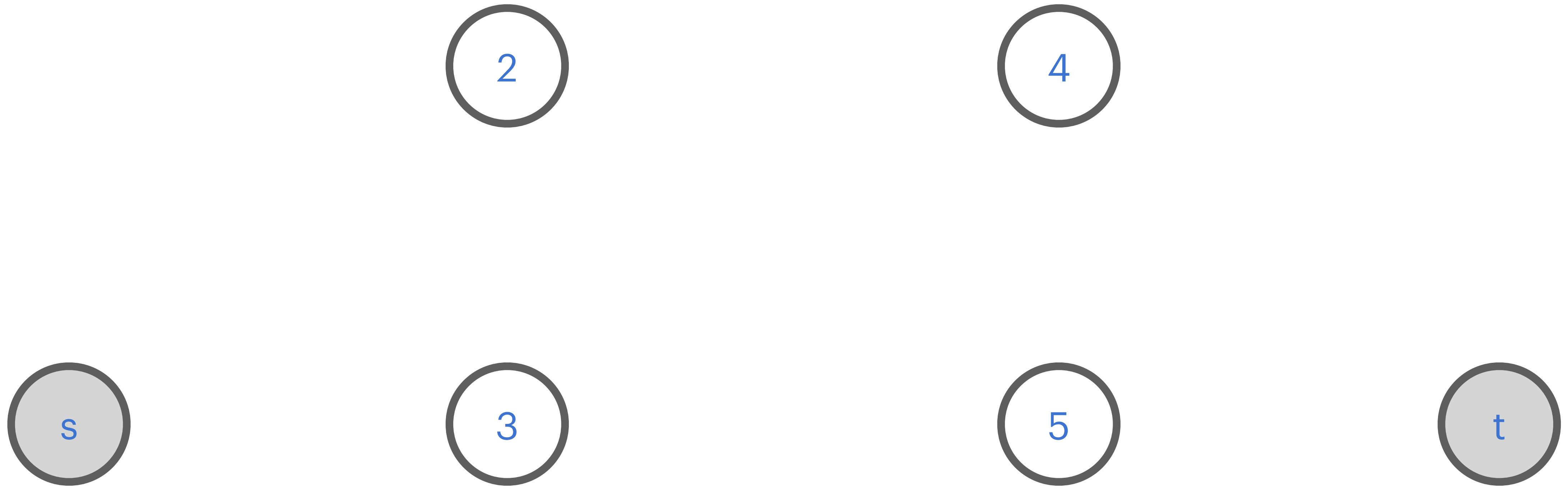




Ford-Fulkerson algorithm: demo2

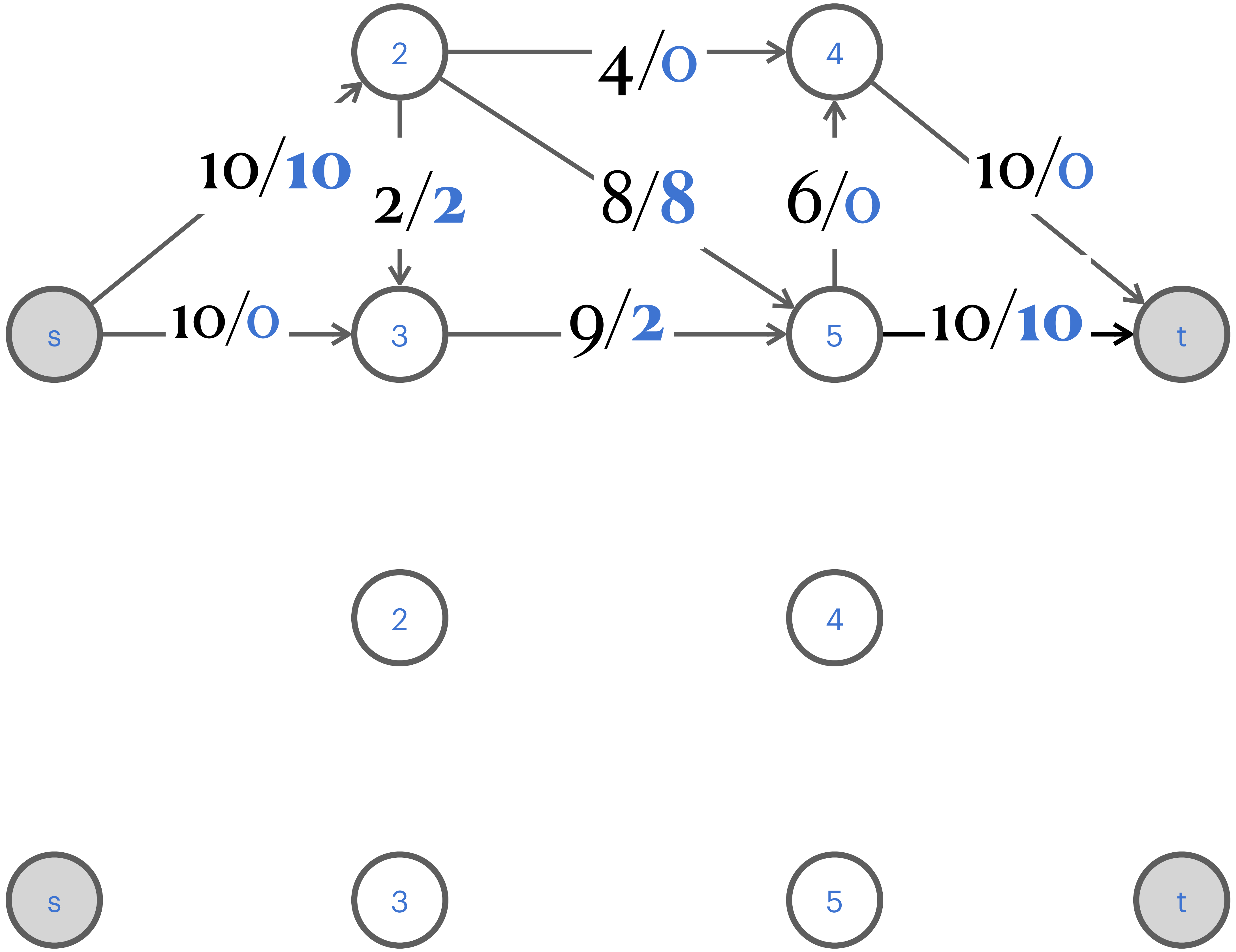
$v(f) = 8$





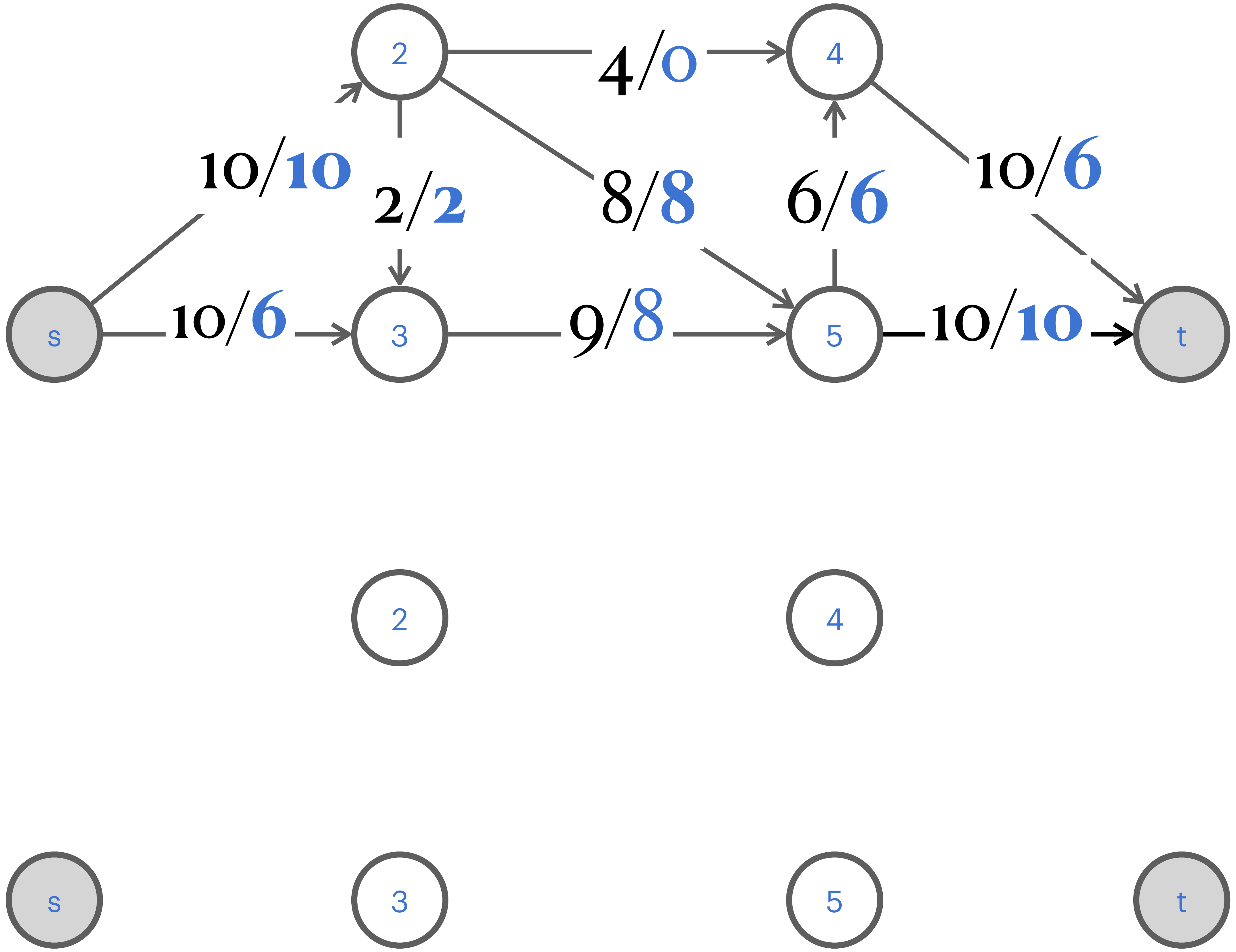
Ford-Fulkerson algorithm: demo3

$v(f) = 10$



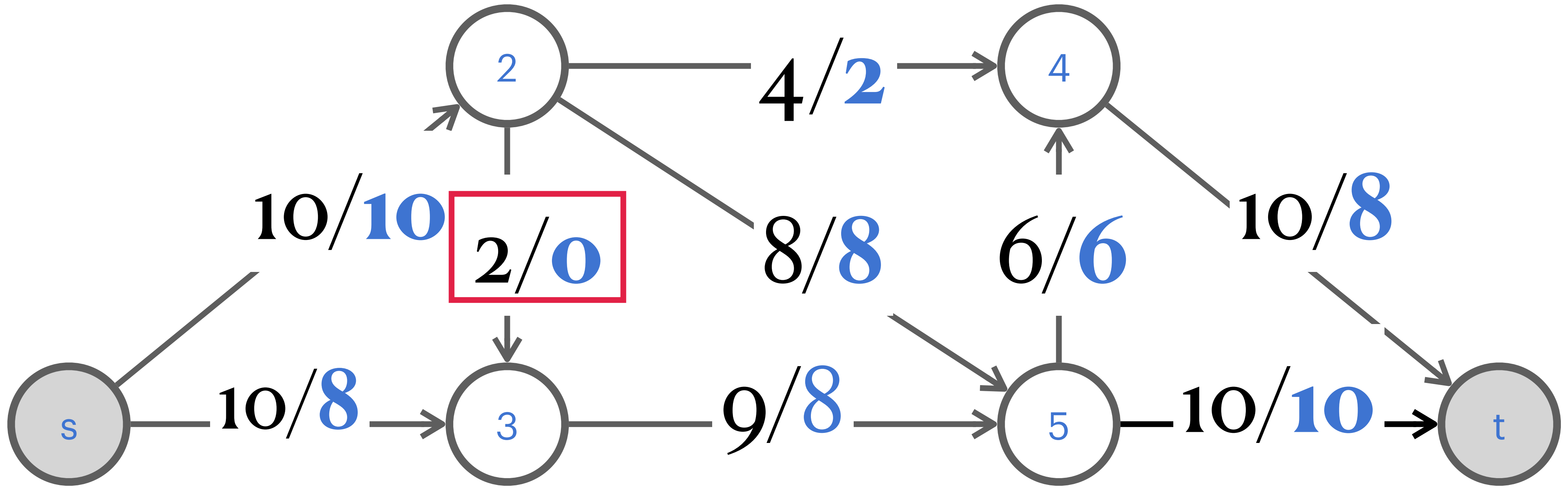
Ford-Fulkerson algorithm: demo4

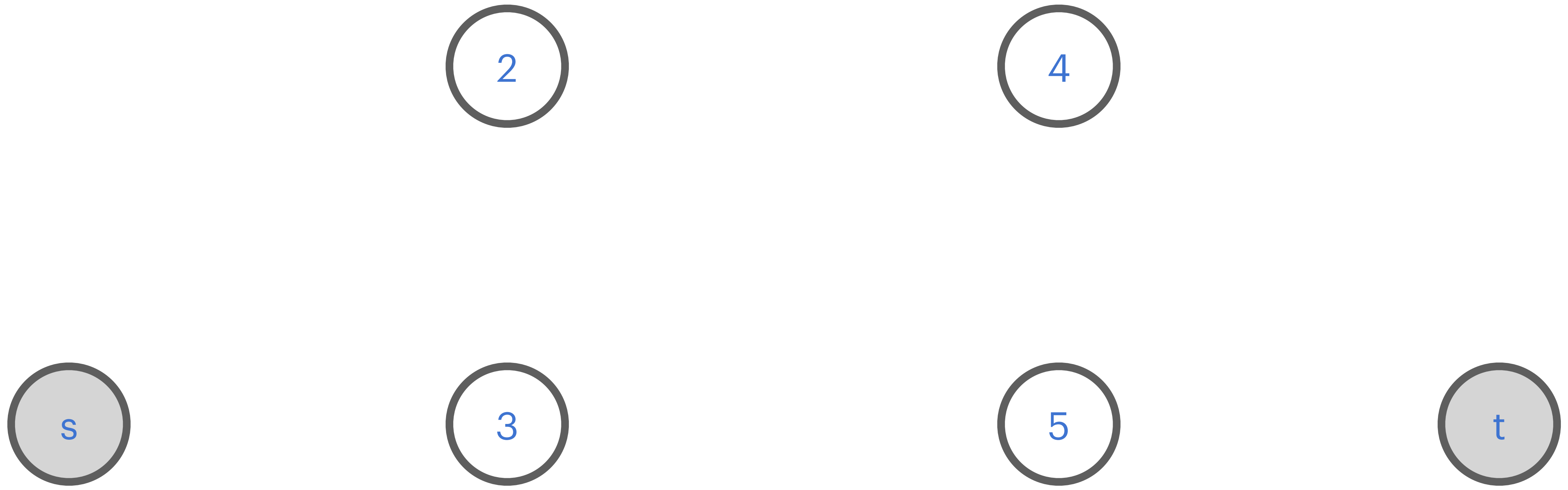
$v(f) = 16$



Ford-Fulkerson algorithm: demo5

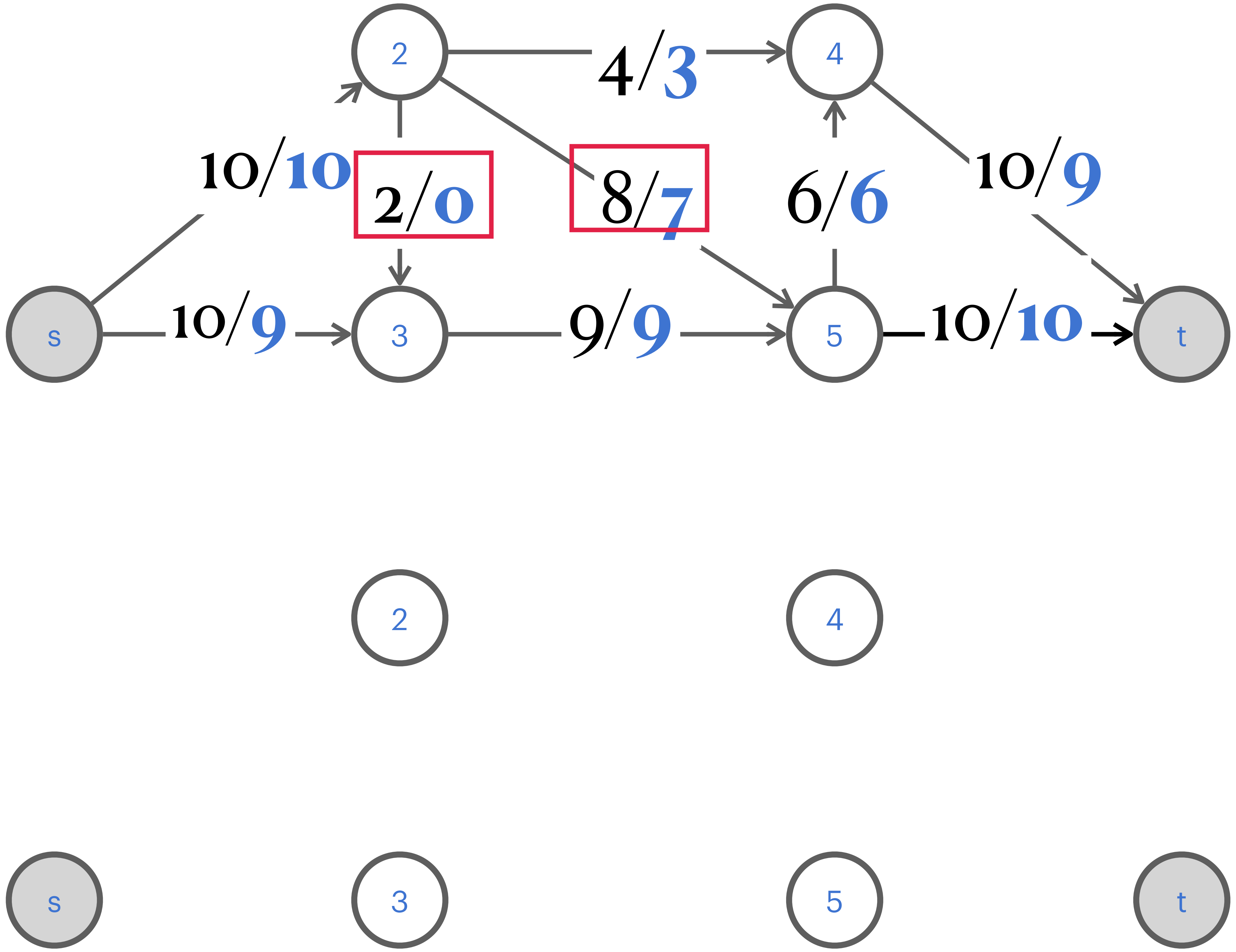
$v(f) = 18$





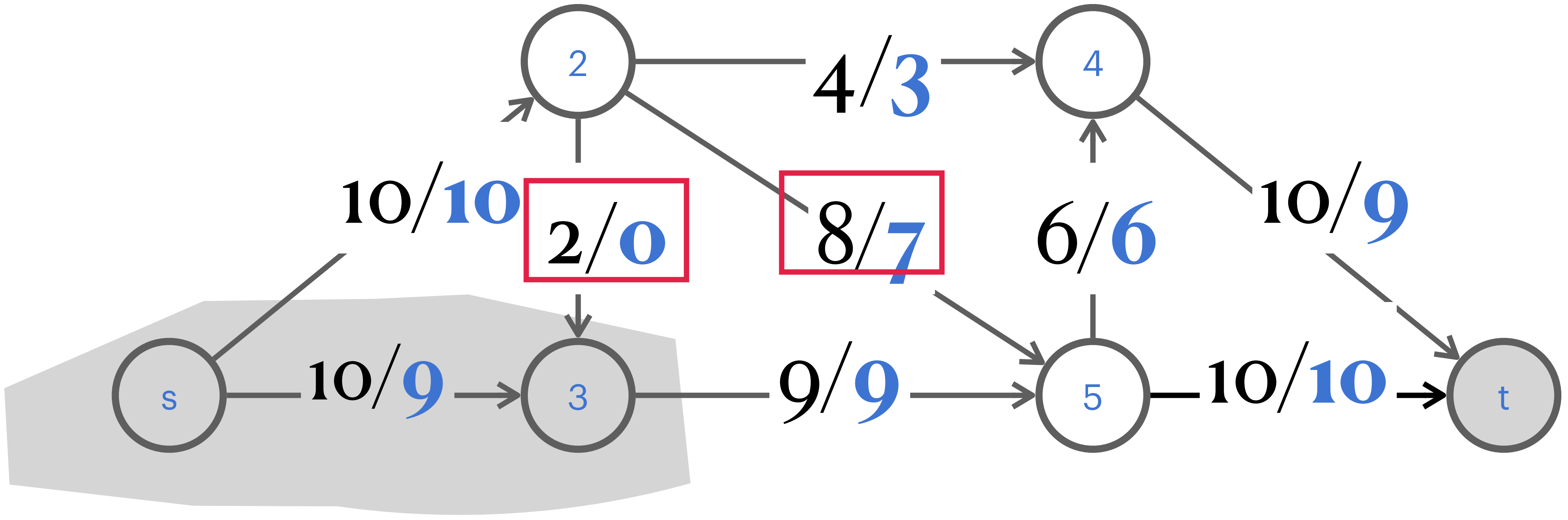
Ford-Fulkerson algorithm: demo6

$v(f) = 19$



Ford-Fulkerson algorithm: demo7

$v(f) = 19$



Cut $(A = \{s,3\}, B = S - A)$, $cap(A, B) = 19$

Fork-Fulkerson algorithm: summary so far

```
Ford-Fulkerson
While you can
    Greedily push flow
    Update residual graph
```

- **Correctness.** Augment path theorem.
- **Running time.** Does it terminate at all?

Fork-Fulkerson algorithm: analysis

- **Assumption.** All capacities are integers between 1 and C.
- **Invariant.** Every flow value $f(e)$ and every residual capacity $c_f(e)$ remains an integer throughout the algorithm.

- **Theorem.** Ford-Fulkerson terminates in at most nC iterations.
- **Proof.**
  - Each augmentation increases flow value by at least 1.
  - There are at most nC units of capacity leaving source s.

Running time: $O(mnc)$. Space $O(m + n)$.

Find an augmenting path in $O(m)$ time (by BFS/DFS)

More to come on further concerns/improvements ...

- **Integrality theorem.** All If all capacities are integres, then there is a max flow f where every flow value $f(e)$ is an integer.

Scratch