CSE 6140/ CX 4140:

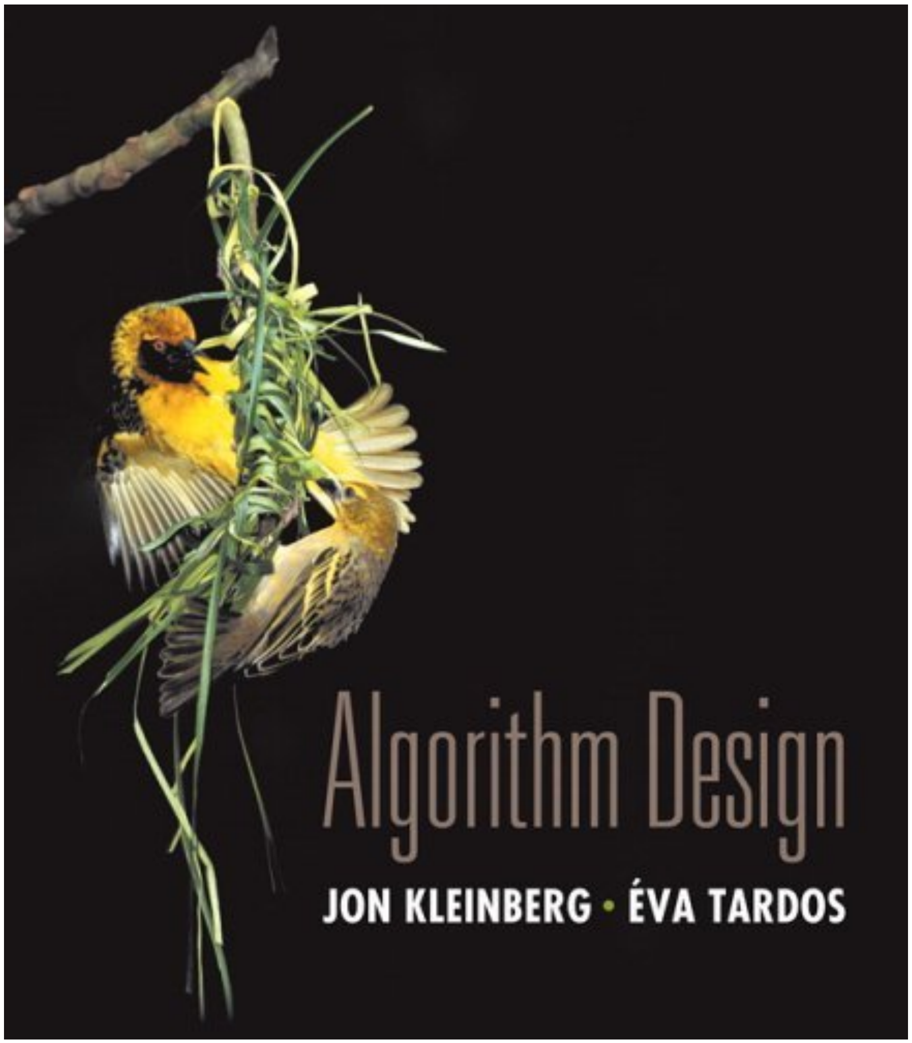
Computational Science and Engineering ALGORITHMS

Instructor: Anne Benoit

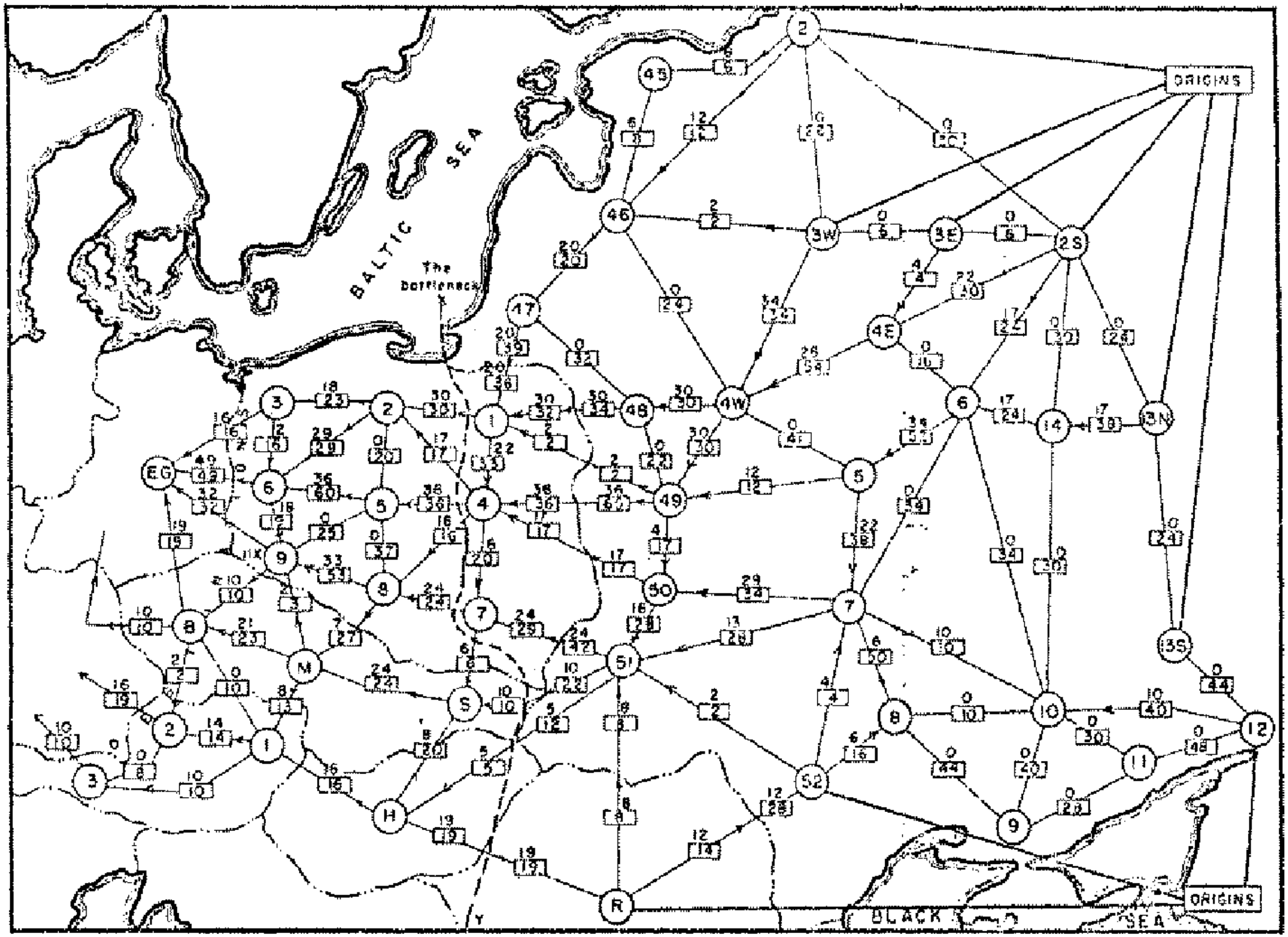
Visiting Associate Professor, CSE

Based on slides by Bistra Dilkina

KT 7 Network Flows



Soviet Rail Network, 1955



Reference: *On the history of the transportation and maximum flow problems.*
Alexander Schrijver in Math Programming, 91: 3, 2002.

Maximum Flow and Minimum Cut

Max flow and min cut.

- Two very rich algorithmic problems.
- Cornerstone problems in combinatorial optimization.
- Beautiful mathematical duality.

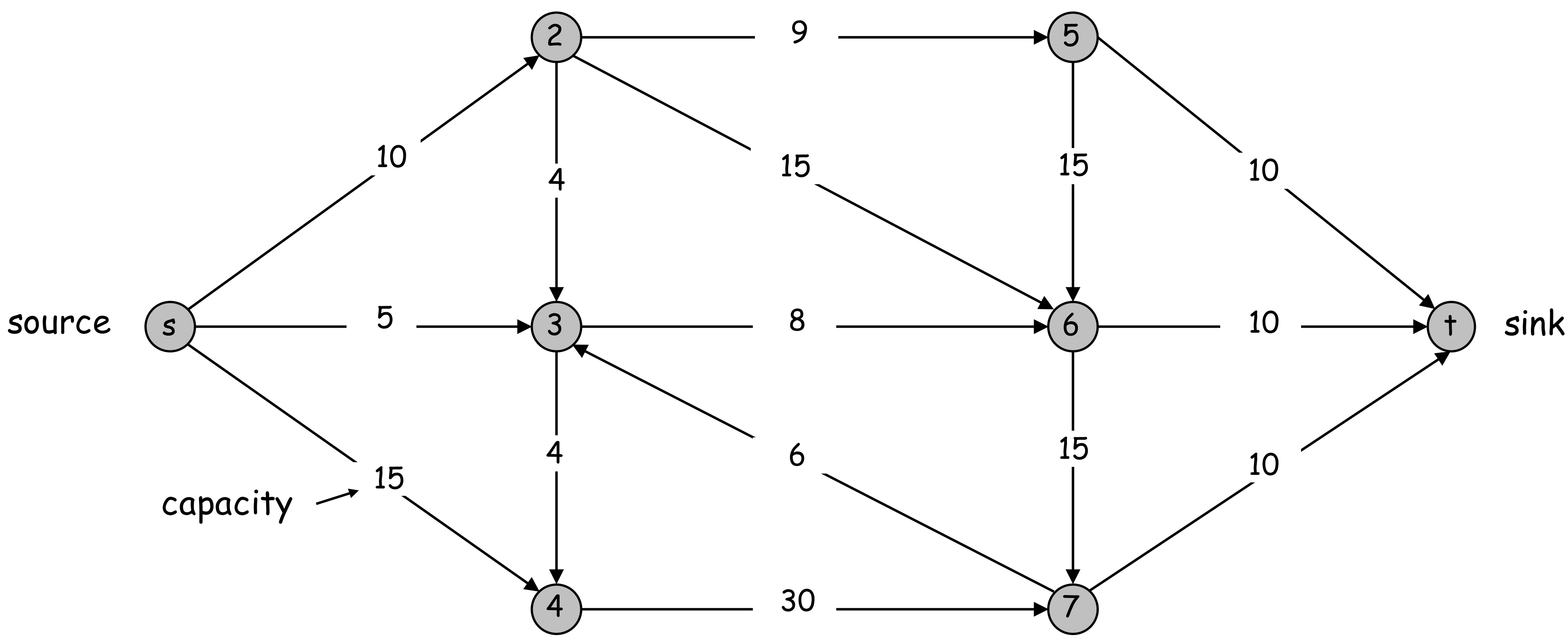
Nontrivial applications / reductions.

- Data mining.
- Open-pit mining.
- Project selection.
- Airline scheduling.
- Bipartite matching.
- Image segmentation.
- Clustering
- Network connectivity.
- Network reliability.
- Distributed computing.
- Egalitarian stable matching.
- Network intrusion detection.
- Multi-camera scene reconstruction.
- Data privacy.
- Many many more …

Flow Network

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.



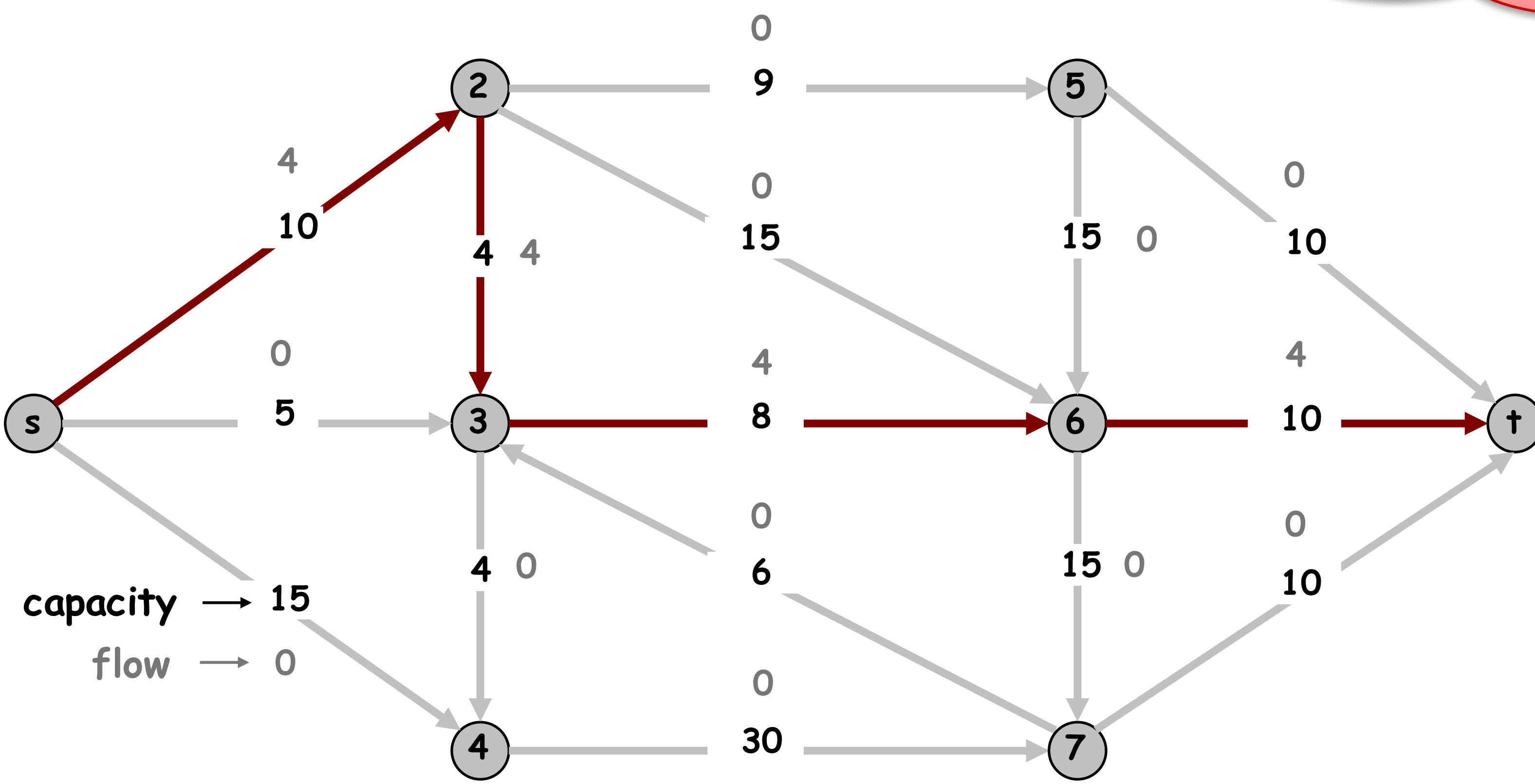
Flows

Def. An s-t flow is a function f from E to real numbers that satisfies:

- For each e ∈ E: $0 \leq f(e) \leq c(e)$ [capacity]
- For each v ∈ V – {s, t}: $\sum_{e \text{ into } v} f(e) = \sum_{e \text{ out of } v} f(e)$ [conservation]

* flow: abstract entity generated at source, transmitted across edges, absorbed at sink
* assume no arcs enter s or leave t (no loss of generality)

water flowing **from source to sink**



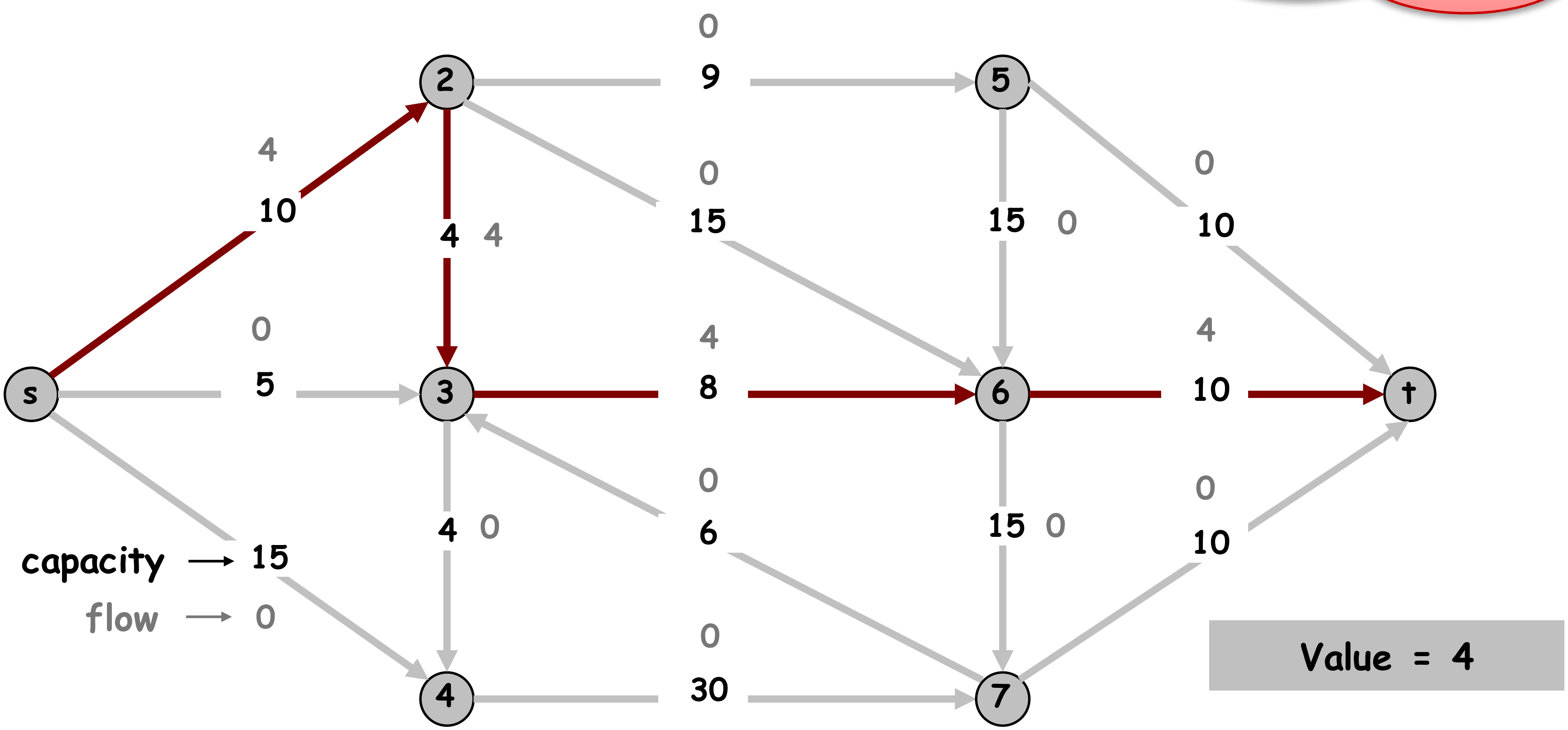
Flows

Def. An s-t flow is a function f from E to real numbers that satisfies:

- For each $e \in E$: $\quad 0 \leq f(e) \leq c(e)$ [capacity]
- For each $v \in V - \{s, t\}$: $\quad \displaystyle\sum_{e \text{ into } v} f(e) = \sum_{e \text{ out of } v} f(e)$ [conservation]

Def. The value of a flow f is: $\quad v(f) = \displaystyle\sum_{e \text{ out of } s} f(e)$

water flowing
from source to sink



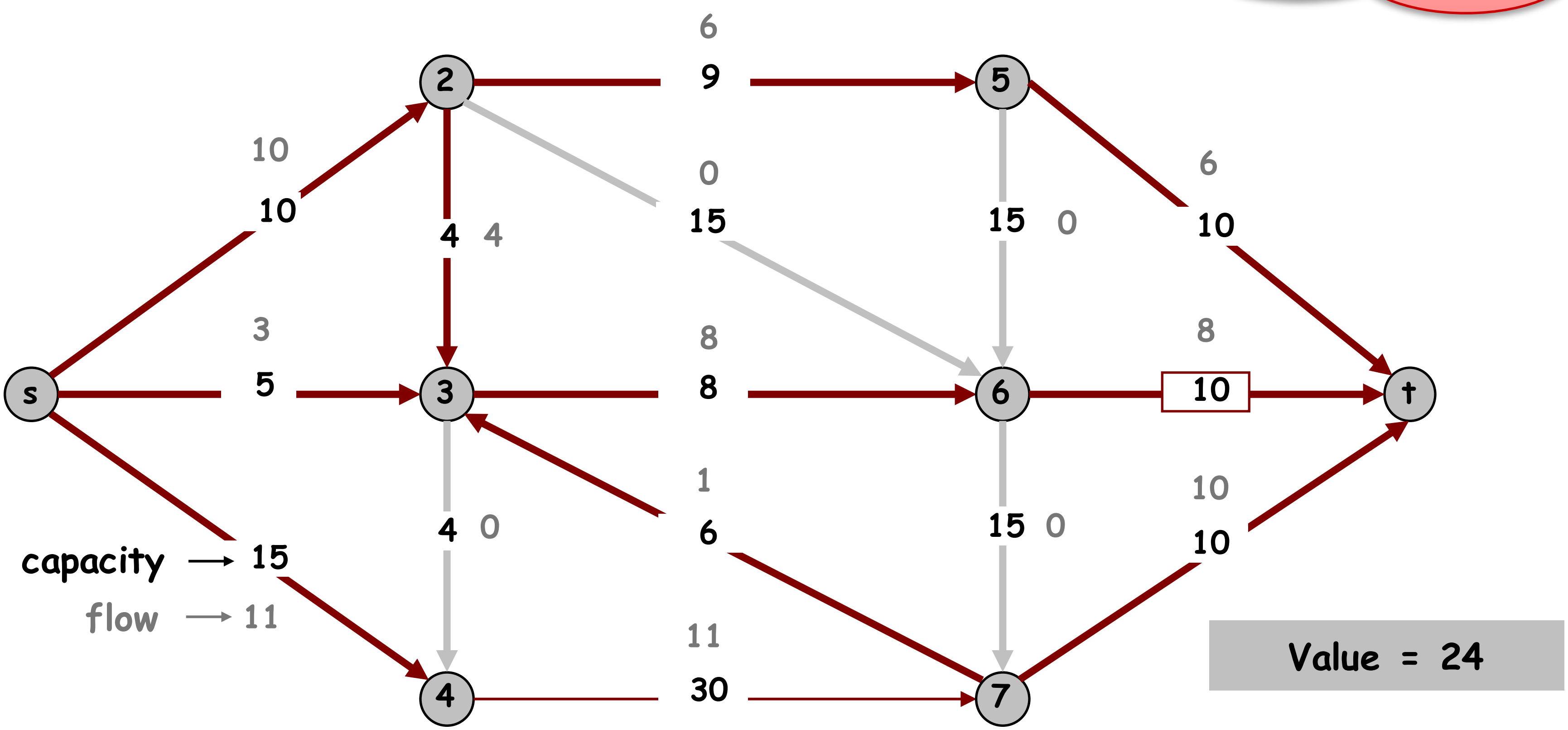
Flows

Def. An s-t flow is a function f from E to real numbers that satisfies:

- For each $e \in E$: $0 \leq f(e) \leq c(e)$ [capacity]
- For each $v \in V - \{s, t\}$: $\sum_{e \text{ into } v} f(e) = \sum_{e \text{ out of } v} f(e)$ [conservation]

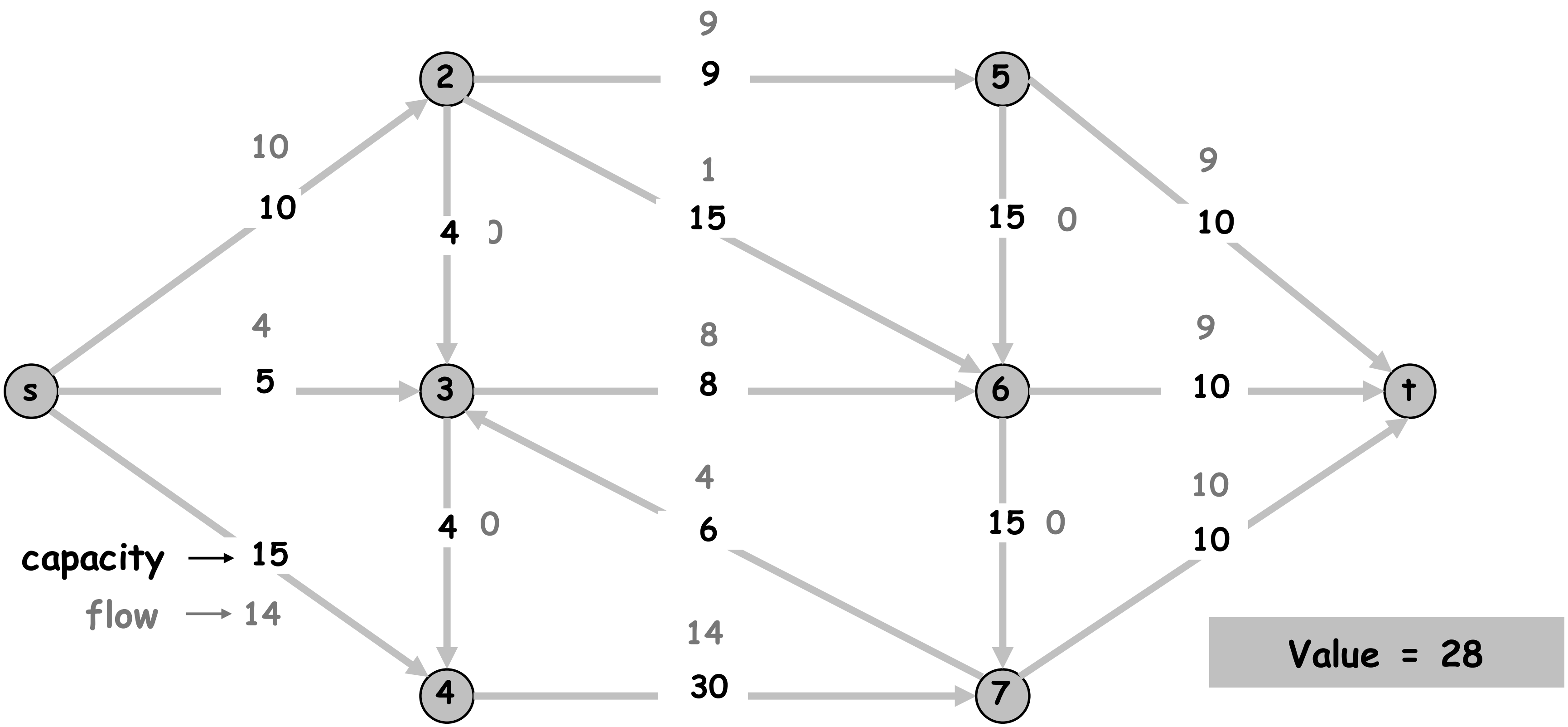
Def. The value of a flow f is: $v(f) = \sum_{e \text{ out of } s} f(e)$.

water flowing
from source to sink



Maximum Flow Problem

Max flow problem. Find s-t flow of maximum value.

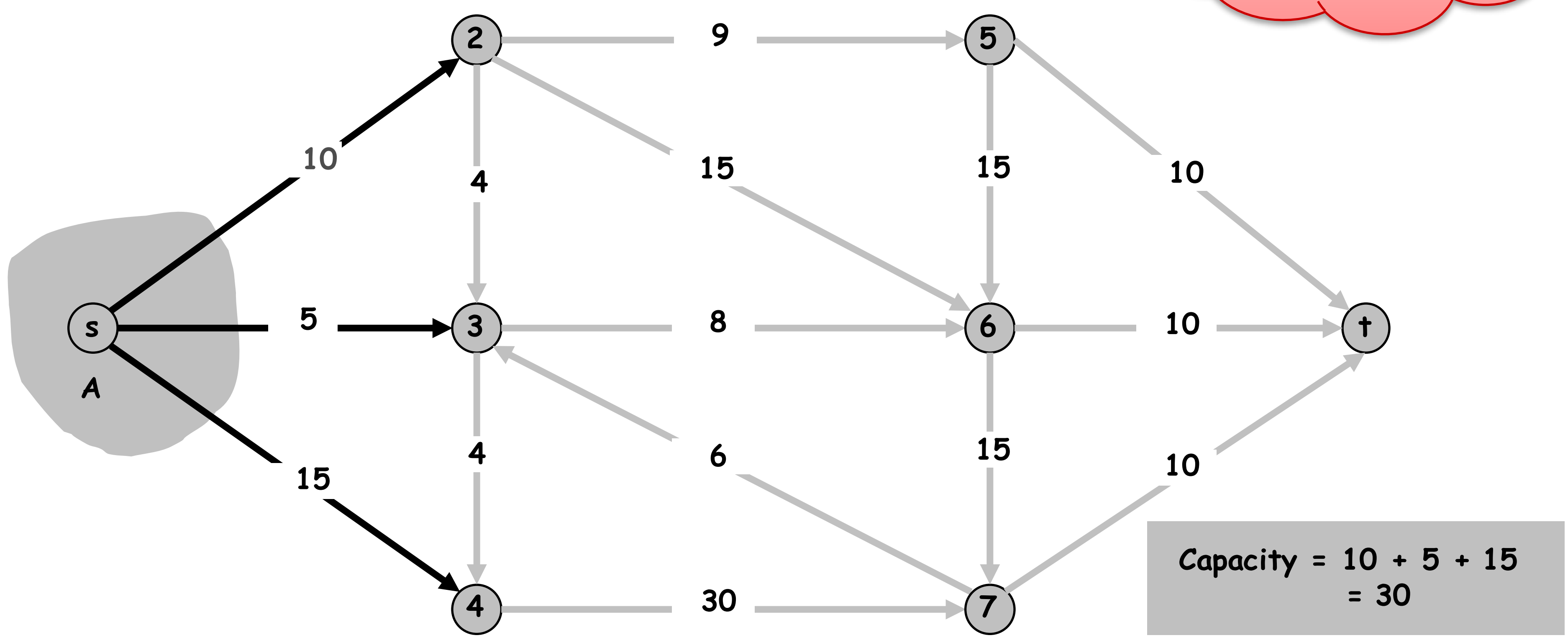


Cuts

Def. An s-t cut is a partition (A, B) of V with s ∈ A and t ∈ B.

Def. The capacity of a cut (A, B) is: $cap(A, B) = \sum_{e \text{ out of } A} c(e)$

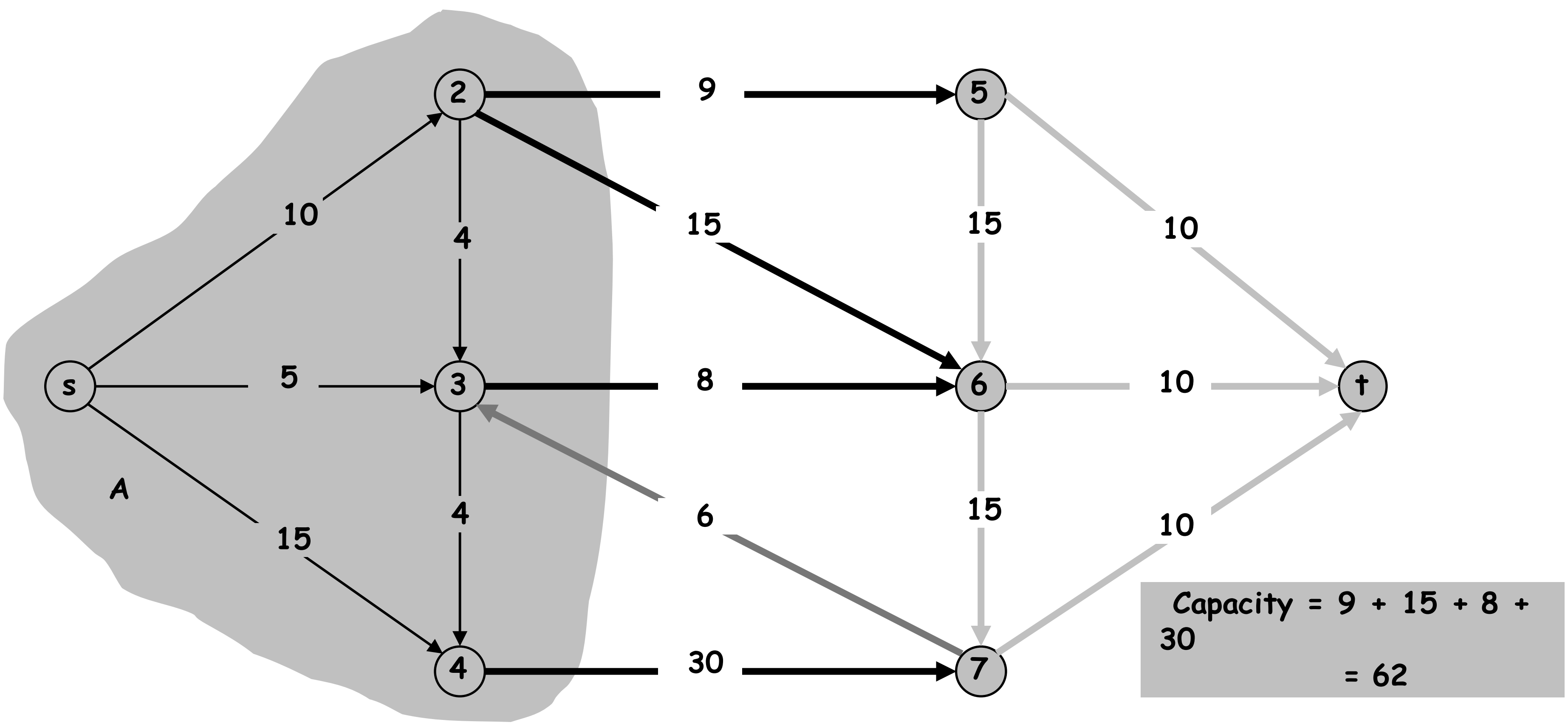
we don't count edges **into** A



Cuts

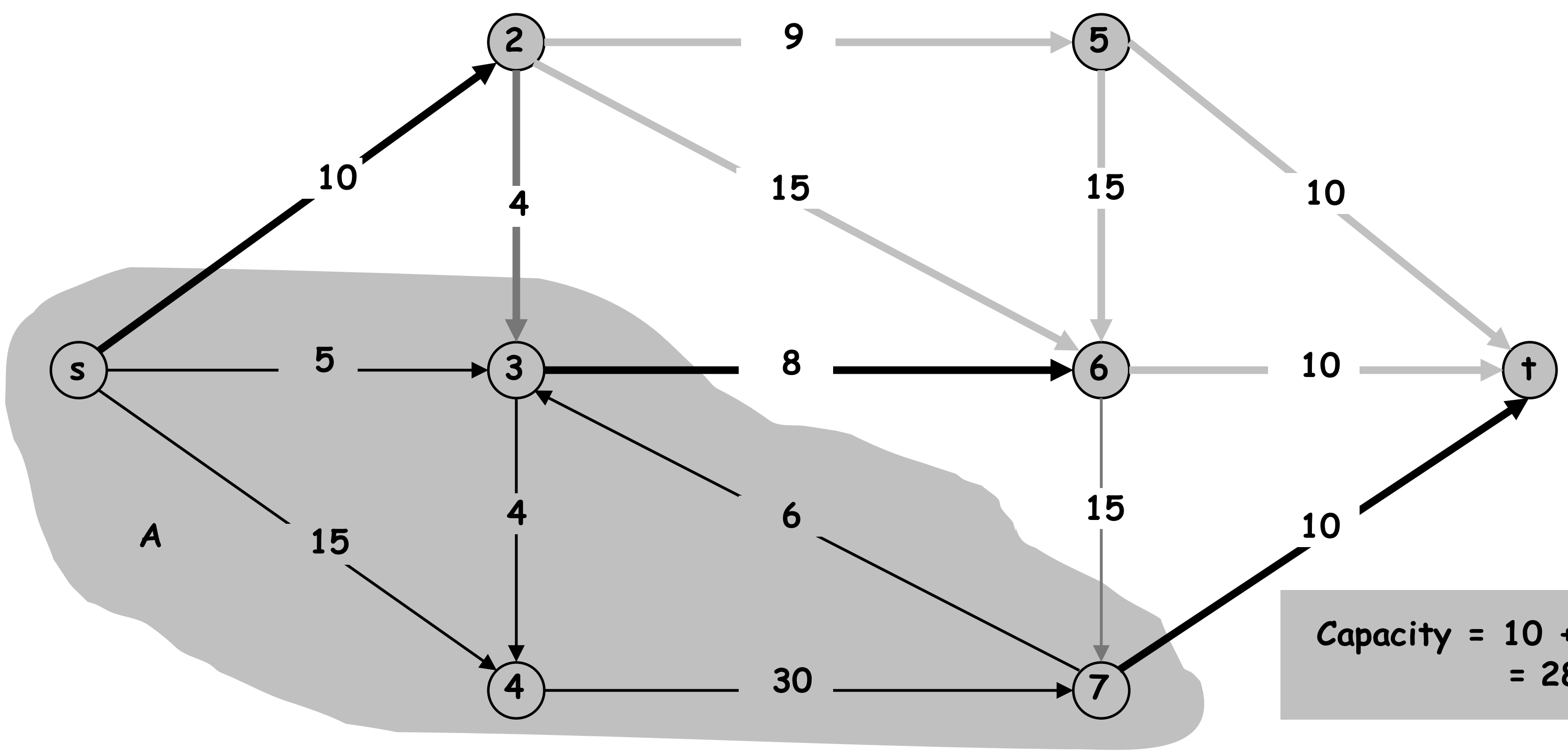
Def. An s-t cut is a partition (A, B) of V with s ∈ A and t ∈ B.

Def. The capacity of a cut (A, B) is: $cap(A, B) = \sum_{e \text{ out of } A} c(e)$



Minimum Cut Problem

Min s-t cut problem. Find an s-t cut of minimum capacity.

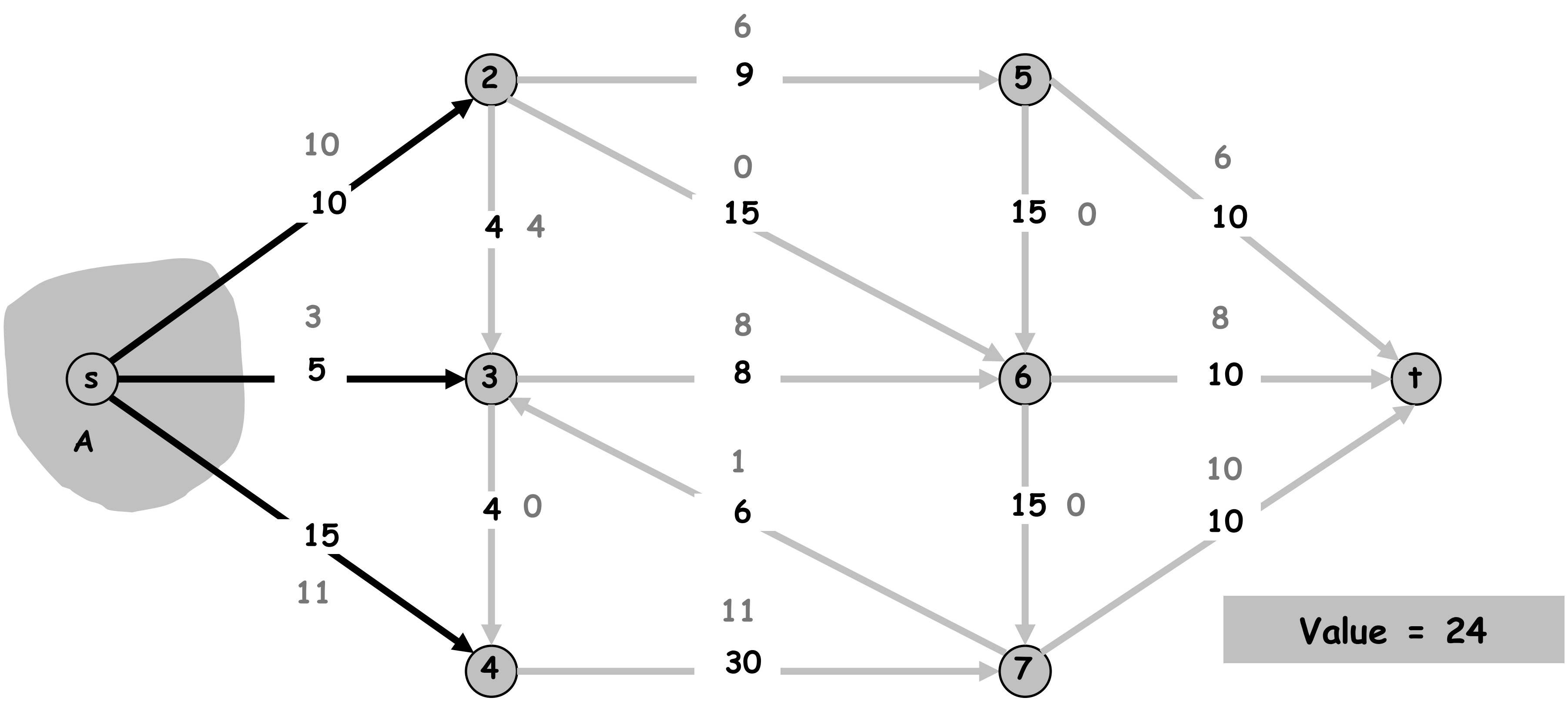


Capacity = 10 + 8 + 10
= 28

Flows and Cuts

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.

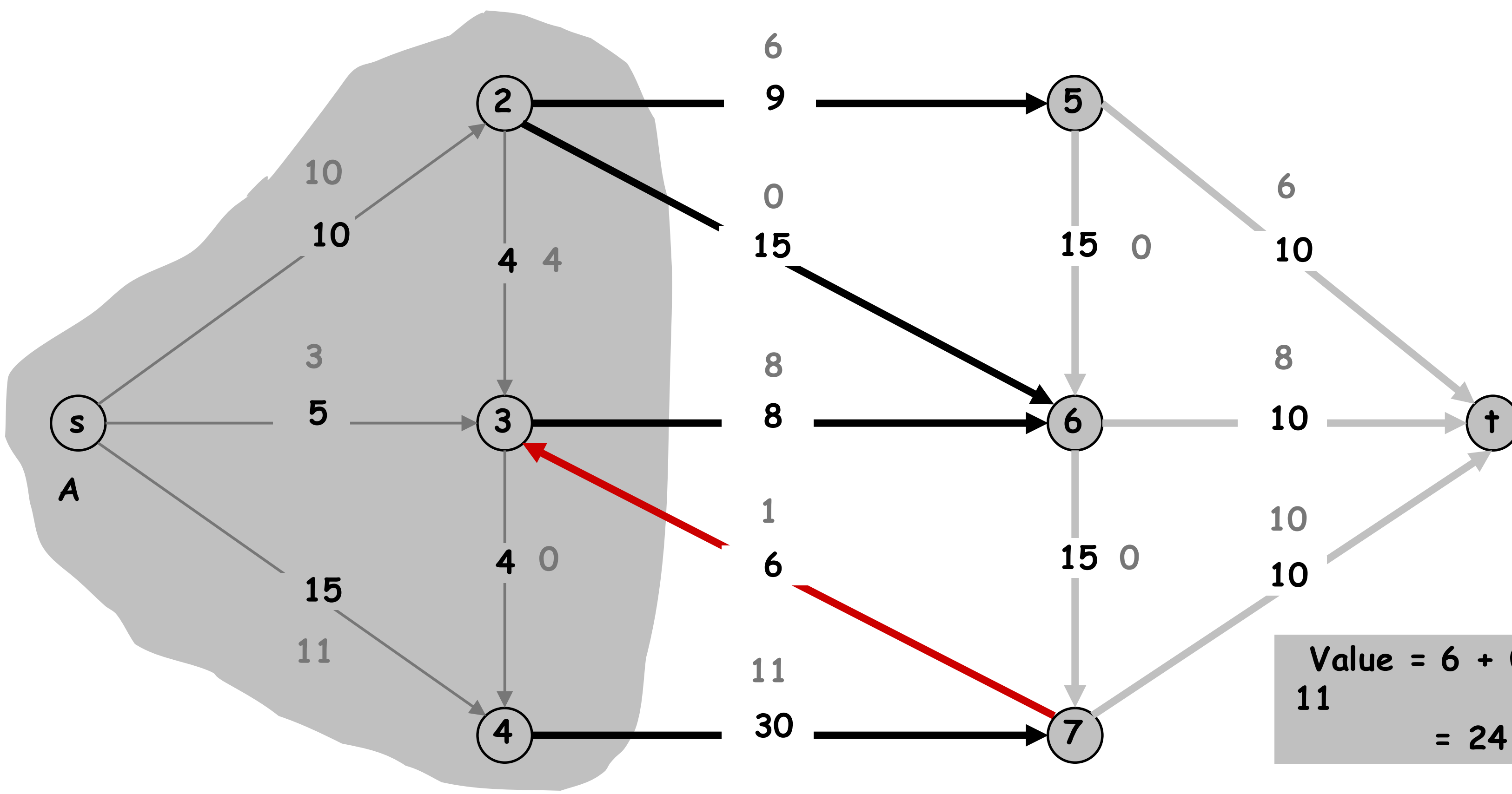
$$\sum_{e \text{ out of } A} f(e) - \sum_{e \text{ in to } A} f(e) = v(f)$$



Flows and Cuts

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.

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

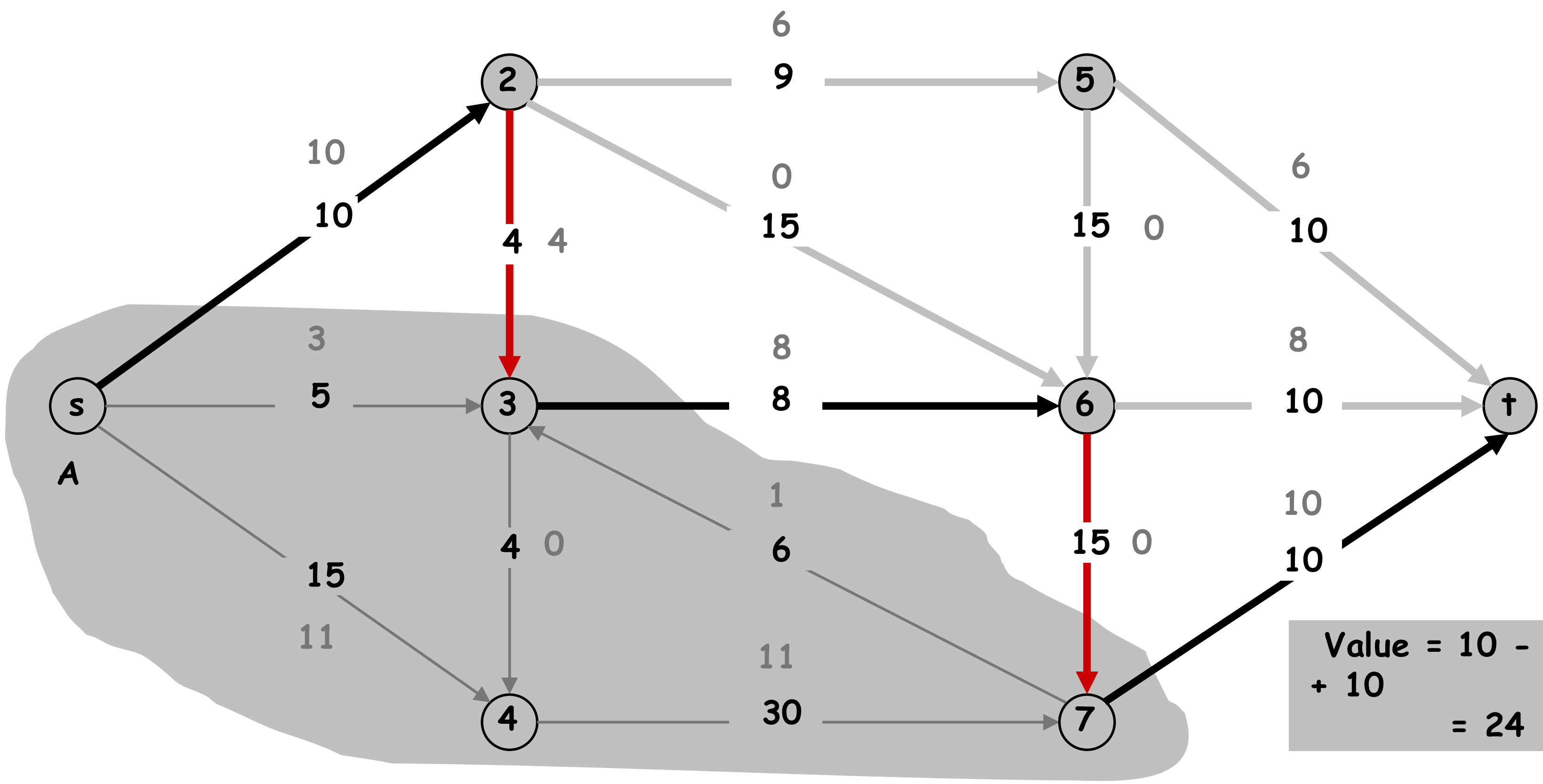


Value = 6 + 0 + 8 - 1 + 11
= 24

Flows and Cuts

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.

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



Value = 10 - 4 + 8 - 0 + 10
= 24

Flows and Cuts

Flow value lemma. Let f be any flow, and let (A, B) be any s-t cut. Then

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

Proof.

$$v(f) = \sum_{e \text{ out of } s} f(e)$$

by flow conservation, all terms except v = s are 0 →

$$= \sum_{v \in A} \left(\sum_{e \text{ out of } v} f(e) - \sum_{e \text{ in to v}} f(e) \right)$$

Ignore all edges that are between two nodes in A, since they appear for both end points with opposite signs →

$$= \sum_{e \text{ out of } A} f(e) - \sum_{e \text{ in to A}} f(e).$$

Question

Two problems

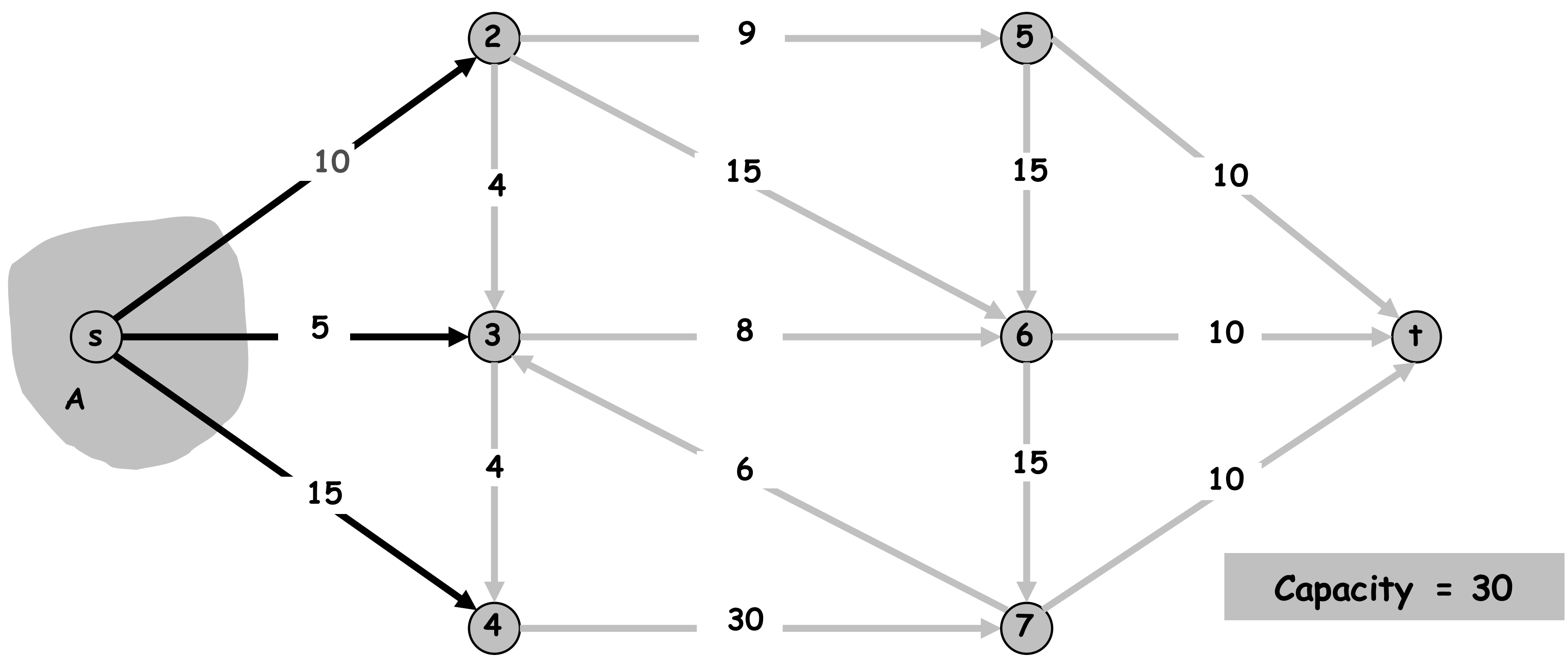
- min cut
- max flow

- How do they relate?

Flows and Cuts

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.

Cut capacity = 30 $\Rightarrow$ Flow value $\leq$ 30



Flows and Cuts

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.

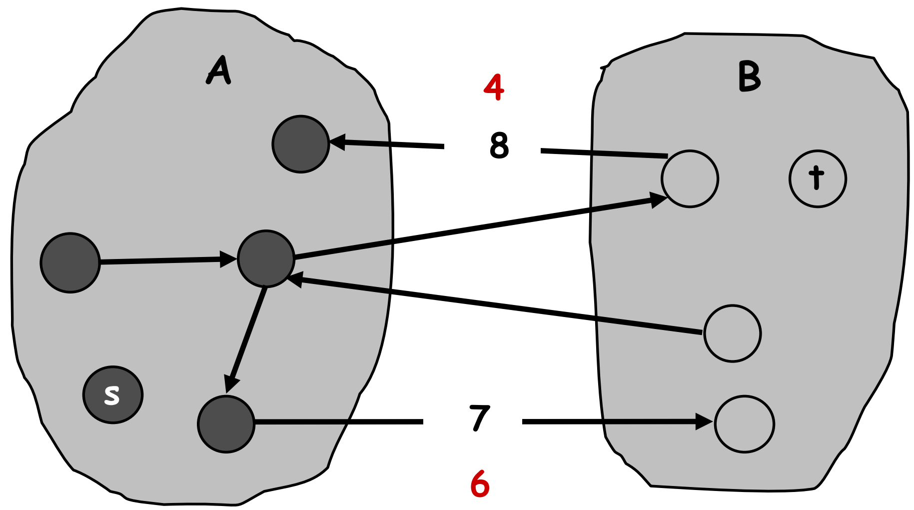
Pf.

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

$$\leq \sum_{e \text{ out of } A} f(e)$$

$$\leq \sum_{e \text{ out of } A} c(e)$$

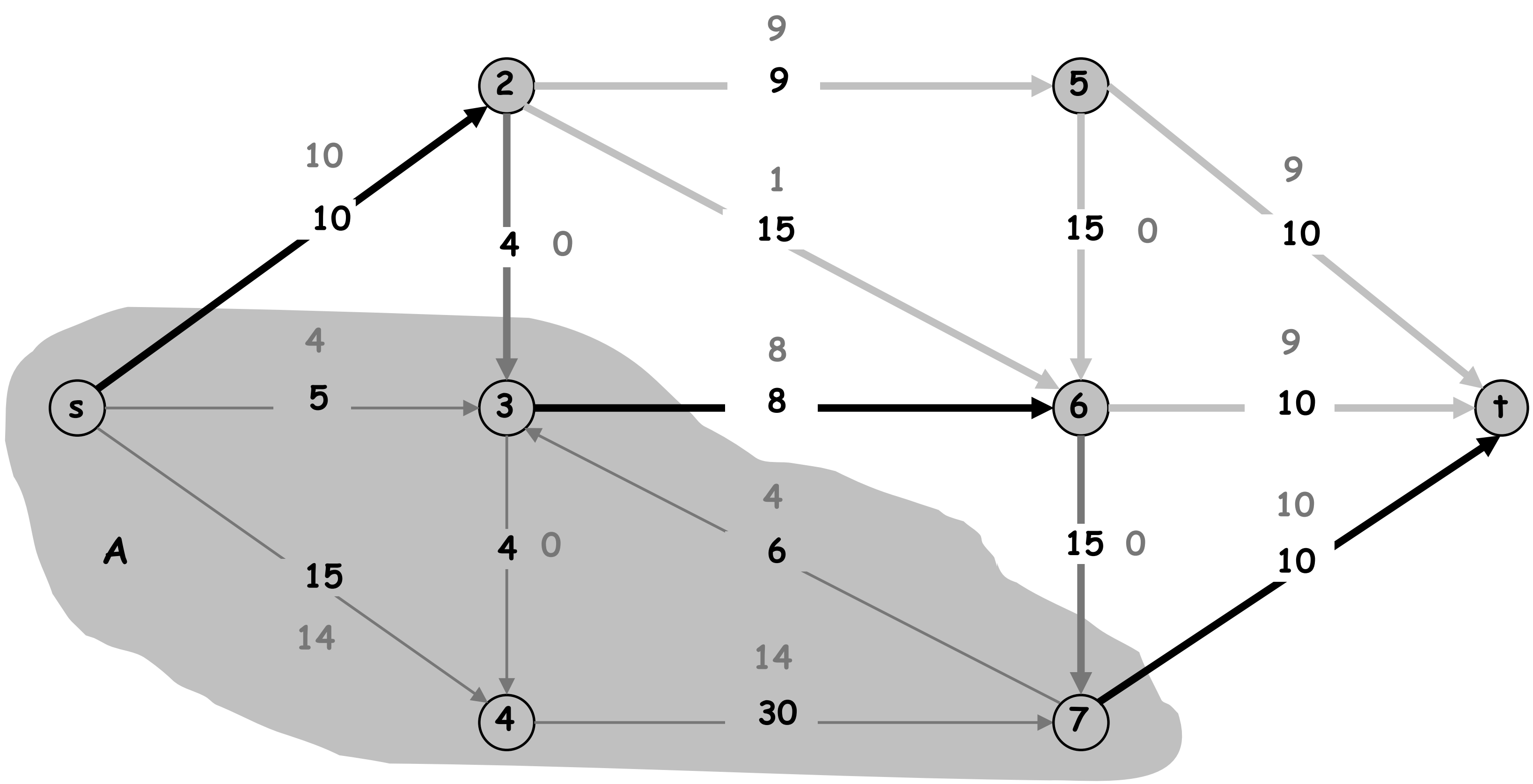
$$= \text{cap}(A,B)$$



Certificate of Optimality

Corollary. 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) is a min s-t cut.

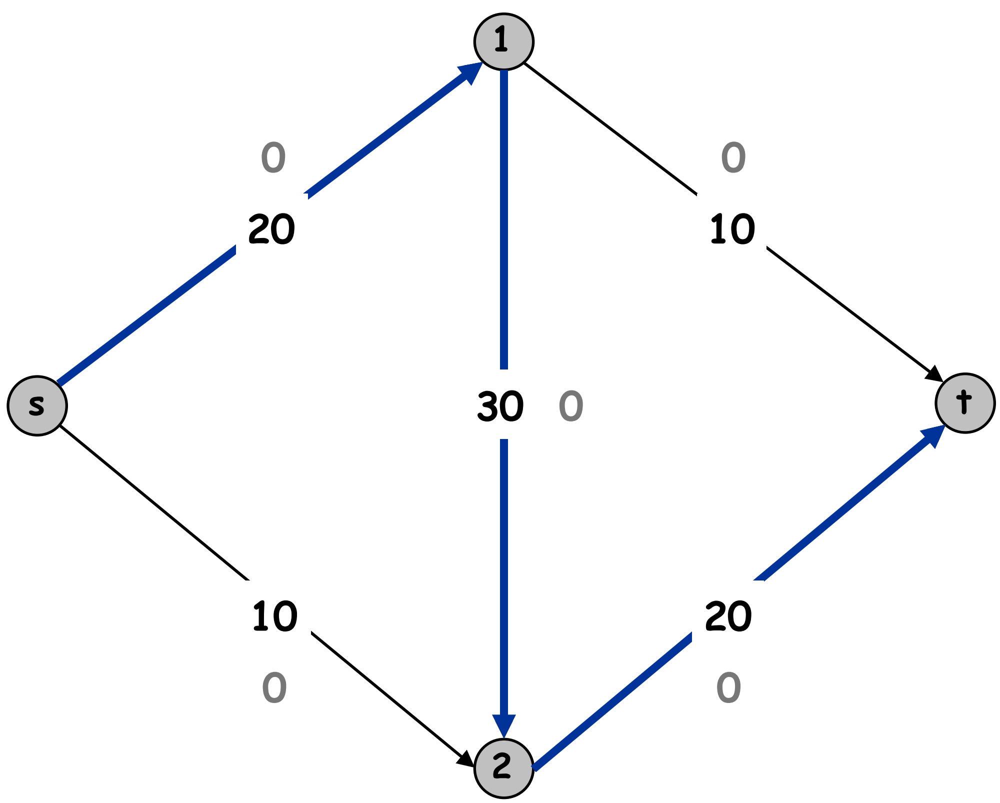
Value of flow = 28
Cut capacity = 28 $\Rightarrow$ Flow value $\leq$ 28



Towards a Max Flow Algorithm

Greedy algorithm.

- Start with $f(e) = 0$ for all edge $e \in E$.
- Find an s-t path P where each edge has $f(e) < c(e)$.
- Augment flow along path P.
- Repeat until you get stuck.

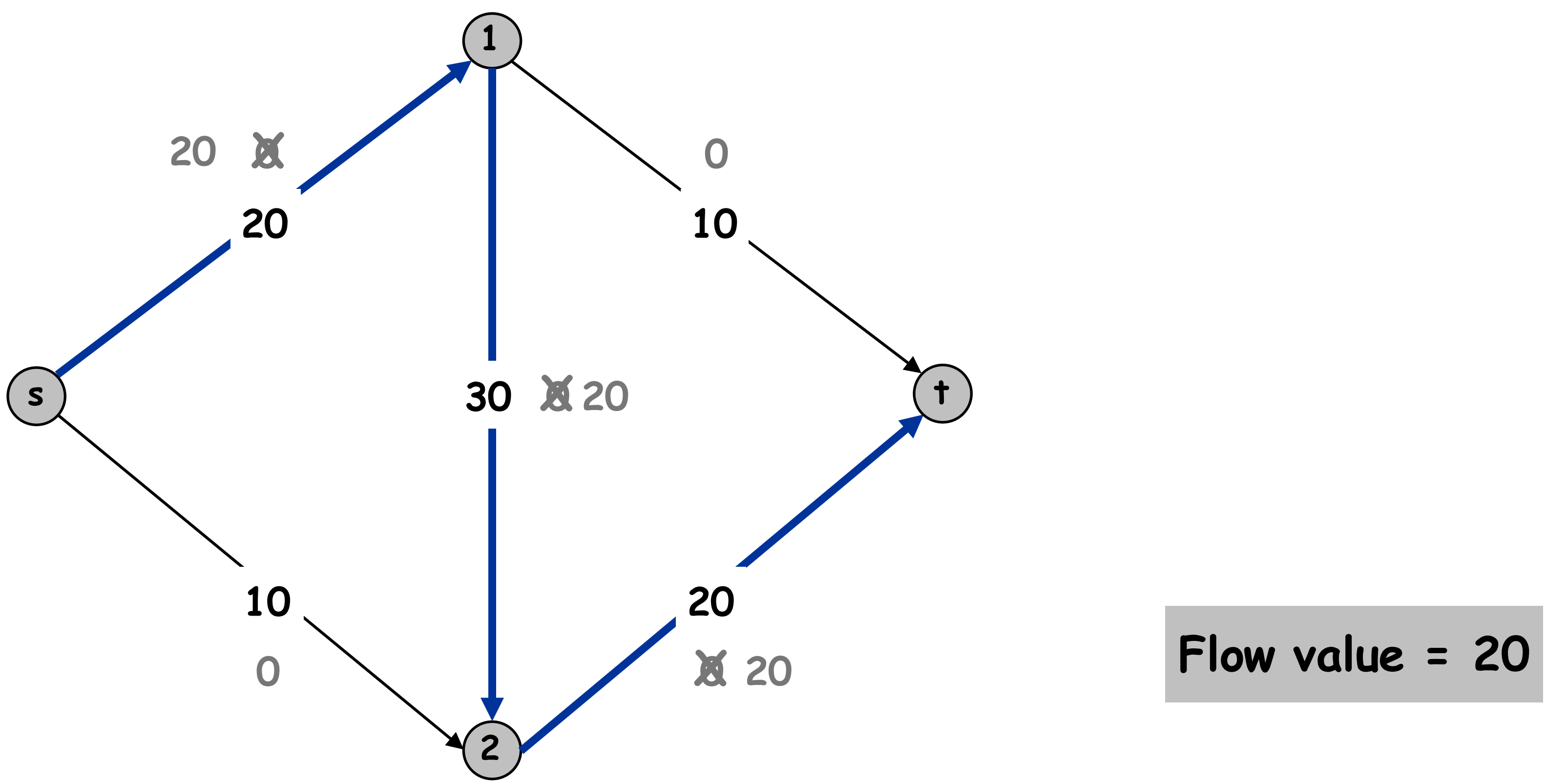


Flow value = 0

Towards a Max Flow Algorithm

Greedy algorithm.

- Start with $f(e) = 0$ for all edge $e \in E$.
- Find an s-t path P where each edge has $f(e) < c(e)$.
- Augment flow along path P.
- Repeat until you get stuck.



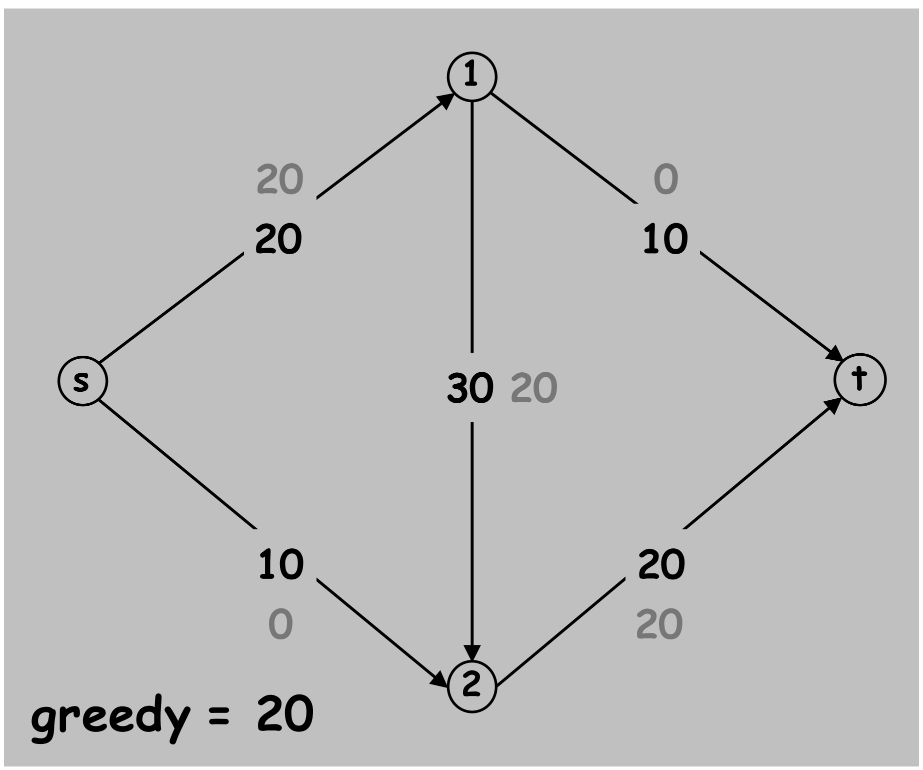
Towards a Max Flow Algorithm

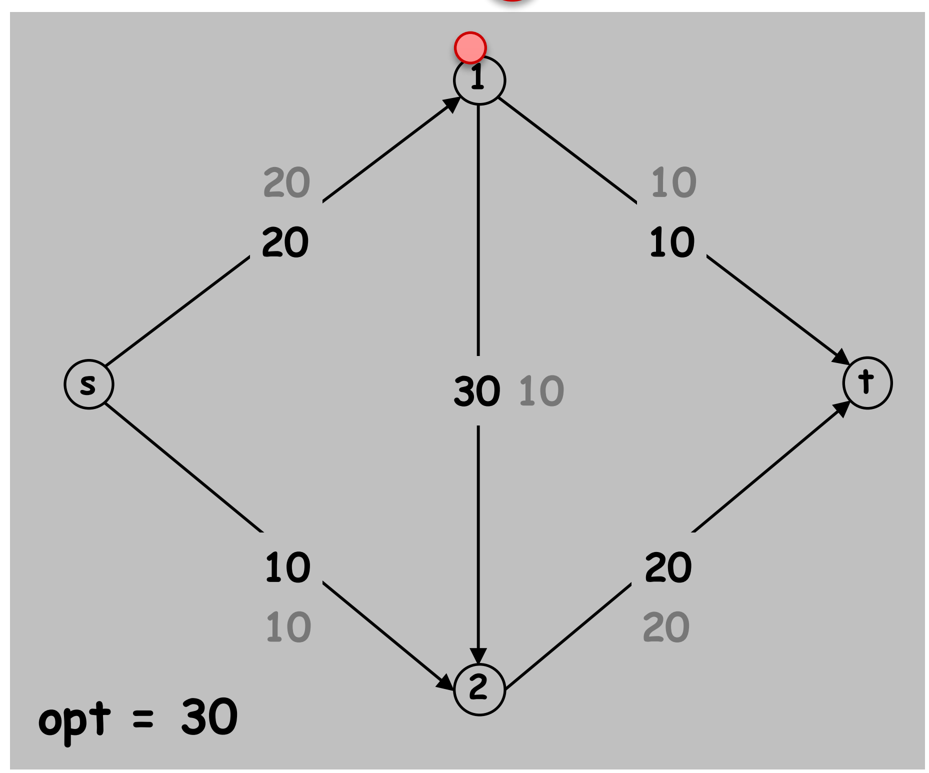
How can we get from greedy to opt here? What if we **push water back** across middle edge?

Greedy algorithm.

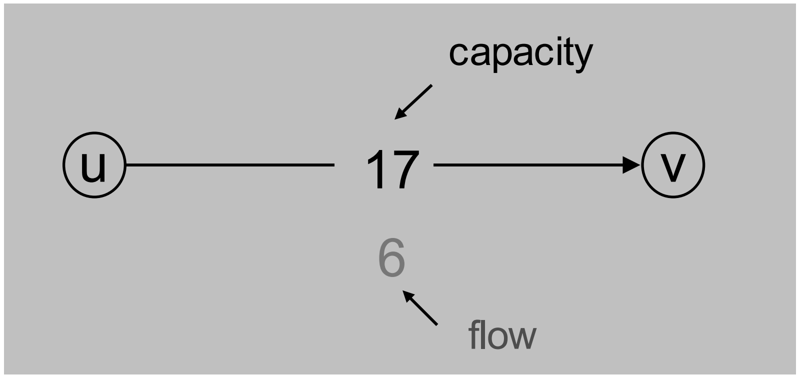
- Start with $f(e) = 0$ for all edge $e \in E$.
- Find an s-t path P where each edge has $f(e) < c(e)$.
- Augment flow along path P.
- Repeat until you get stuck.

↖ locally optimality $\not\Rightarrow$ global optimality





Residual Graph



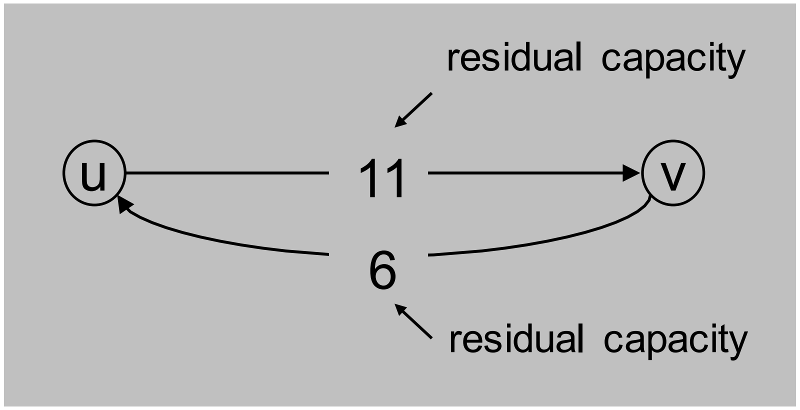
Original edge: $e = (u, v) \in E$.

- Flow $f(e)$, capacity $c(e)$.

Residual edge.

- "Undo" flow sent.
- Any forward edge $e = (u, v)$ and backward edge $e^R = (v, u)$.
- Residual capacity:

$$c_f(e) = \begin{cases} c(e) - f(e) & \text{if } e \in E \\ f(e) & \text{if } e^R \in E \end{cases}$$

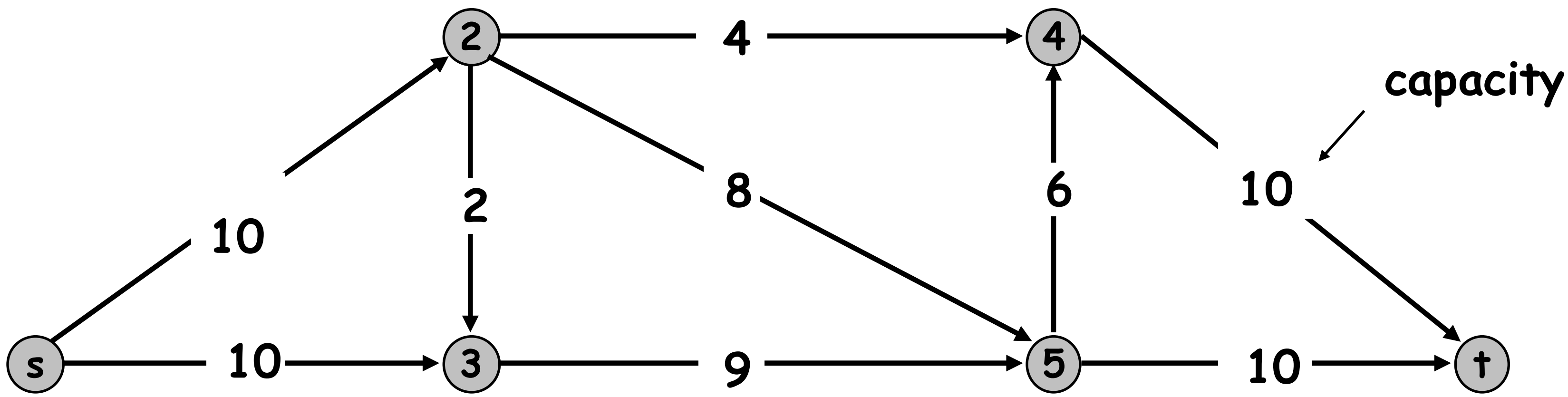


Residual graph: $G_f = (V, E_f)$.

- Residual edges with nonzero/positive residual capacity.
- $E_f = \{e : c_f(e) > 0\}$, i.e. $E_f = \{e : f(e) < c(e)\} \cup \{e^R : f(e) > 0\}$.
- Edges in E with flow=capacity are only present in reverse direction in E_f
- Edges in E with no flow are only present in their original direction in E_f

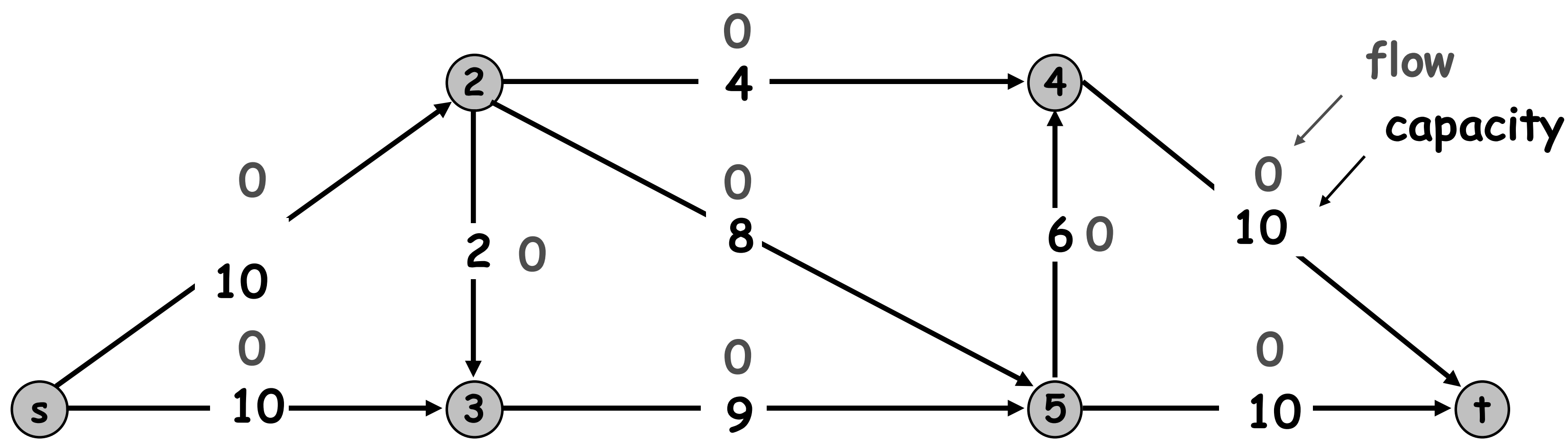
Ford-Fulkerson Algorithm

G:



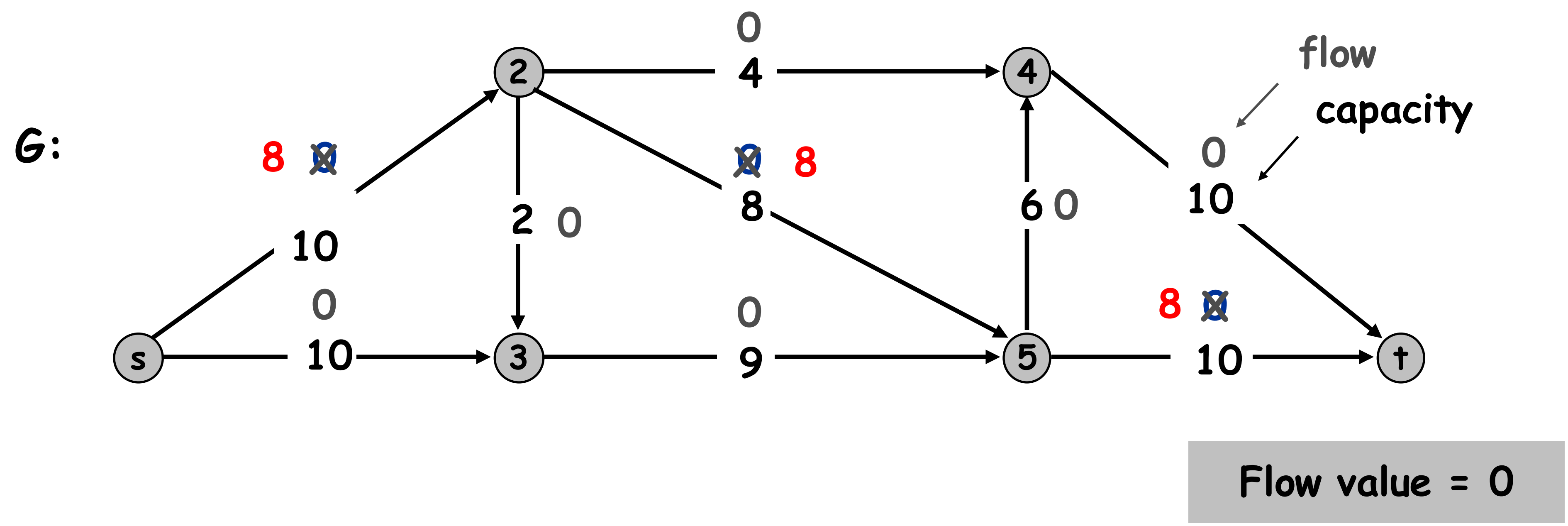
Ford-Fulkerson Algorithm

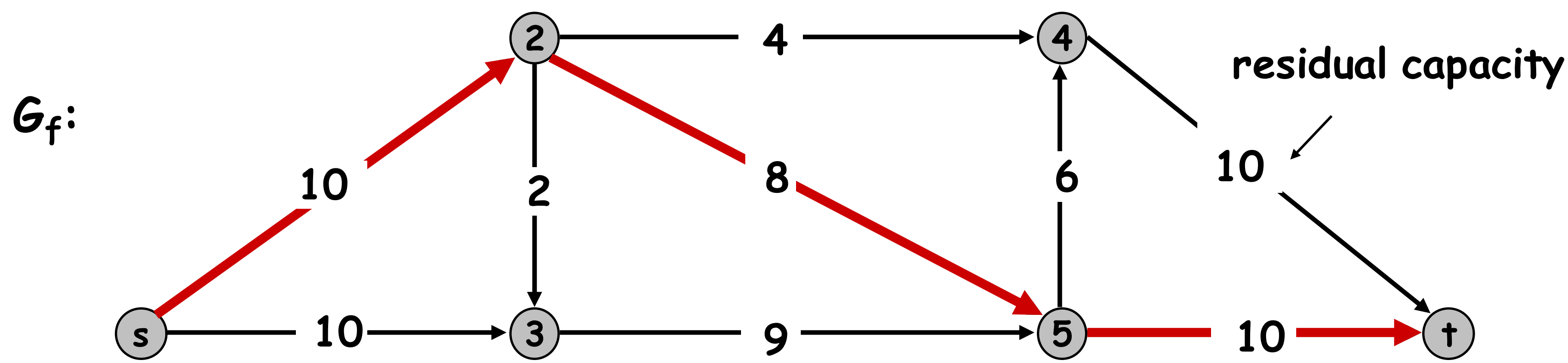
G:



Flow value = 0

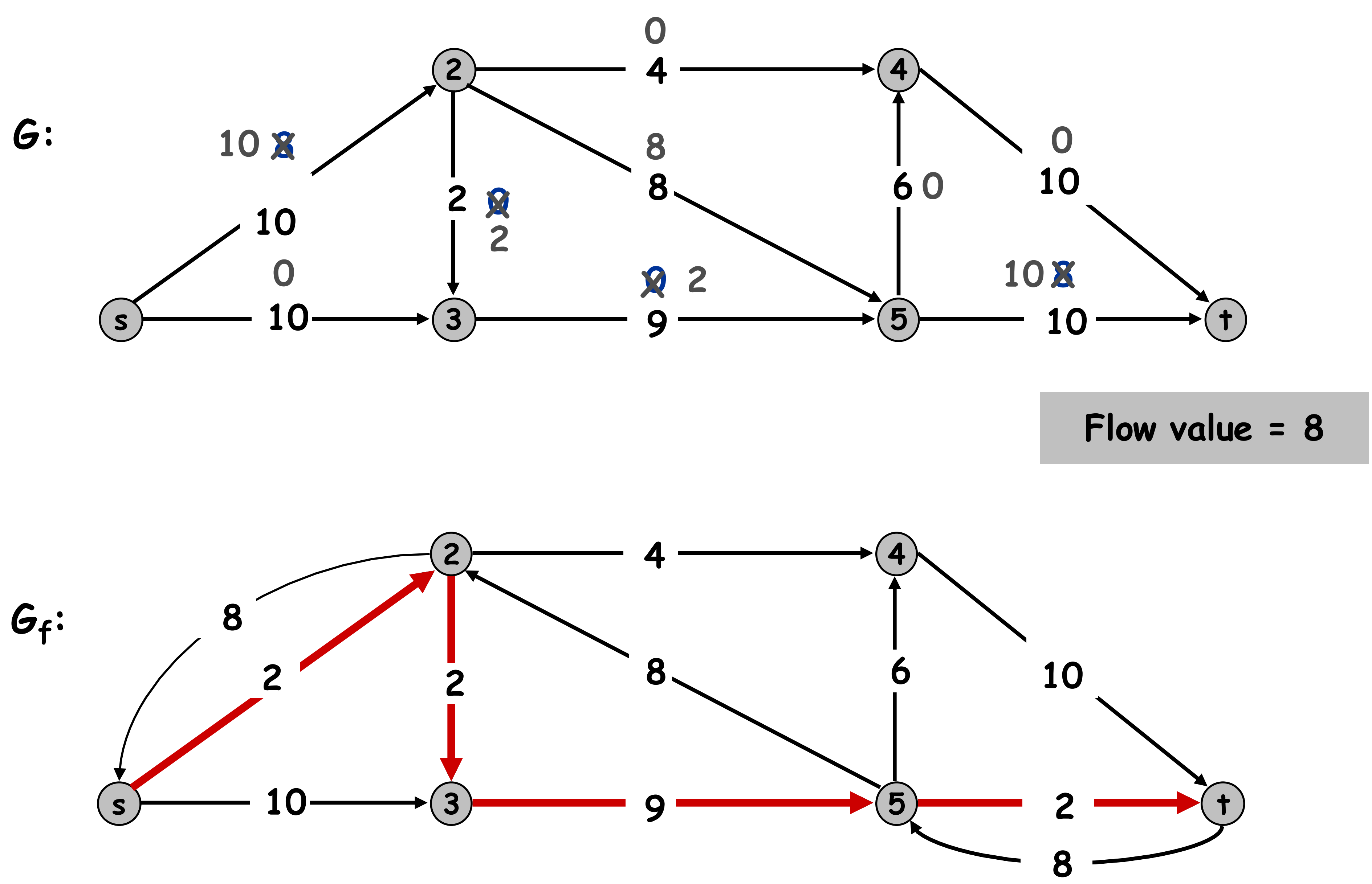
Ford-Fulkerson Algorithm



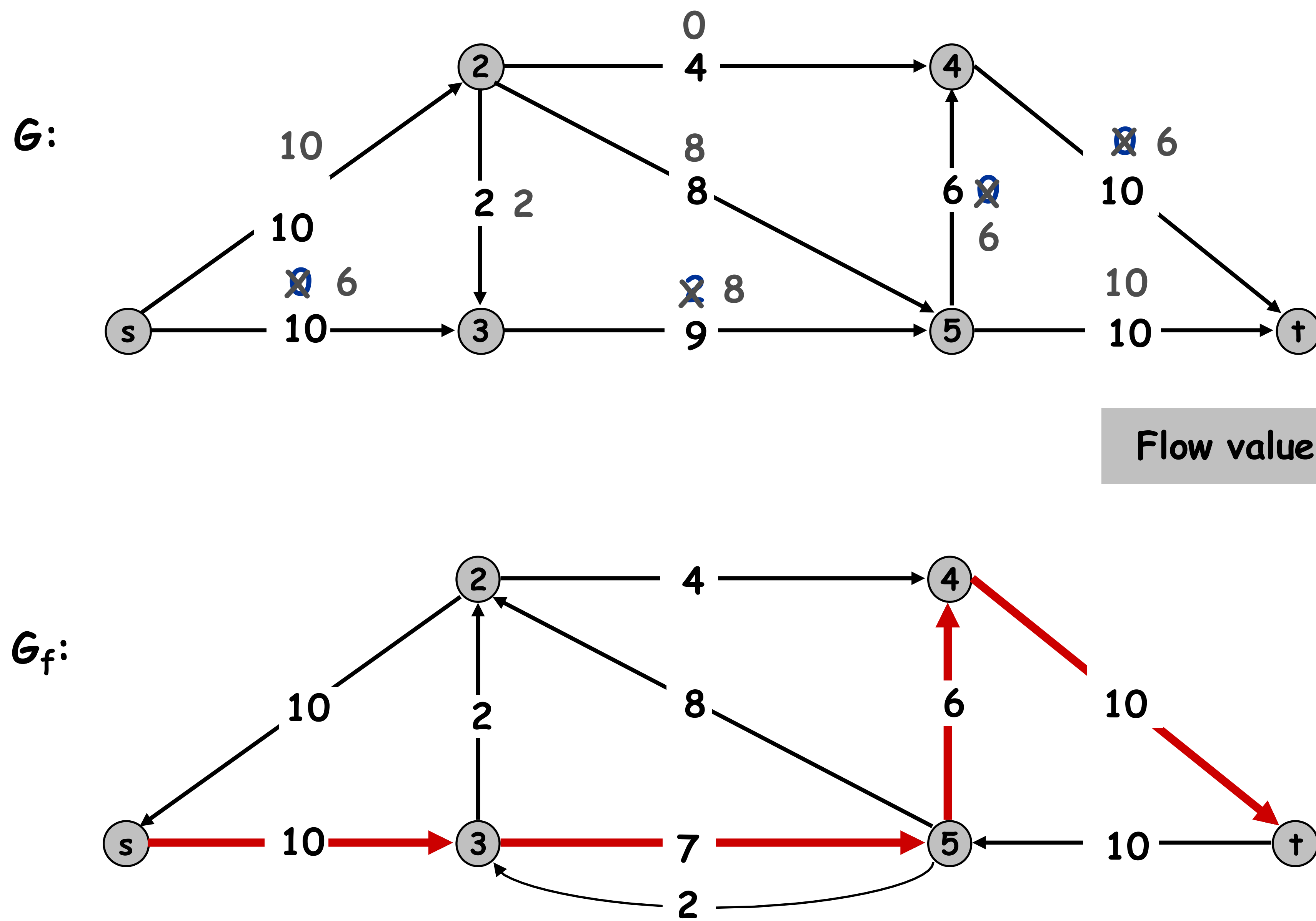


Bottleneck along red path?

Ford-Fulkerson Algorithm



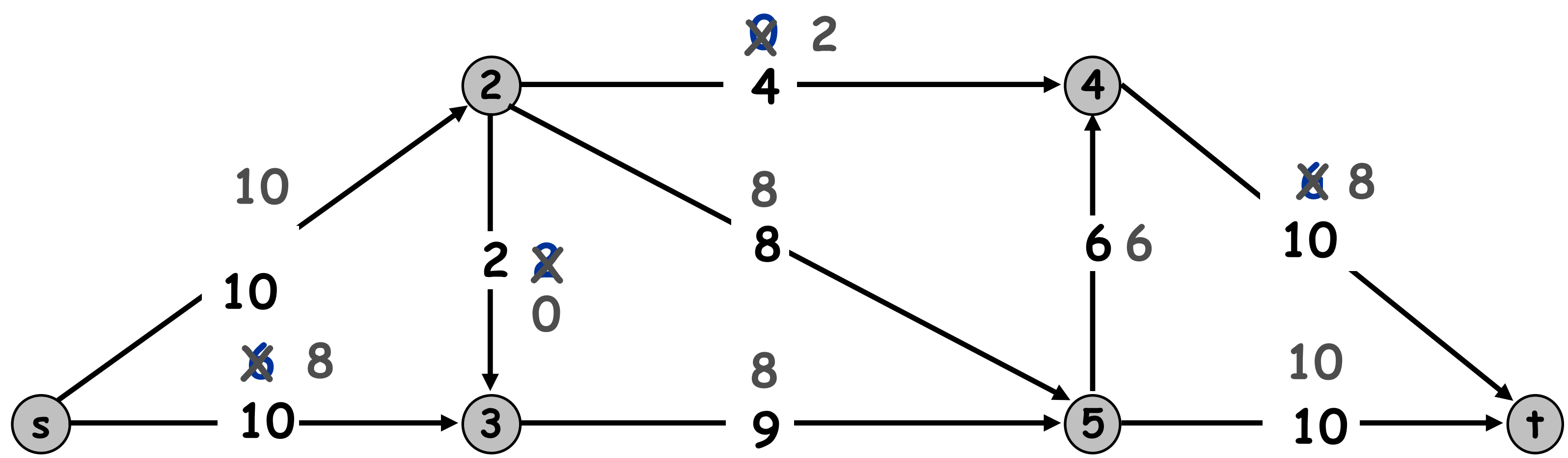
Ford-Fulkerson Algorithm



Flow value = 10

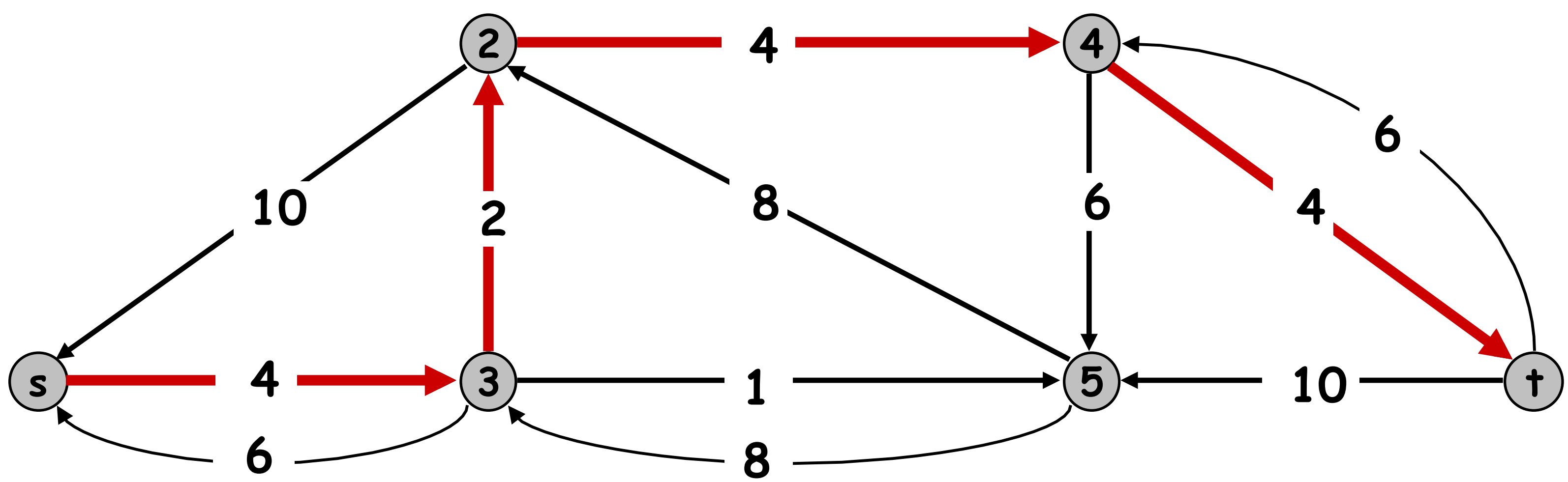
Ford-Fulkerson Algorithm

G:

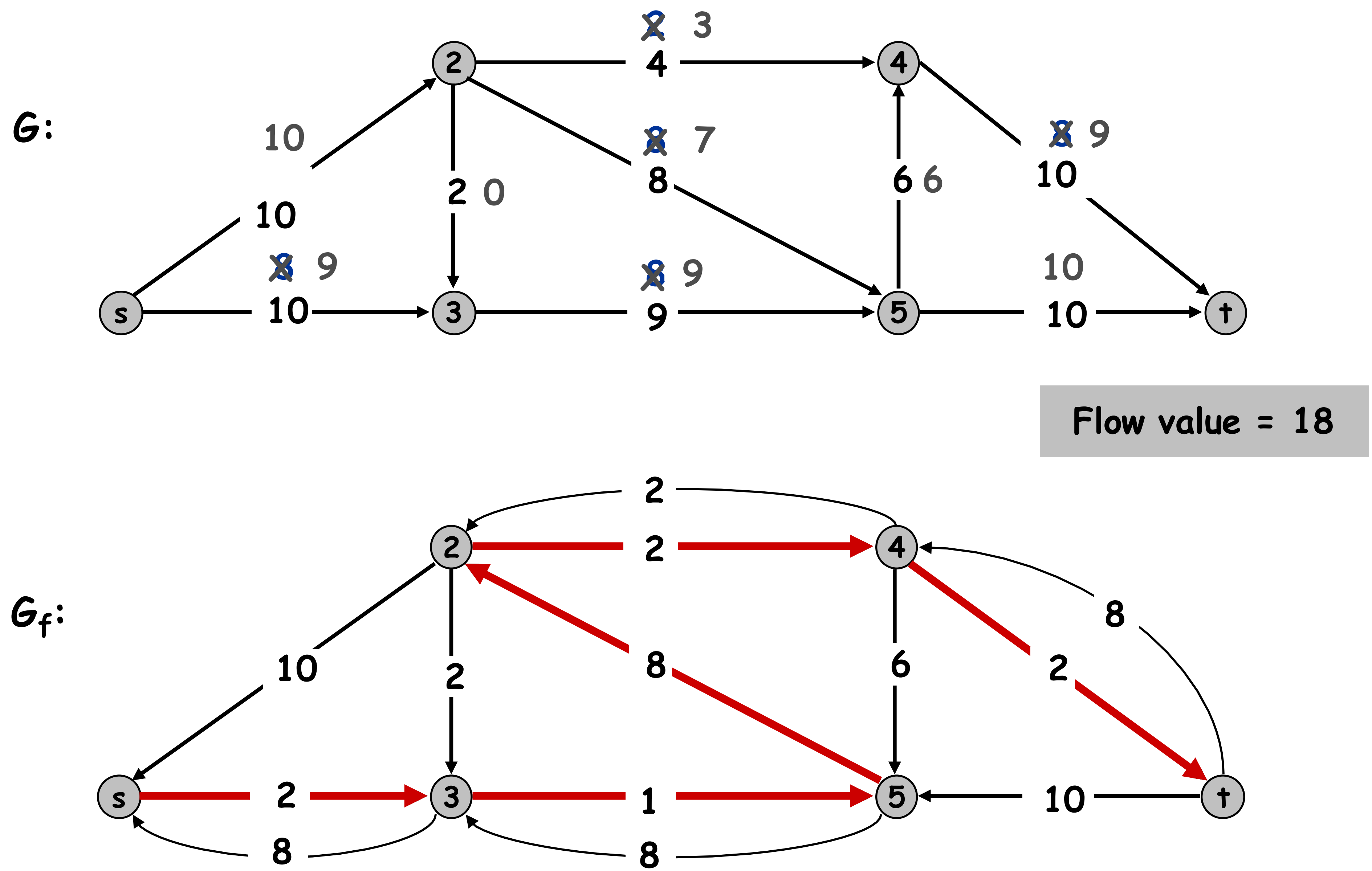


Flow value = 16

G_f:

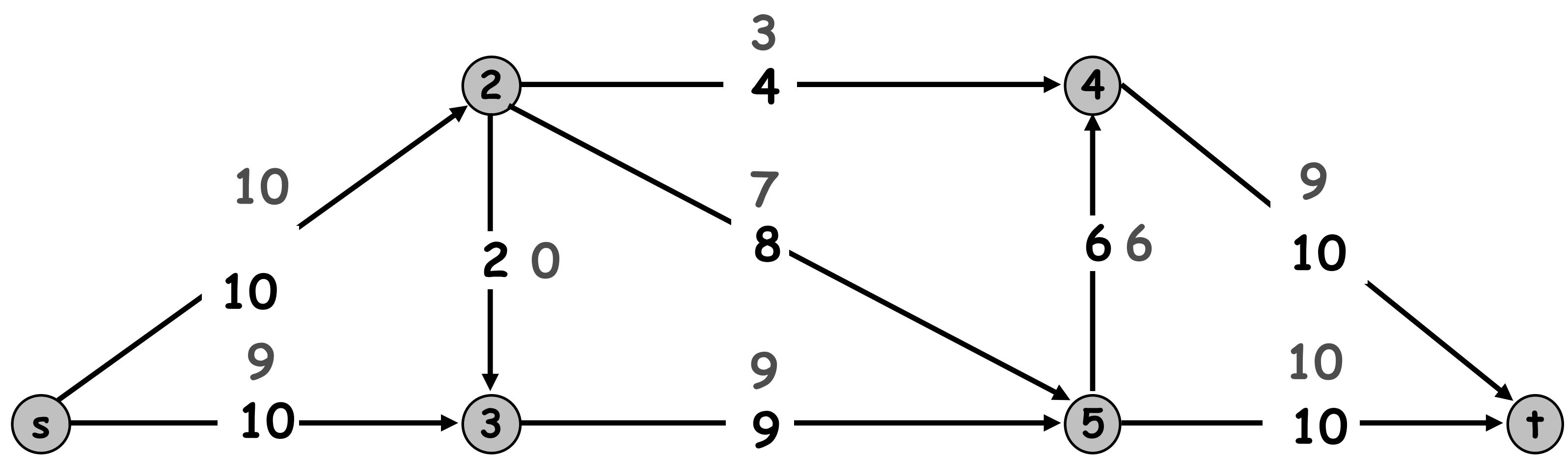


Ford-Fulkerson Algorithm



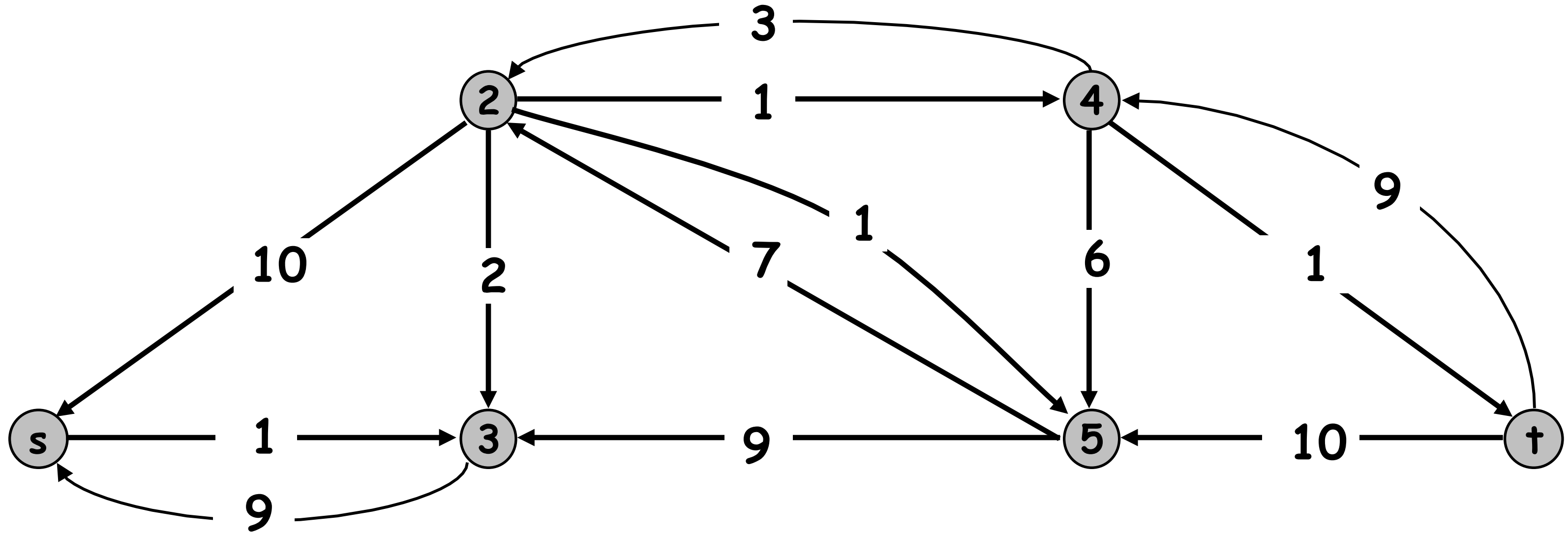
Ford-Fulkerson Algorithm

G:

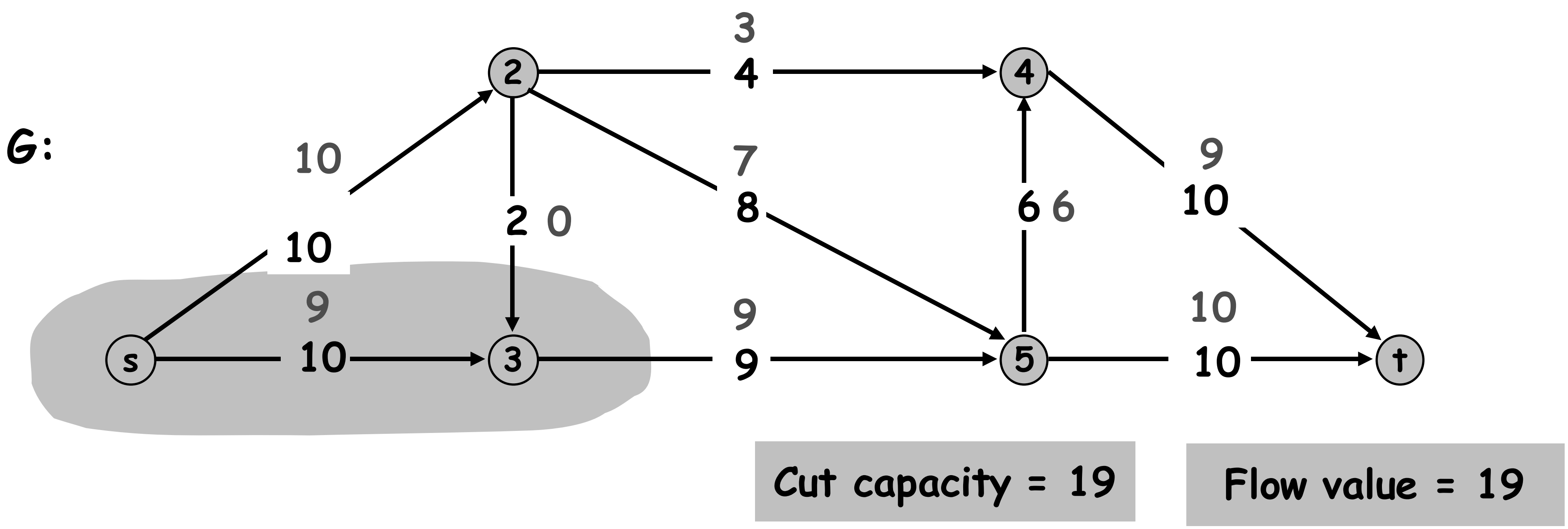


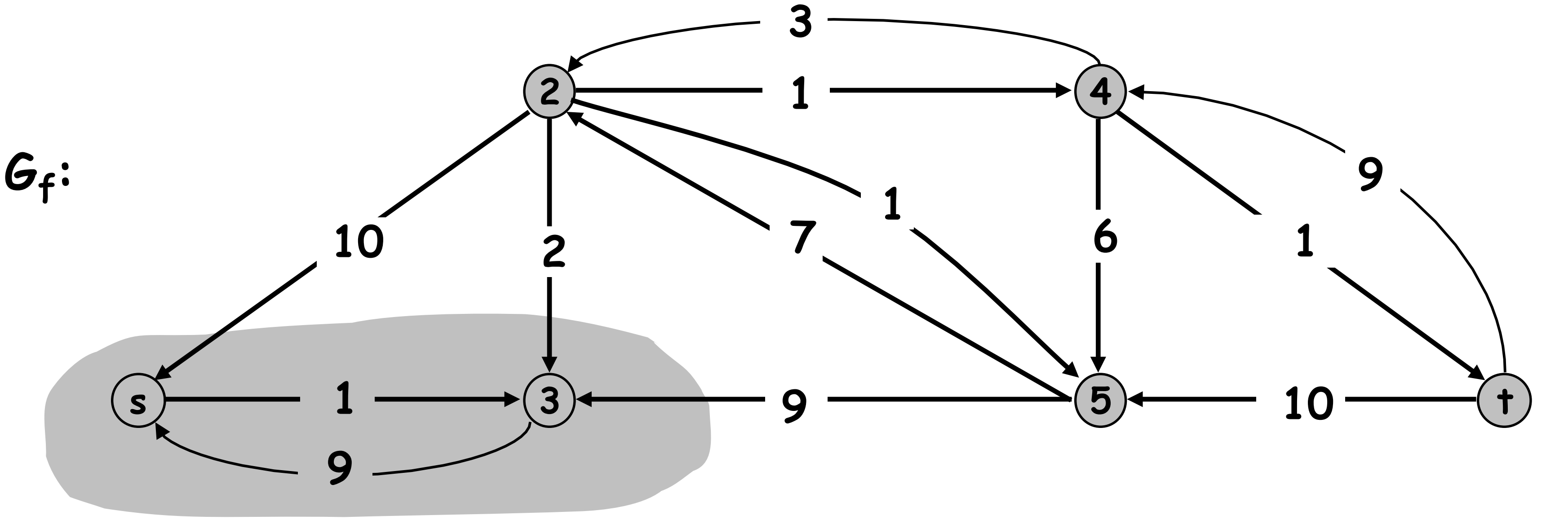
Flow value = 19

G_f:



Ford-Fulkerson Algorithm





Augmenting Path Algorithm

```
Ford-Fulkerson(G, s, t, c) {
   foreach e ∈ E  f(e) ← 0
   Gf ← residual graph

   while (there is an s-t path P in Gf) {
      f ← Augment(f, c, P)
      update Gf
   }
   return f
}
```

```
Augment(f, c, P) {
   b ← bottleneck(P)
   foreach e ∈ P {
      if (e ∈ E) f(e) ← f(e) + b
      else [eR ∈ E] f(eR) ← f(eR) - b
   }
   return f
}
```

Min residual capacity of an edge in P

forward edge

backward edge

Max-Flow Min-Cut Theorem

Augmenting path theorem. Flow f is a max flow iff there are no augmenting paths.

Max-flow min-cut theorem. [Elias-Feinstein-Shannon 1956, Ford-Fulkerson 1956]
The value of the max flow is equal to the value of the min s-t cut.
We have the equivalence between:

(i) There exists an s-t cut (A, B) such that v(f) = cap(A, B).
(ii) Flow f is a max flow.
(iii) There is no augmenting path relative to f.