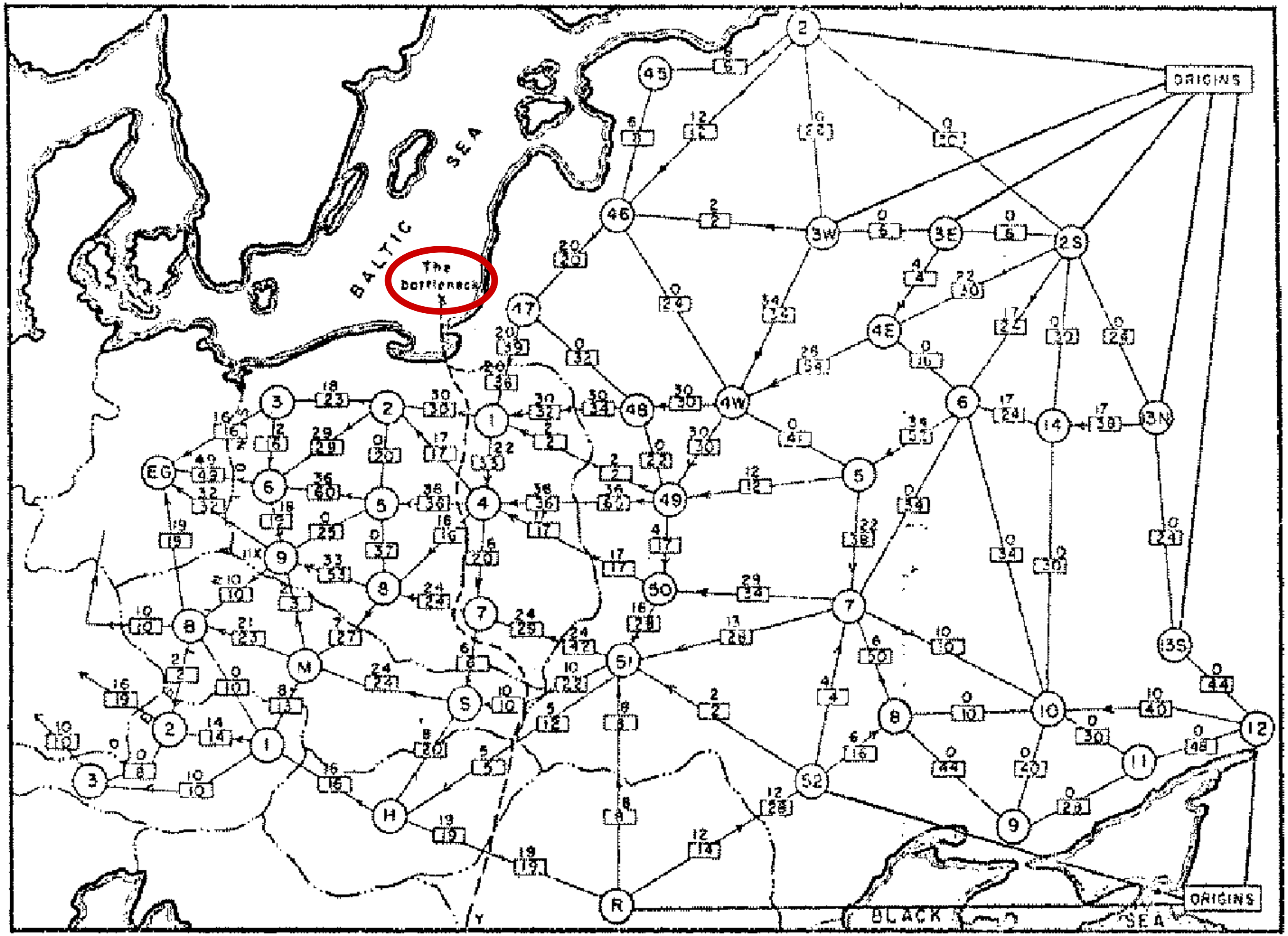
7.1 Network Flow

Soviet Rail Network, 1955



Reference: *On the history of the transportation and maximum flow problems.*
Alexander Schrijver in Math Programming, 91: 3, 2002. (See "External Links")

Maximum Flow and Minimum Cut

Max flow and min cut.

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

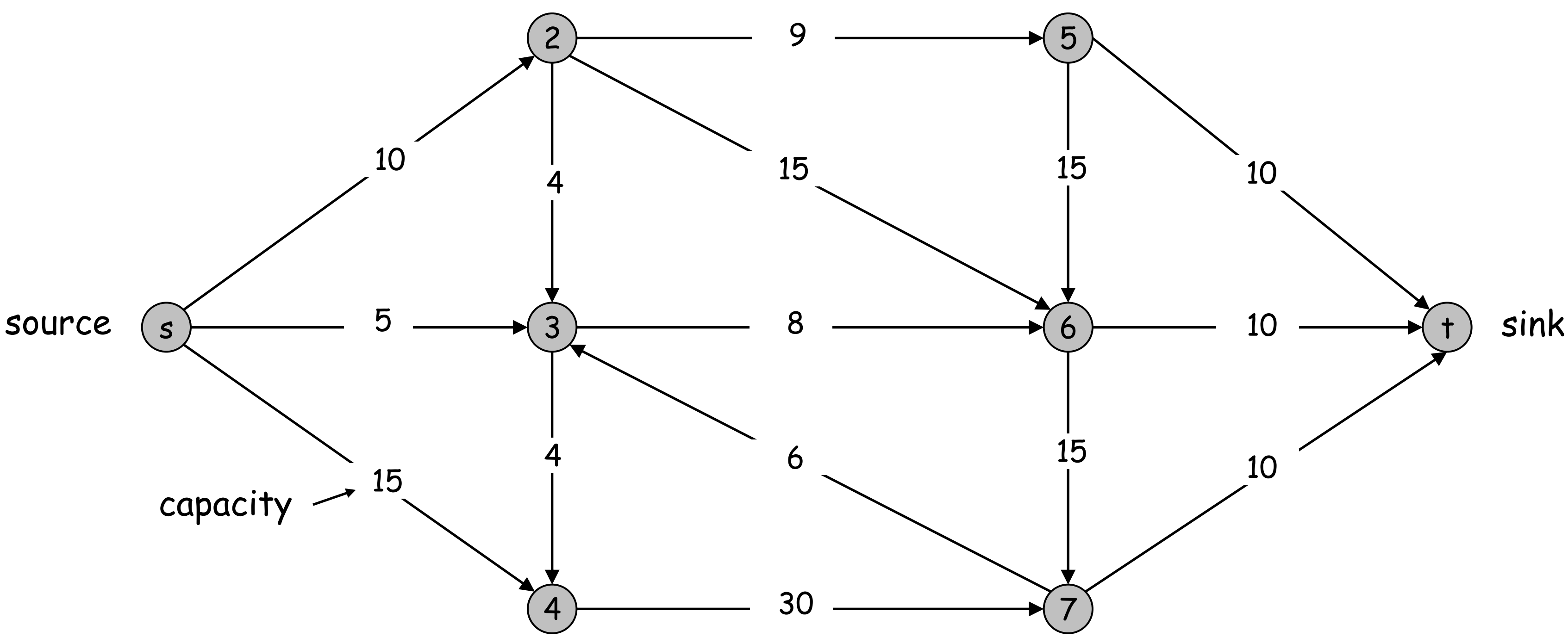
Nontrivial applications / reductions.

- Data mining.
- Open-pit mining.
- Project selection.
- Airline scheduling.
- Bipartite matching.
- Baseball elimination.
- Image segmentation.
- Network connectivity.
- Network reliability.
- Distributed computing.
- Egalitarian stable matching.
- Security of statistical data.
- Network intrusion detection.
- Multi-camera scene reconstruction.
- Many many more . . .

Minimum Cut Problem

Flow network.

- Abstraction for material flowing through the edges.
- G = (V, E) = directed graph
- Two distinguished nodes: s = source, t = sink.
- c(e) = capacity of edge e.

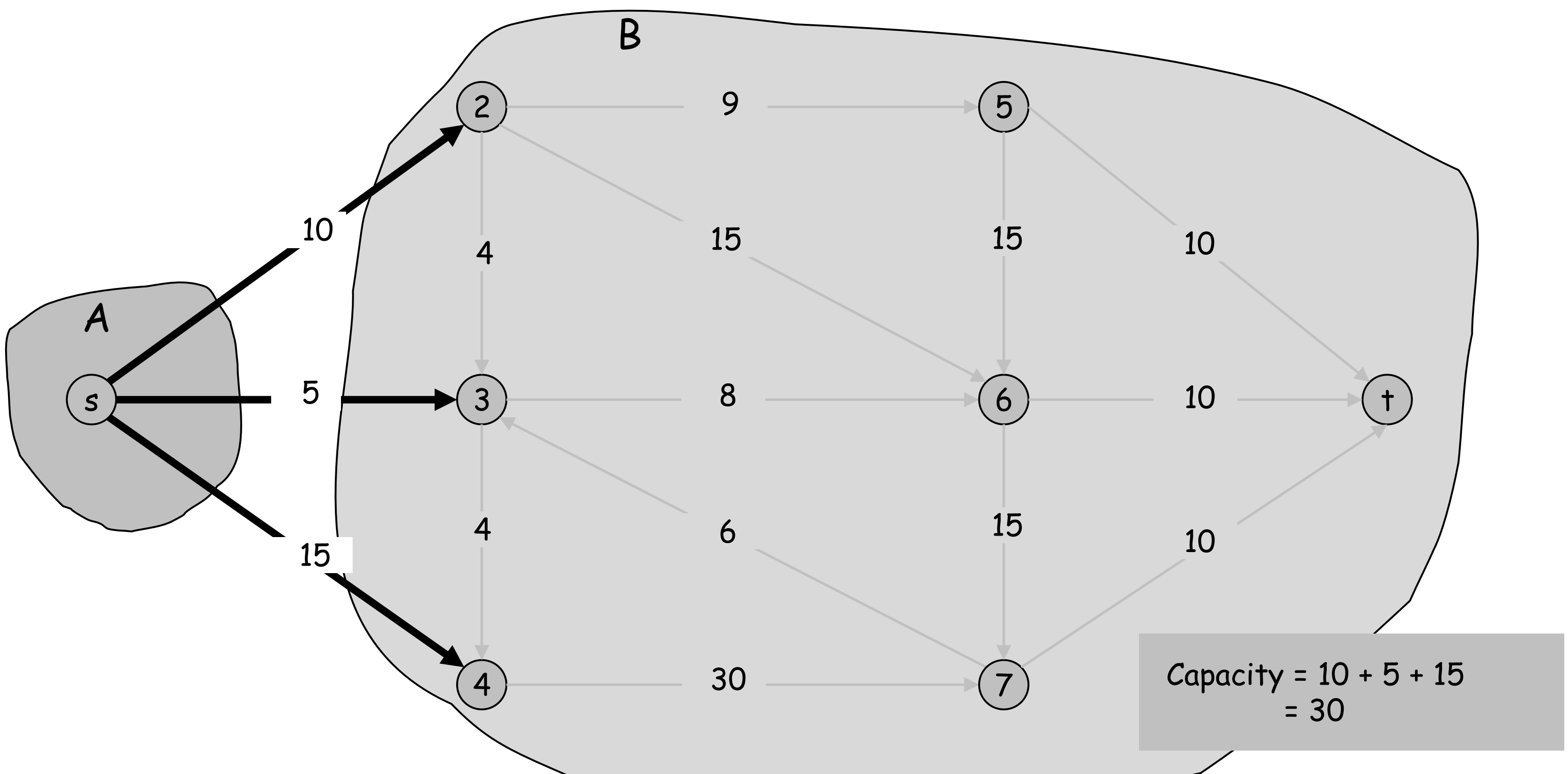


Cuts

Def. An **s-t cut** is a partition (A, B) of V with $s \in A$ and $t \in B$.

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

Q. What is the capacity of the s-t cut ({s}, V-{s}) ?

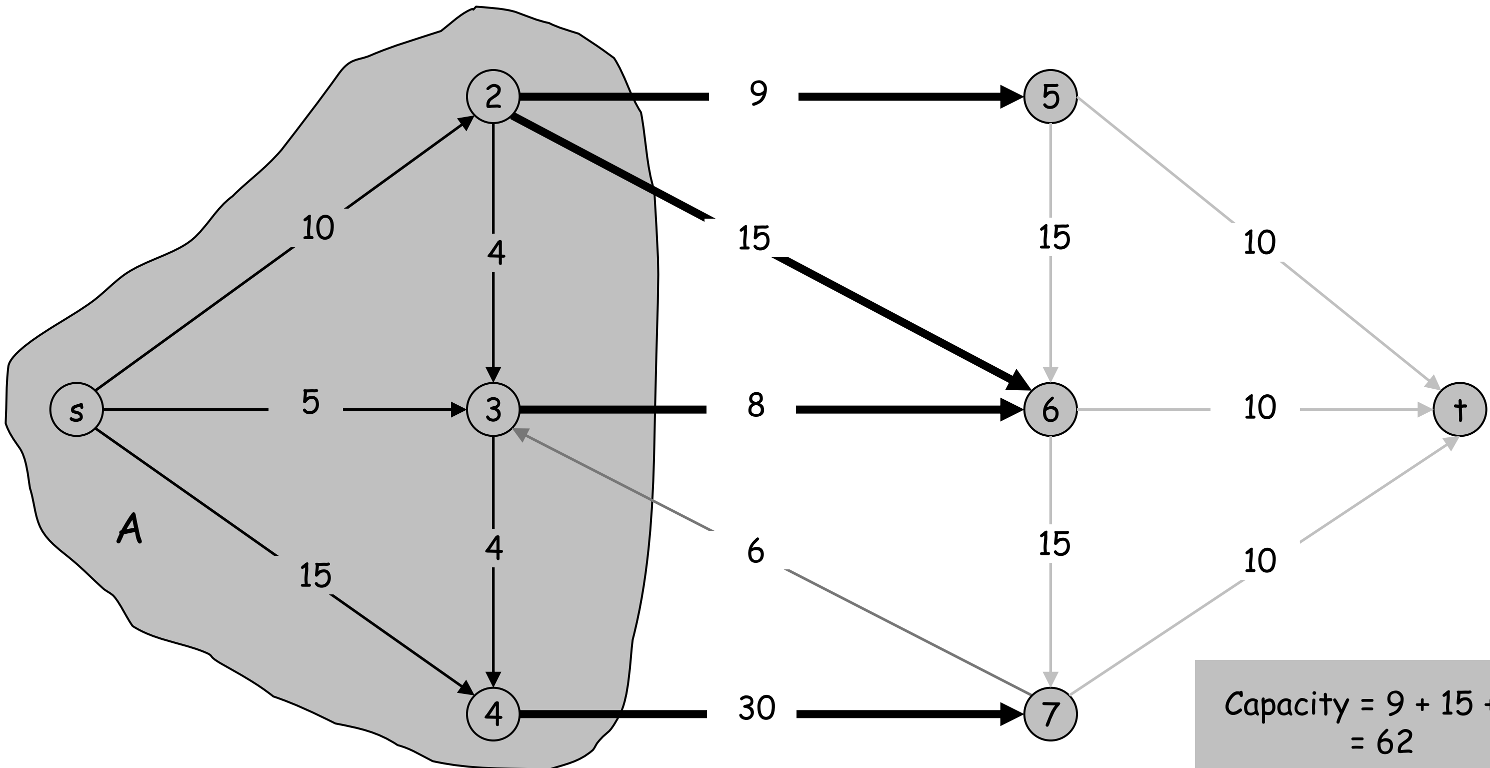


Cuts

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

Q. What is the capacity of the s-t cut ({s,2,3,4}, {5,6,7,t}) ?

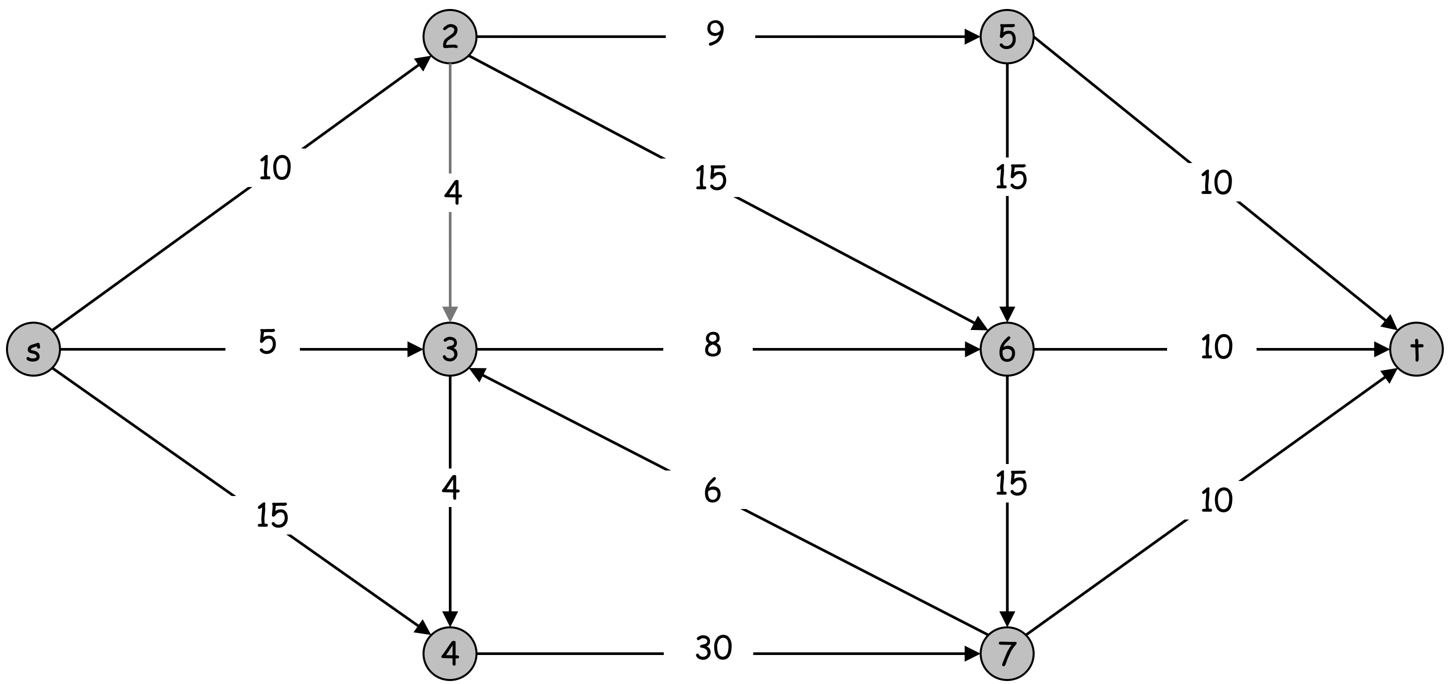


Capacity = 9 + 15 + 8 + 30
= 62

Minimum Cut Problem

Min s-t cut problem. Find an s-t cut of minimum capacity. (a bottleneck)

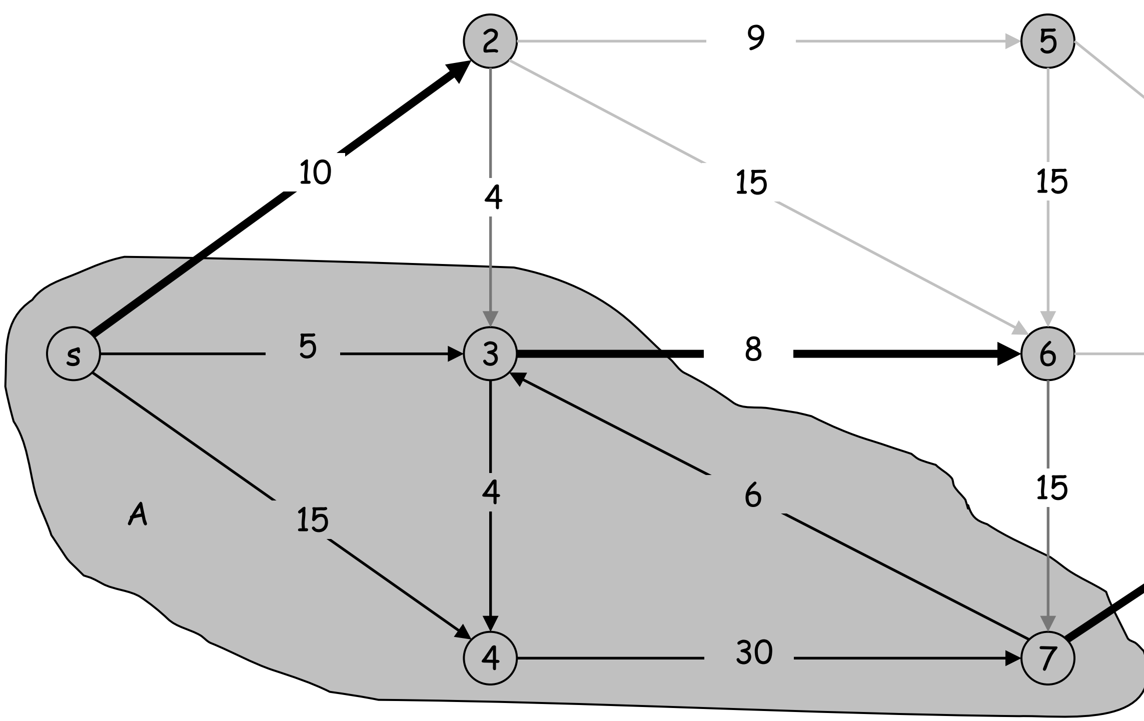
Q. What is the capacity of the minimum s-t cut? (1 min)



Minimum Cut Problem

Min s-t cut problem. Find an s-t cut of minimum capacity. (a bottleneck)

Q. What is the capacity of the minimum s-t cut? (1 min)



Capacity = 10 + 8 + 10
= 28

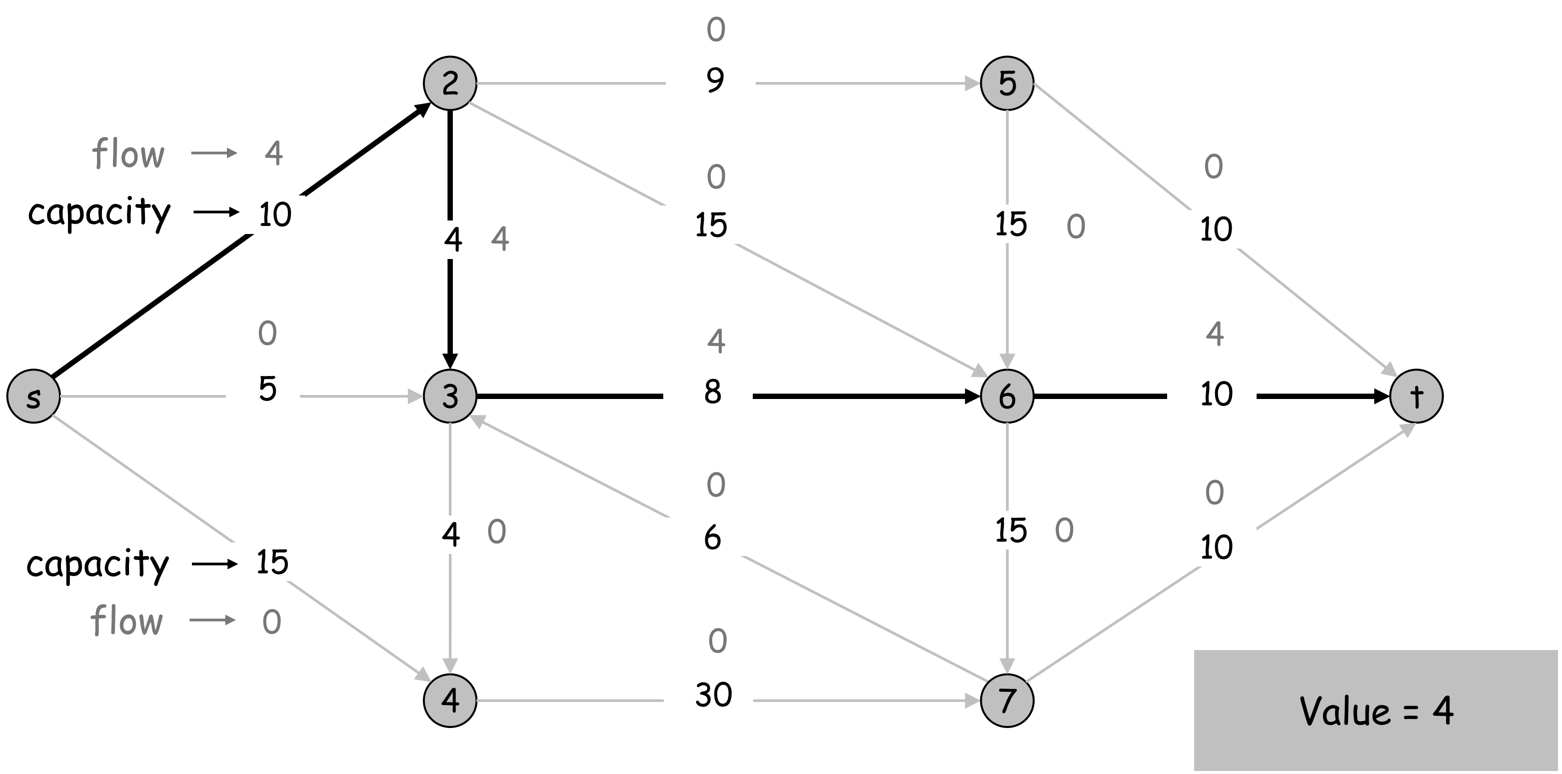
Flows

Def. An s-t flow is a function 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: $v(f) = \sum_{e \text{ out of } s} f(e)$.

Q. Is the flow below correct?



Flows

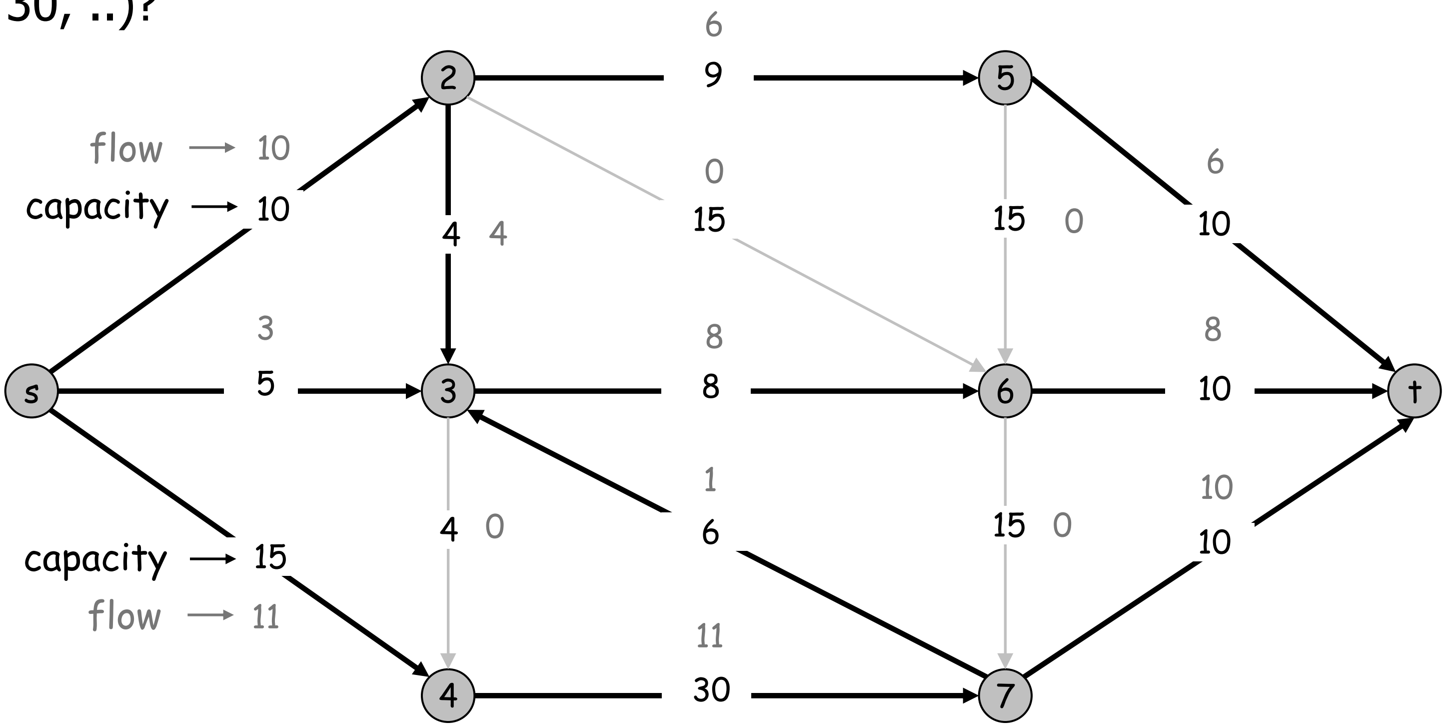
Def. An s-t flow is a function that satisfies:

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

Q. What is the value of this flow?

(24, 30, ..)?



Flows

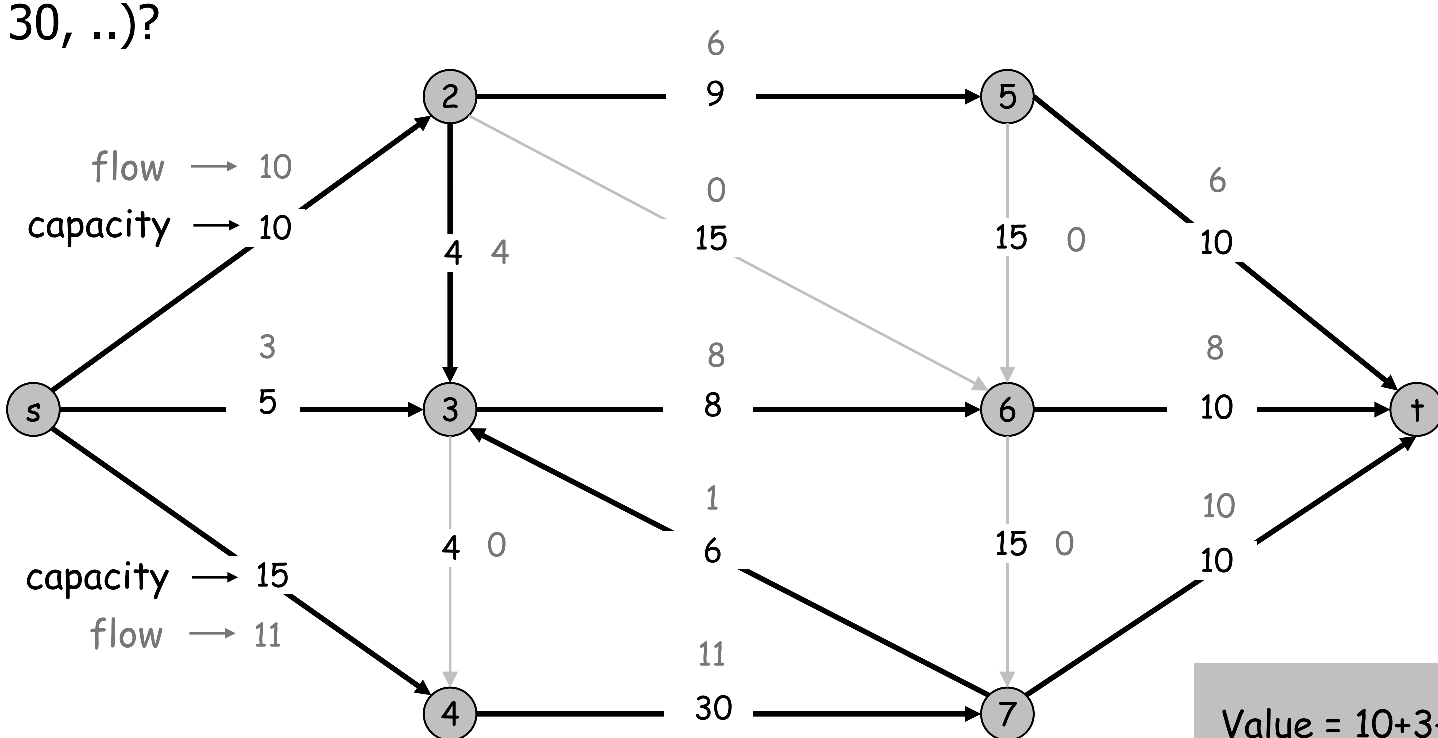
Def. An s-t flow is a function that satisfies:

- For each $e \in E$: $0 \le f(e) \le c(e)$ (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: $v(f) = \sum_{e \text{ out of } s} f(e)$.

Q. What is the value of this flow?

(24, 30, ..)?

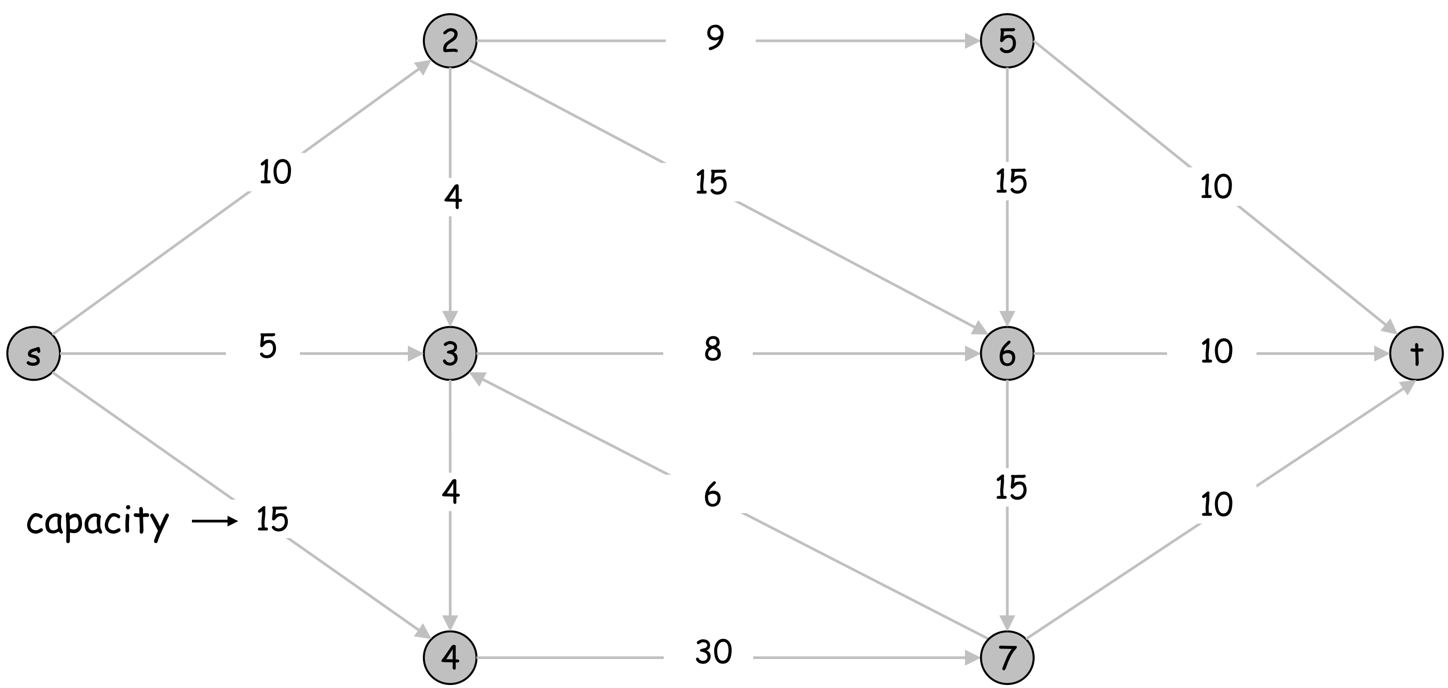


Value = 10+3+11 = 24

Maximum Flow Problem

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

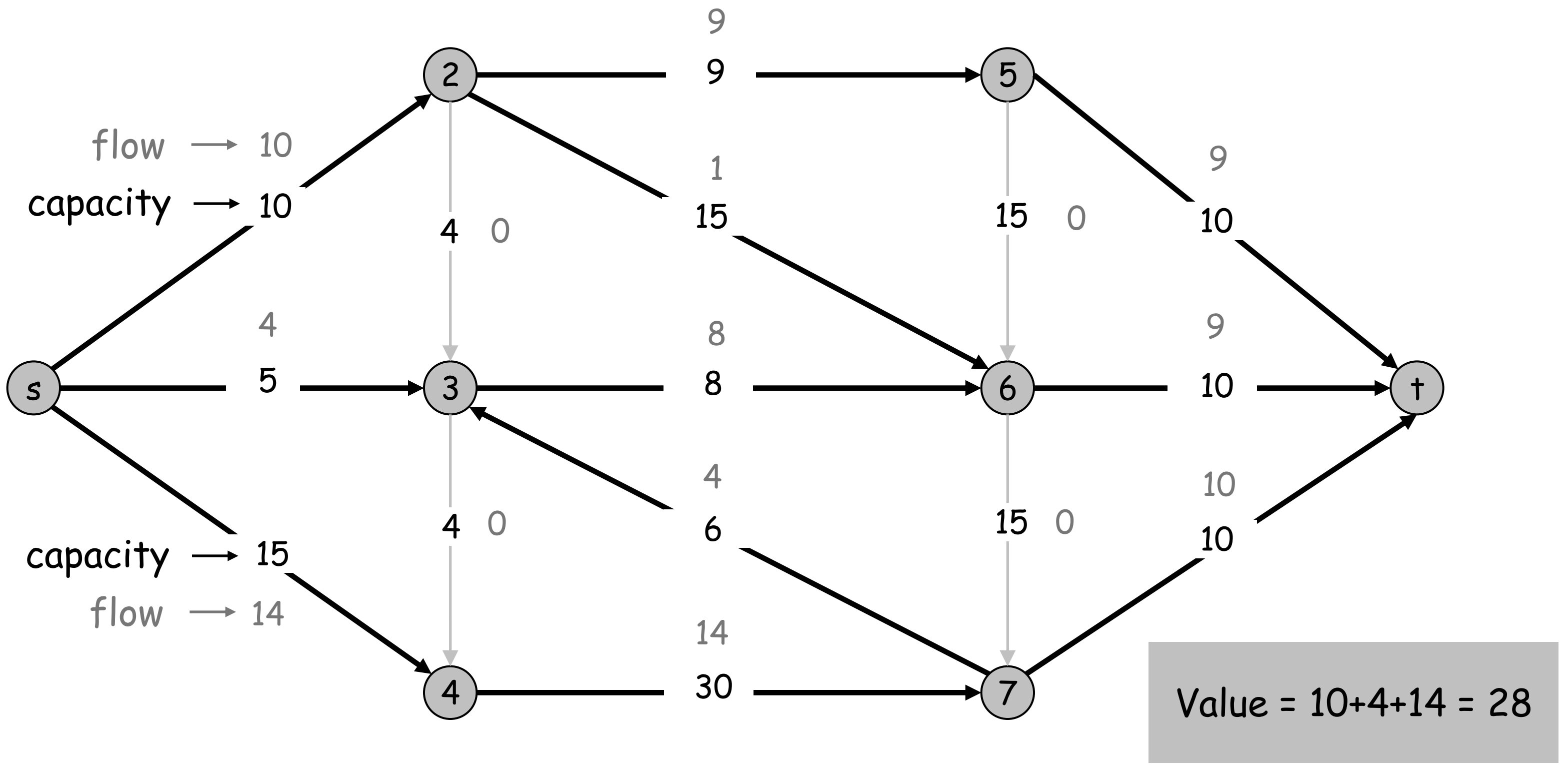
Q. What is the value of the maximum flow here? (1 min)



Maximum Flow Problem

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

Q. What is the value of the maximum flow here? (1 min)



Towards Solving the Maximum Flow Problem

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

Flow value lemma. The net flow across any cut is equal to flow leaving s.

Weak duality. For any s-t cut (A, B) we have $v(f) \leq cap(A, B)$.

Corollary. If $v(f) = cap(A, B)$, then f is a max flow.

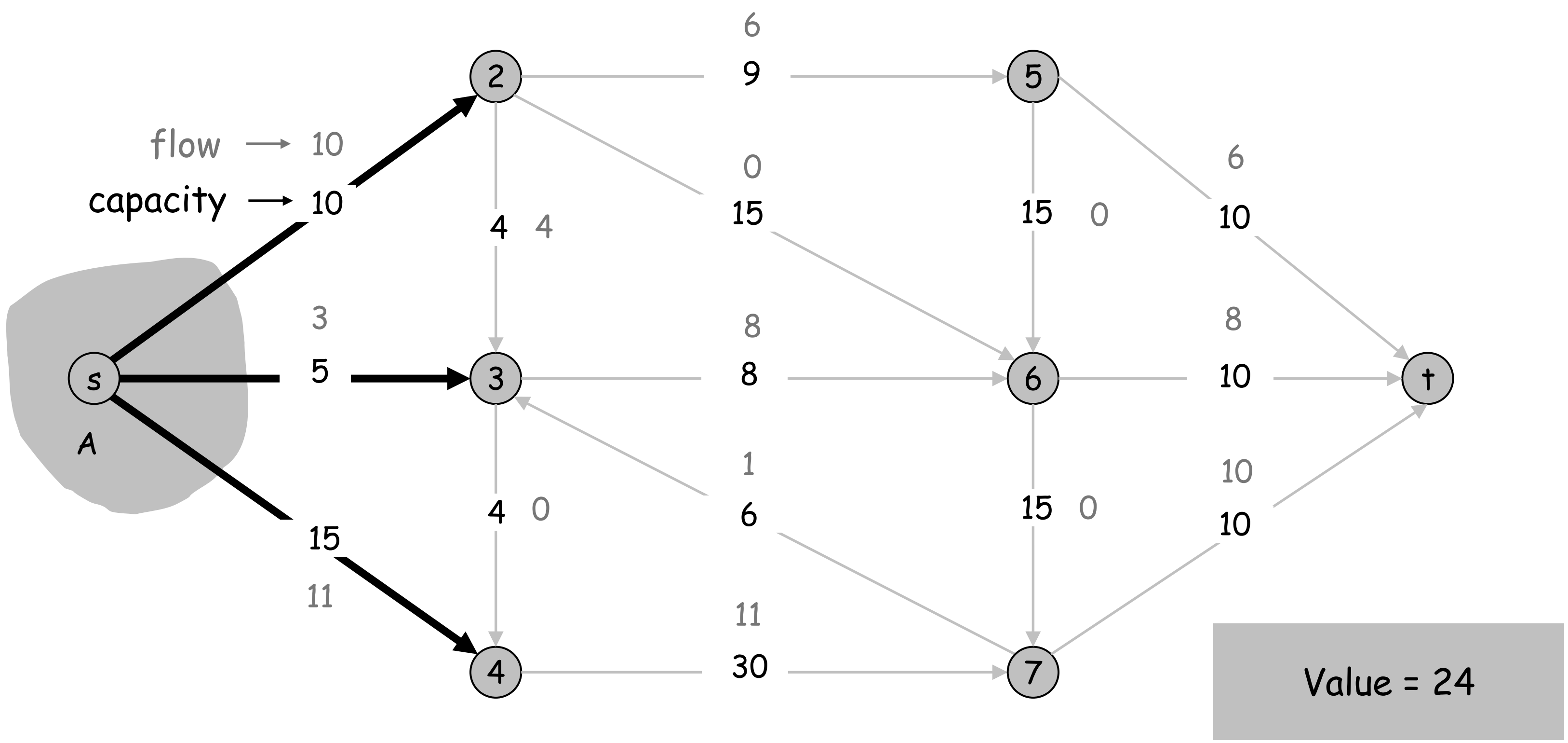
Max-flow algorithm

Max-flow min-cut theorem. [Ford-Fulkerson 1956]
The value of the max flow is equal to the capacity of the min cut.

Flows and Cuts

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

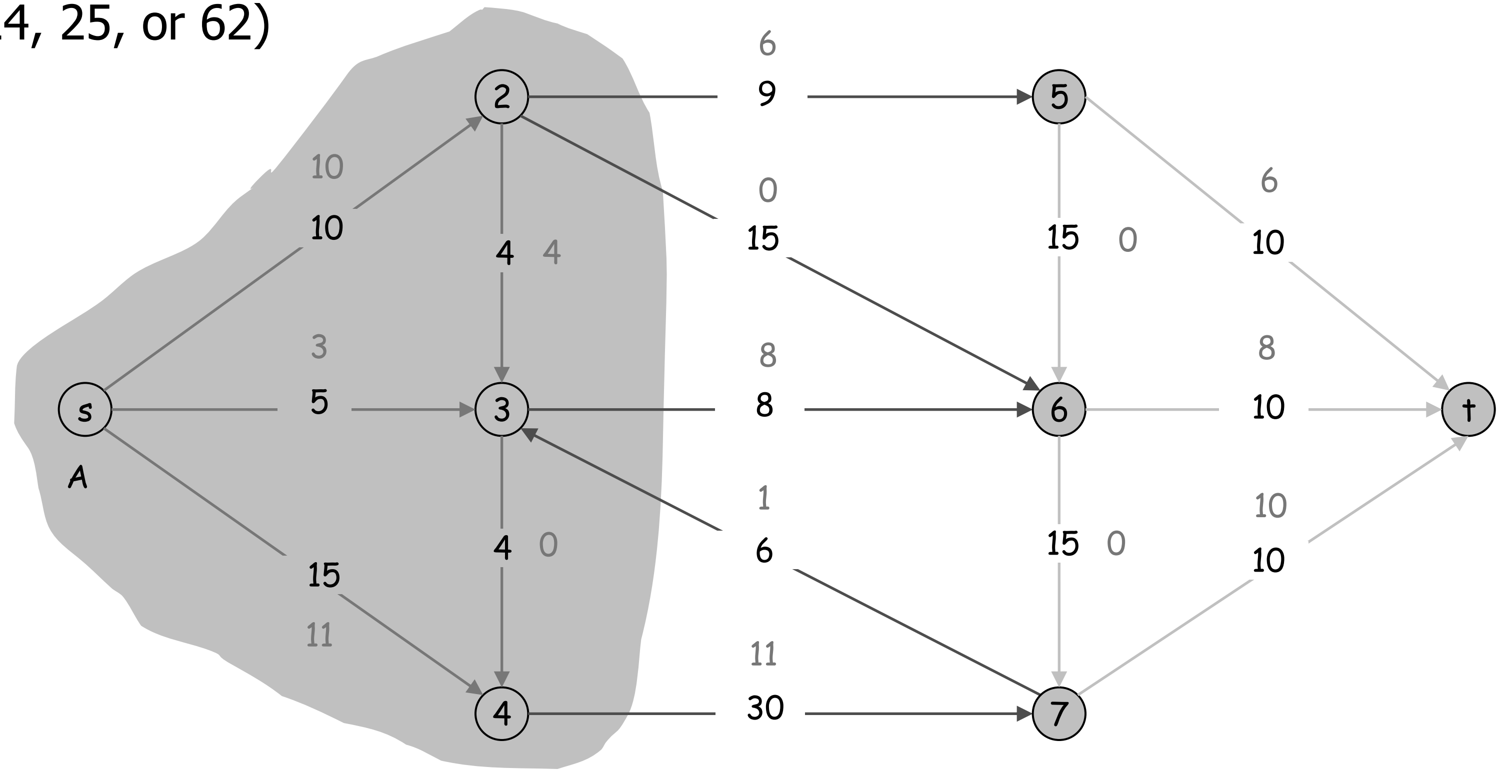
$$\sum_{e \text{ out of } A} f(e) - \sum_{e \text{ into } A} f(e) = v(f)$$



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

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

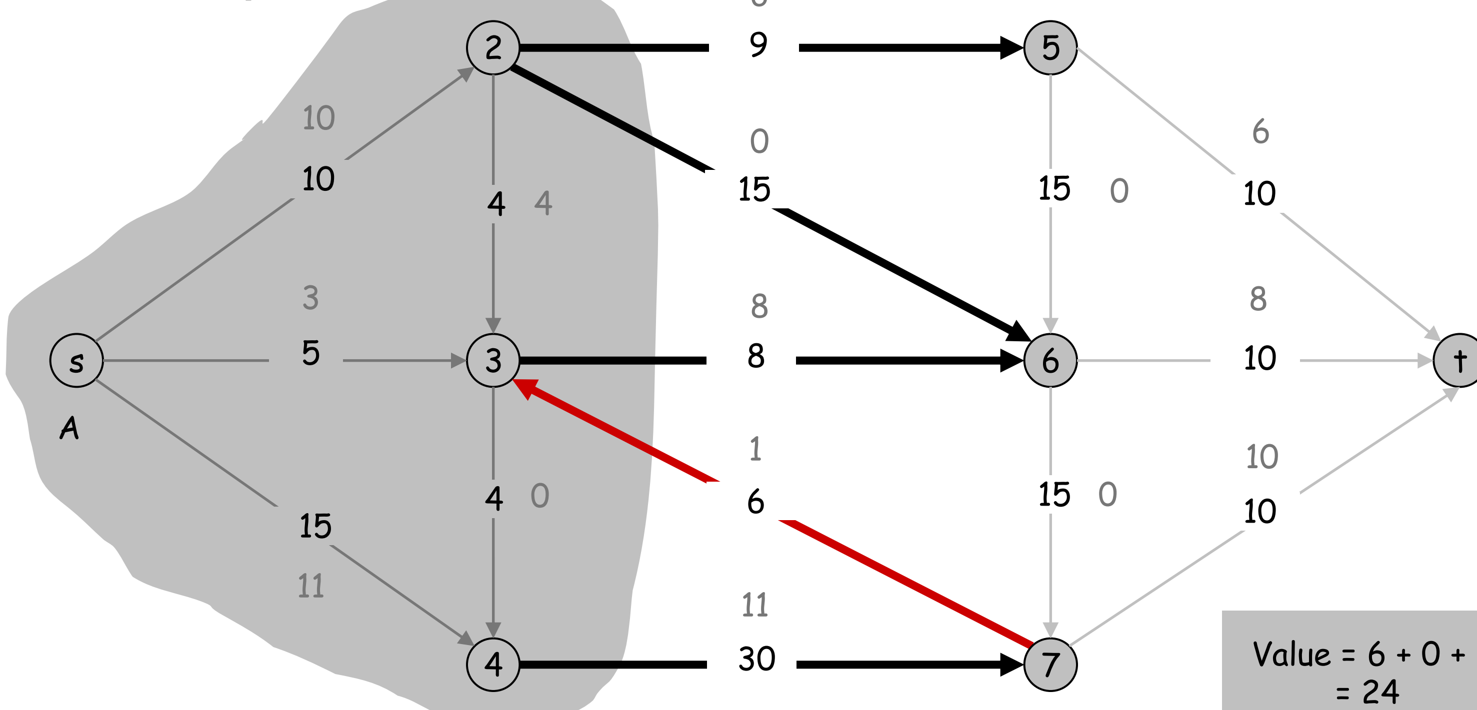
Q. What is the net flow sent across the cut ({s,2,3,4}, {5,6,7,t}) ?
(24, 25, or 62)



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

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

Q. What is the net flow sent across the cut ({s,2,3,4}, {5,6,7,t}) ? (24, 25, or 62)



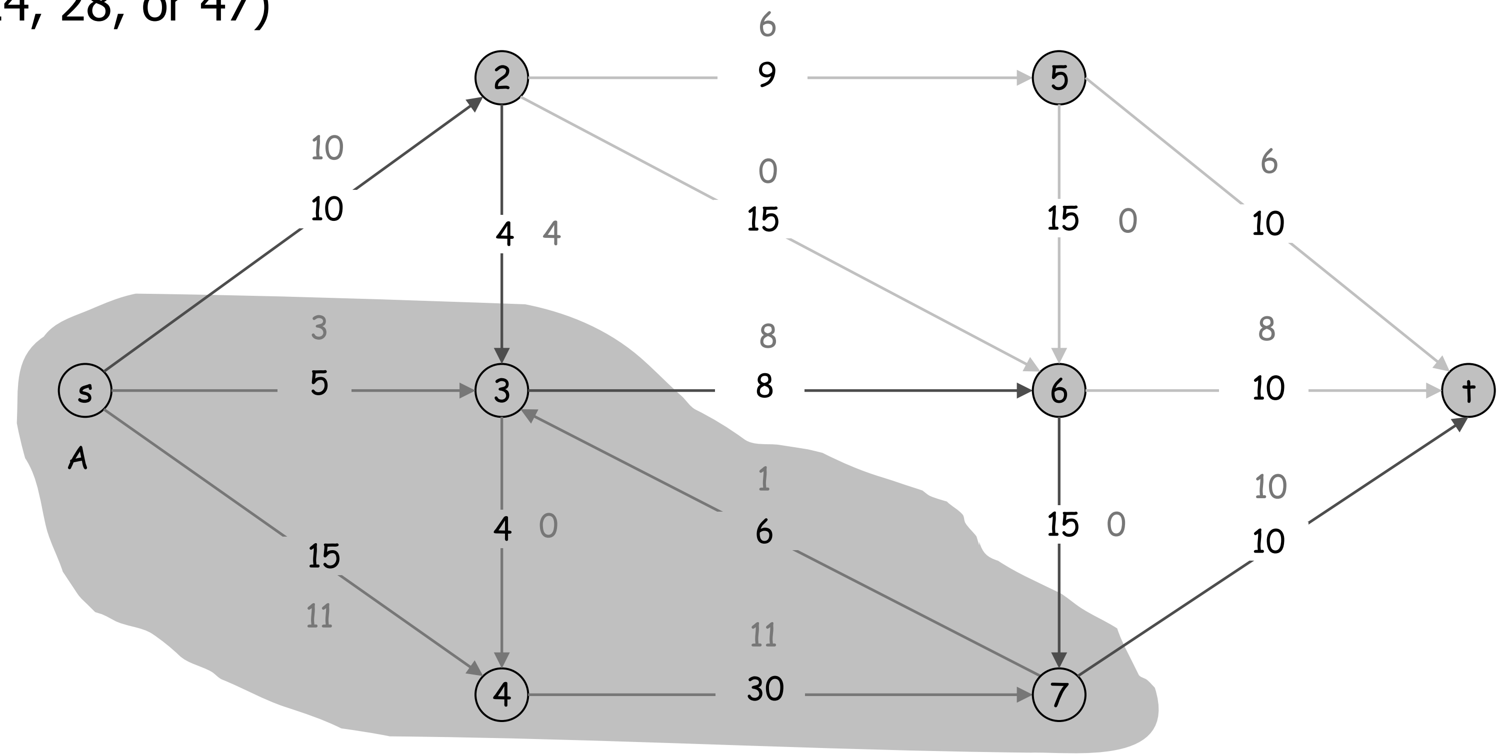
Value = 6 + 0 + 8 - 1 + 11
= 24

Flows and Cuts

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

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

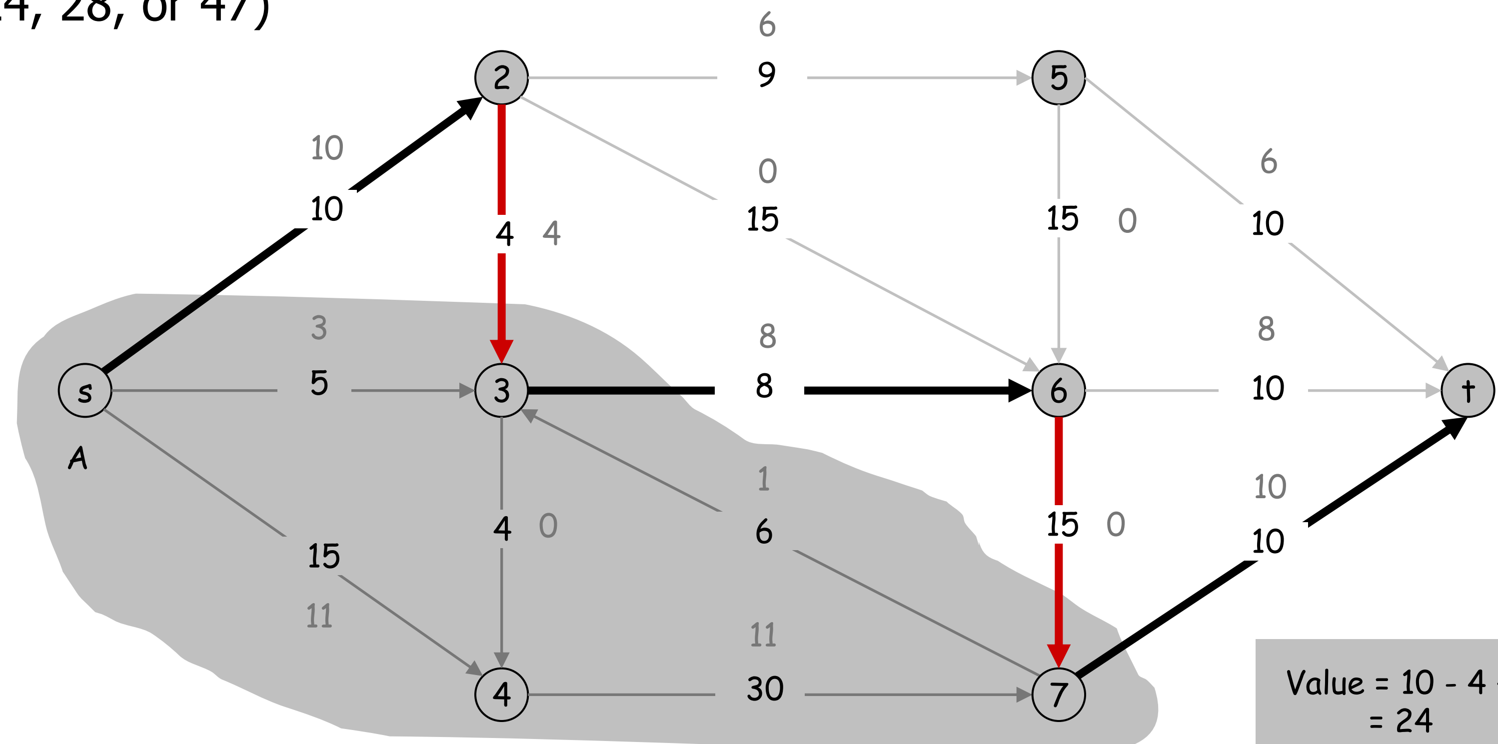
Q. What is the net flow sent across the cut ({s,3,4,7}, {2,5,6,t}) ? (24, 28, or 47)



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

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

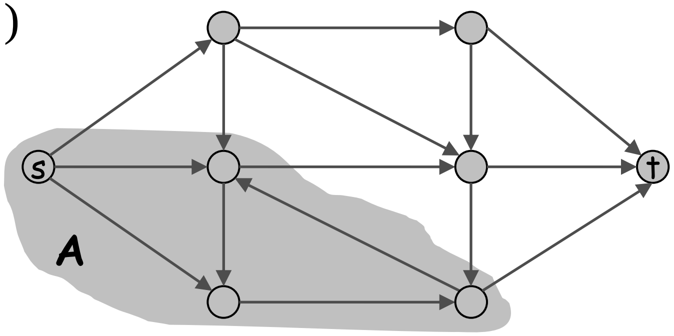
Q. What is the net flow sent across the cut ({s,3,4,7}, {2,5,6,t}) ? (24, 28, or 47)



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

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

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

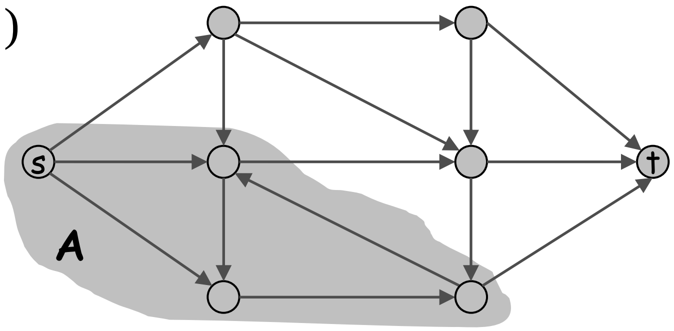


Pf.

Q. How to start?

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

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

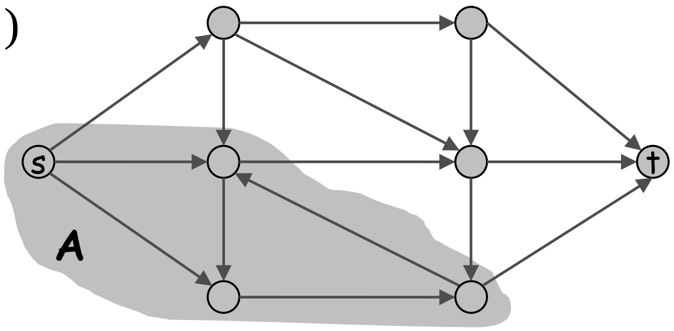


Pf.

$$\begin{aligned} v(f) &= \\ &= \\ &\vdots \\ &= \sum_{e \text{ out of } A} f(e) - \sum_{e \text{ into } A} f(e). \end{aligned}$$

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

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



Pf.

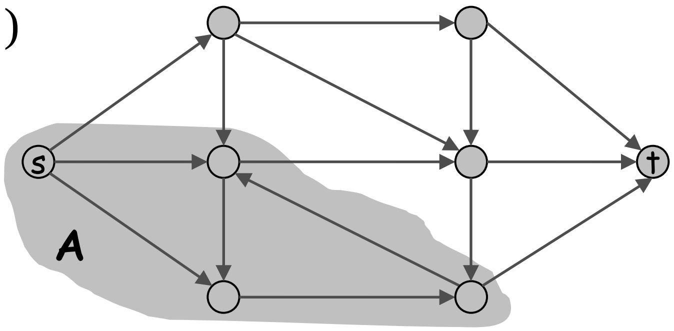
$$\begin{aligned} v(f) &= \sum_{e \text{ out of } s} f(e) \\ &= \\ &\vdots \\ &= \sum_{e \text{ out of } A} f(e) - \sum_{e \text{ in to } A} f(e). \end{aligned}$$

Q. What do we know for nodes v ≠ s in A on: $\sum_{e \text{ out of } v} f(e) - \sum_{e \text{ in to } v} f(e)$?

Flows and Cuts

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

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



Pf.

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

$$=$$

$$\vdots$$

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

Q. What do we know for nodes v ≠ s in A on: $\sum_{e \text{ out of } v} f(e) - \sum_{e \text{ in to } v} f(e)$?

A. Conservation of flow for v ≠ s or t: $\sum_{e \text{ in to } v} f(e) = \sum_{e \text{ out of } v} f(e)$

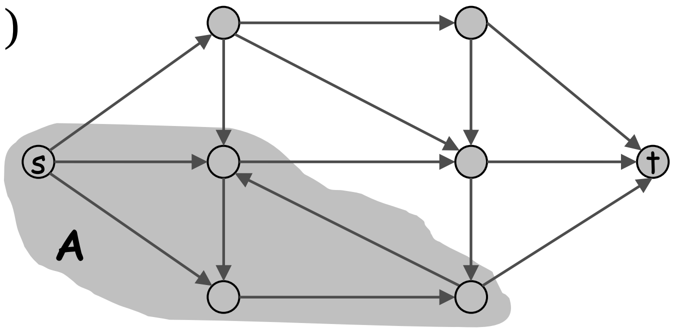
(from definition of flow)

TU Delft

Flows and Cuts

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

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



Pf.

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

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

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

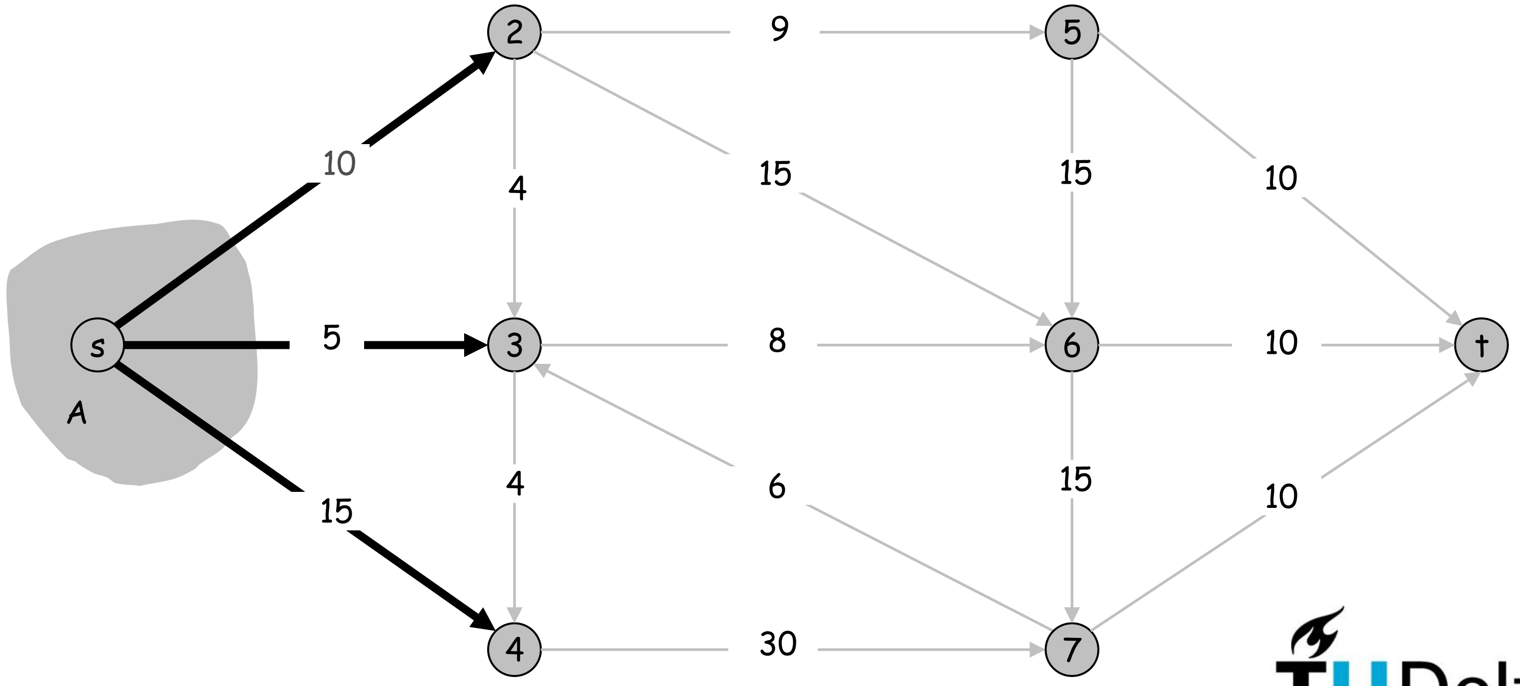
$$= \sum_{e \text{ out of } A} f(e) - \sum_{e \text{ in to } A} f(e).$$

Conservation for v ≠ s, t: $\sum_{e \text{ in to } v} f(e) = \sum_{e \text{ out of } v} f(e)$

This proof can be found on page 347.

Flows and Cuts

Q. Let f be any flow, and let (A, B) be any s-t cut. Can the value of the flow be more than the capacity of the cut?

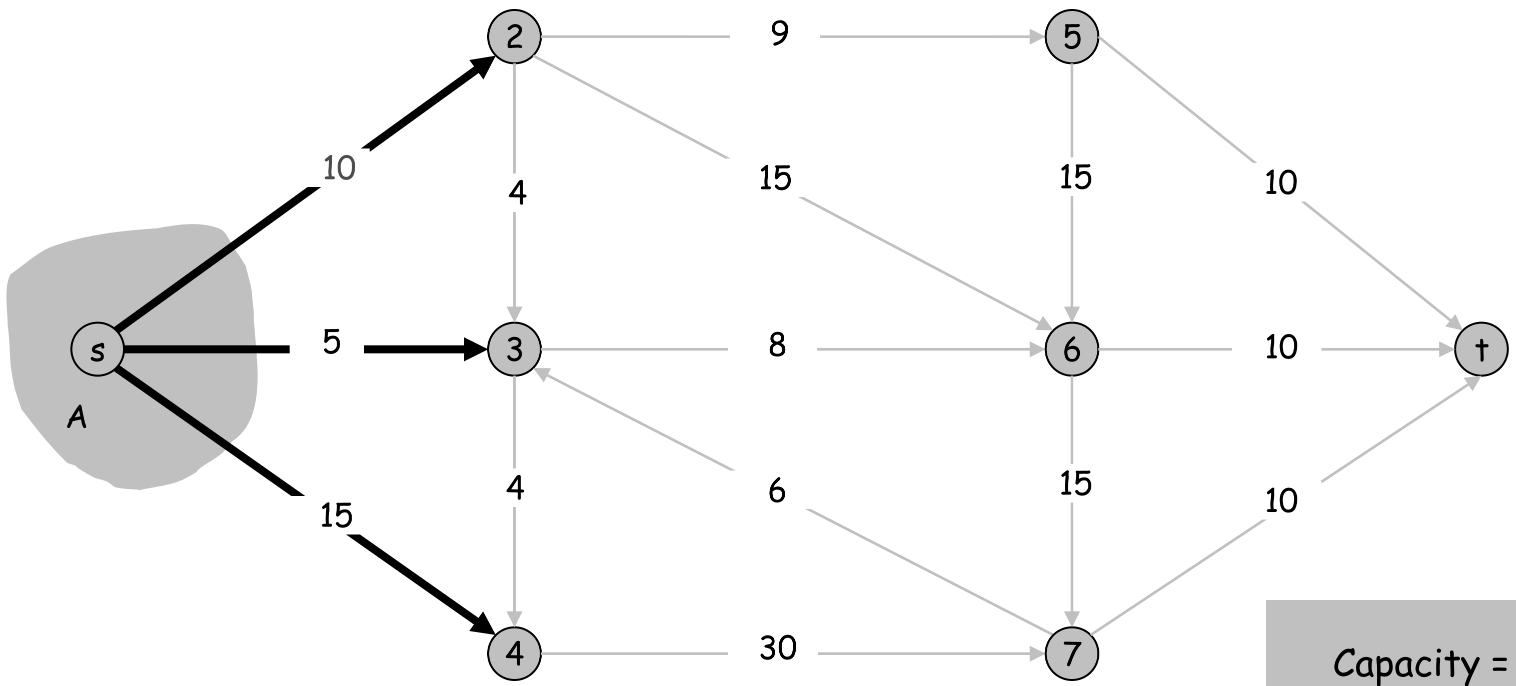


Flows and Cuts

Q. Let f be any flow, and let (A, B) be any s-t cut. Can the value of the flow be more than the capacity of the cut?

A. No. Proof on next slides.

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

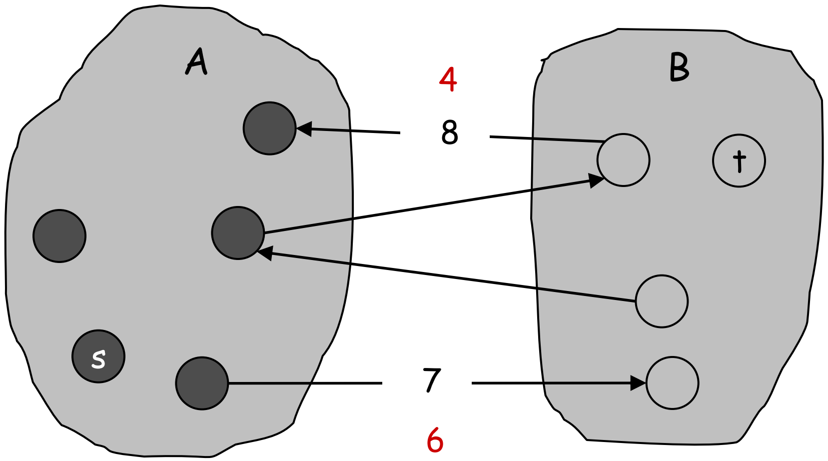


Capacity = 30

Weak duality. Let f be any flow. Then, for any s-t cut (A, B) we have $v(f) \leq cap(A, B)$.

Pf.

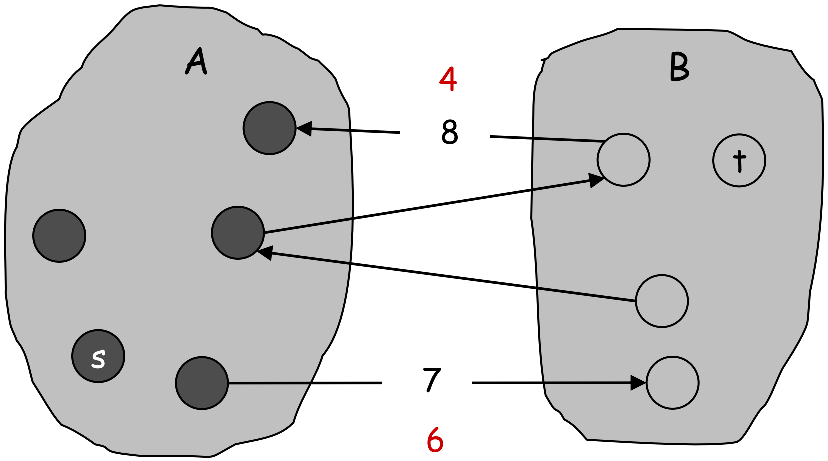
Q. How to start?



Weak duality. Let f be any flow. Then, for any s-t cut (A, B) we have v(f) $\leq$ cap(A, B).

Pf. Let a cut (A,B) be given.

$$\begin{aligned} v(f) &= \\ &\leq \quad M \\ &\leq \\ &= \quad \text{cap}(A, B) \end{aligned}$$

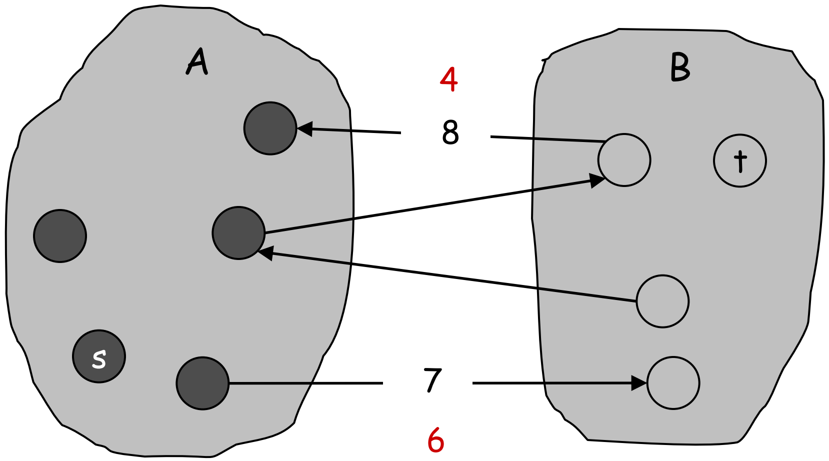


Q. Then what?

Weak duality. Let f be any flow. Then, for any s-t cut (A, B) we have v(f) $\leq$ cap(A, B).

Pf. Let a cut (A,B) be given.

$$\begin{aligned} v(f) &= \\ &\leq M \\ &\leq \sum_{e \text{ out of } A} c(e) \\ &= \text{cap}(A, B) \end{aligned}$$



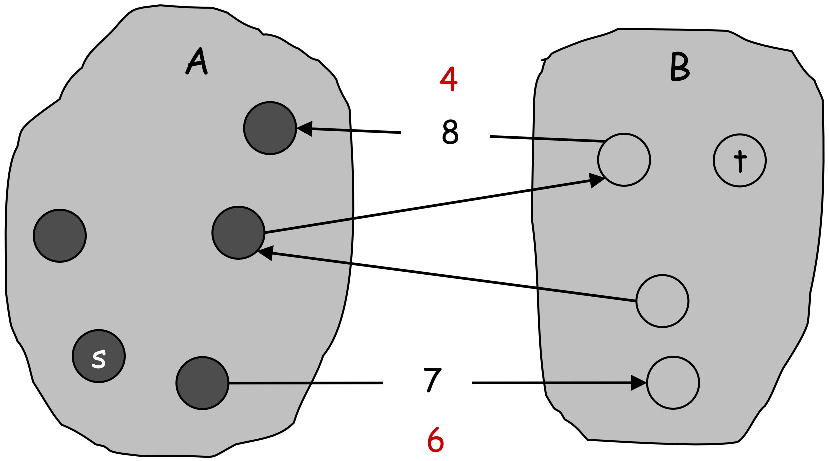
Q. Then what?

A. Use definition of capacity

Weak duality. Let f be any flow. Then, for any s-t cut (A, B) we have v(f) ≤ cap(A, B).

Pf. Let a cut (A,B) be given.

$$\begin{aligned} v(f) &= \sum_{e \text{ out of } A} f(e) - \sum_{e \text{ in to } A} f(e) \\ &\leq M \\ &\leq \sum_{e \text{ out of } A} c(e) \\ &= \text{cap}(A, B) \end{aligned}$$



Q. Then what?

A. Use definition of capacity

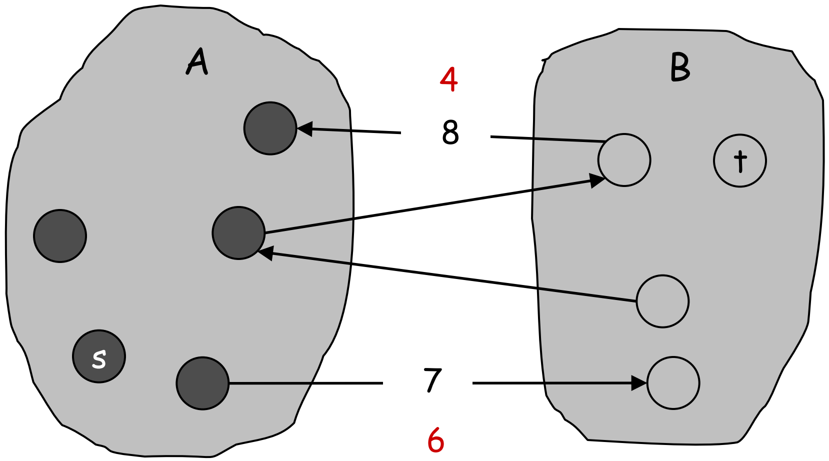
A. Use previous lemma (flow value lemma):

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

Weak duality. Let f be any flow. Then, for any s-t cut (A, B) we have $v(f) \leq cap(A, B)$.

Pf. Let a cut (A,B) be given.

$$
\begin{aligned}
v(f) &= \sum_{e \text{ out of } A} f(e) - \sum_{e \text{ in to } A} f(e) \\
&\leq M \\
&\leq \sum_{e \text{ out of } A} c(e) \\
&= \operatorname{cap}(A, B)
\end{aligned}
$$

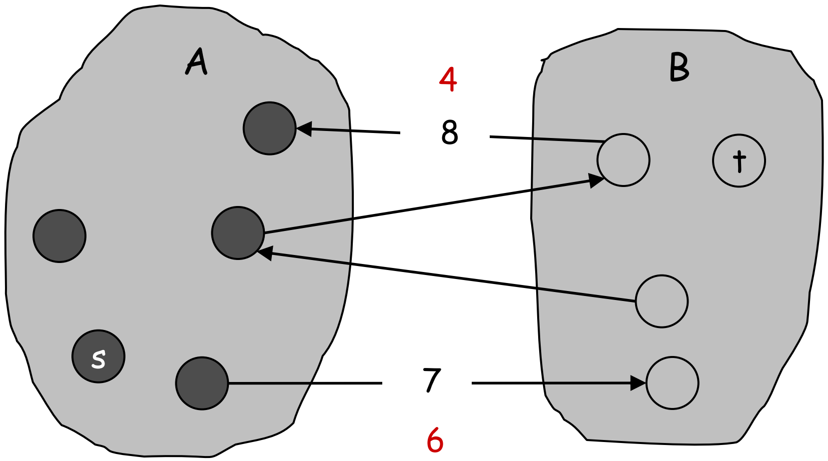


Q. Why should this hold?

Weak duality. Let f be any flow. Then, for any s-t cut (A, B) we have v(f) ≤ cap(A, B).

Pf. Let a cut (A,B) be given.

$$\begin{aligned} v(f) &= \sum_{e \text{ out of } A} f(e) - \sum_{e \text{ in to } A} f(e) \\ &\le \sum_{e \text{ out of } A} f(e) \\ &\le \sum_{e \text{ out of } A} c(e) \\ &= \text{cap}(A, B) \end{aligned}$$



Q. Why should this hold?

A. Use simple arithmetic: $\sum_{e \text{ out of } A} f(e) - \sum_{e \text{ in to } A} f(e) \le \sum_{e \text{ out of } A} f(e)$

A. Use definition (of flow): $0 \le f(e) \le c(e)$

This proof can be found on page 347-348.

TU Delft

Flows and Cuts

Weak duality. Let f be any flow. Then, for any s-t cut (A, B) we have $v(f) \le \text{cap}(A, B)$.

Pf. Let a cut (A,B) be given.

$$\begin{aligned} v(f) &= \sum_{e \text{ out of } A} f(e) - \sum_{e \text{ in to } A} f(e) \\ &\le \sum_{e \text{ out of } A} f(e) \\ &\le \sum_{e \text{ out of } A} c(e) \\ &= \text{cap}(A, B) \end{aligned}$$

(by flow value lemma)

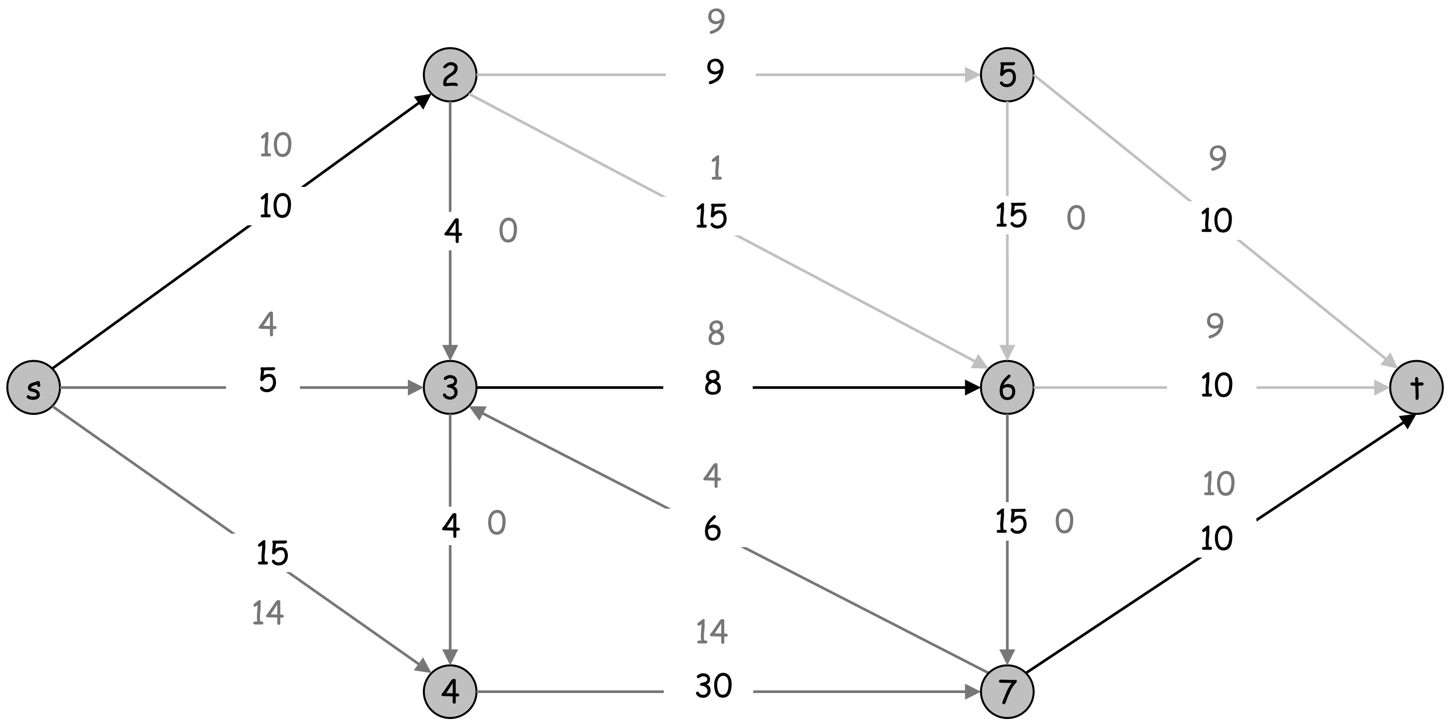
Use definition (of flow):
$0 \le f(e) \le c(e)$

(by definition of capacity)

This proof can be found on page 347-348.

Certificate of Optimality

Q. How can we check when is a flow maximal?

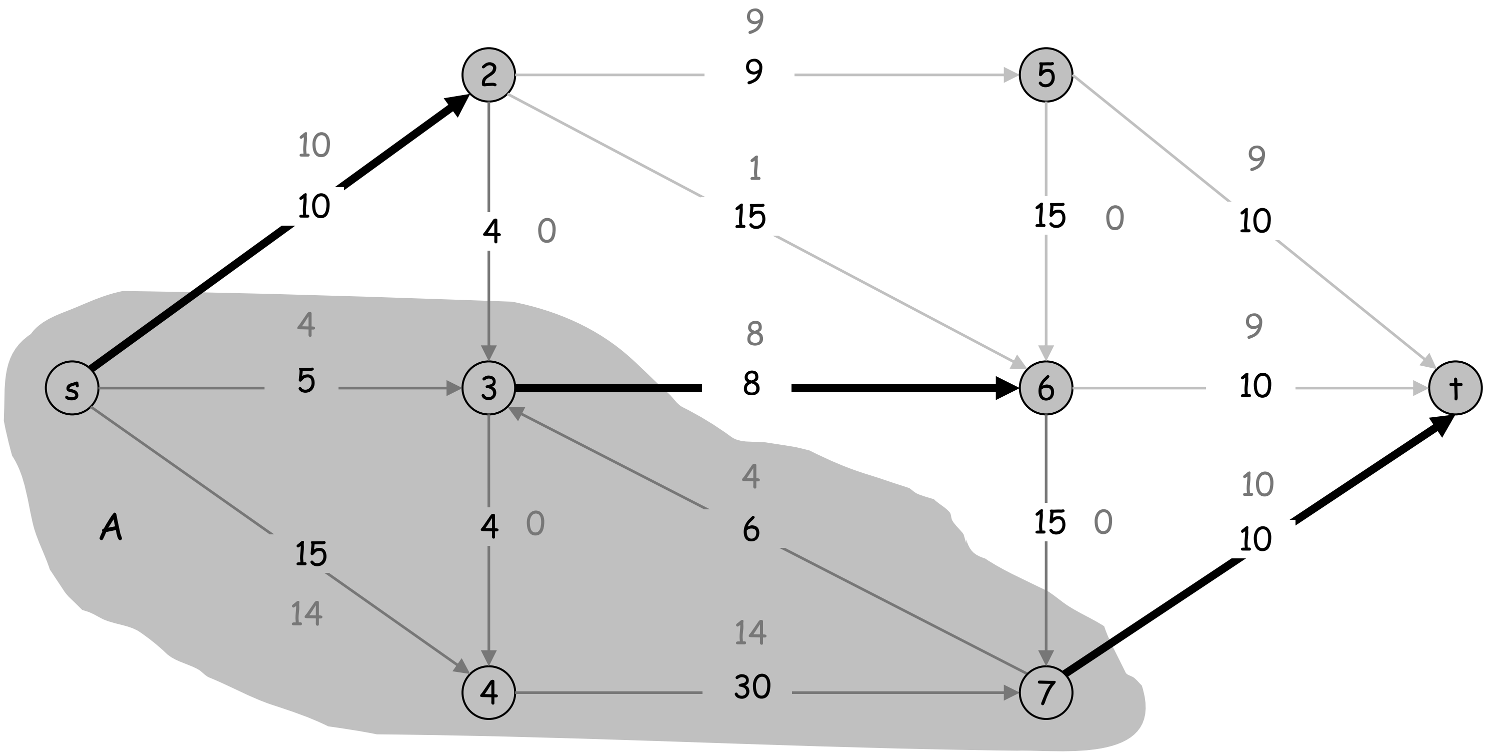


Certificate of Optimality

Q. How can we check when is a flow maximal?

A. If there is a cut (A,B) s.t. v(f) = cap(A, B), then f is a max flow.

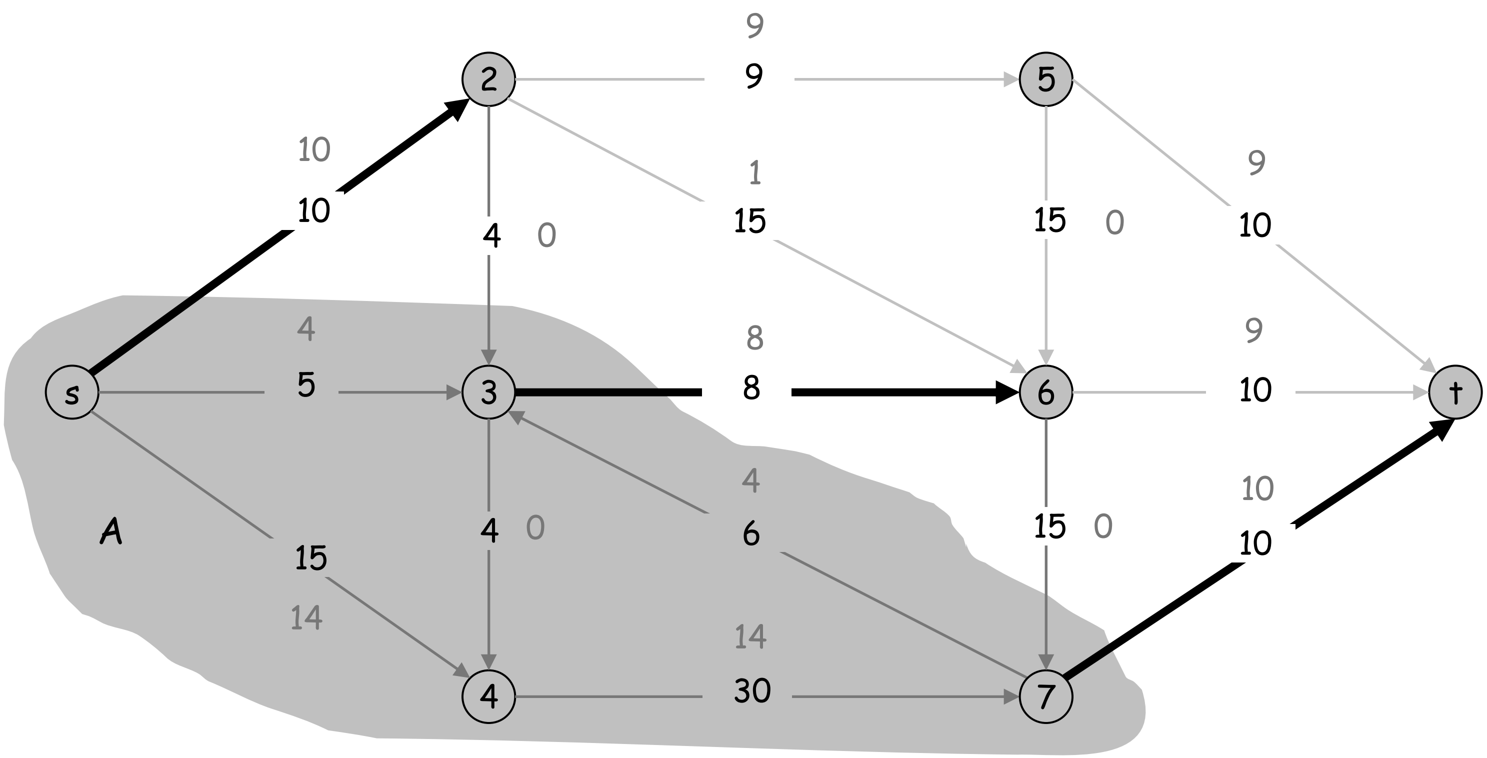
Value of flow = 10+4+14 = 28
Cut capacity = 10+8+10 = 28 $\Rightarrow$ Flow value $\leq$ 28



Certificate of Optimality

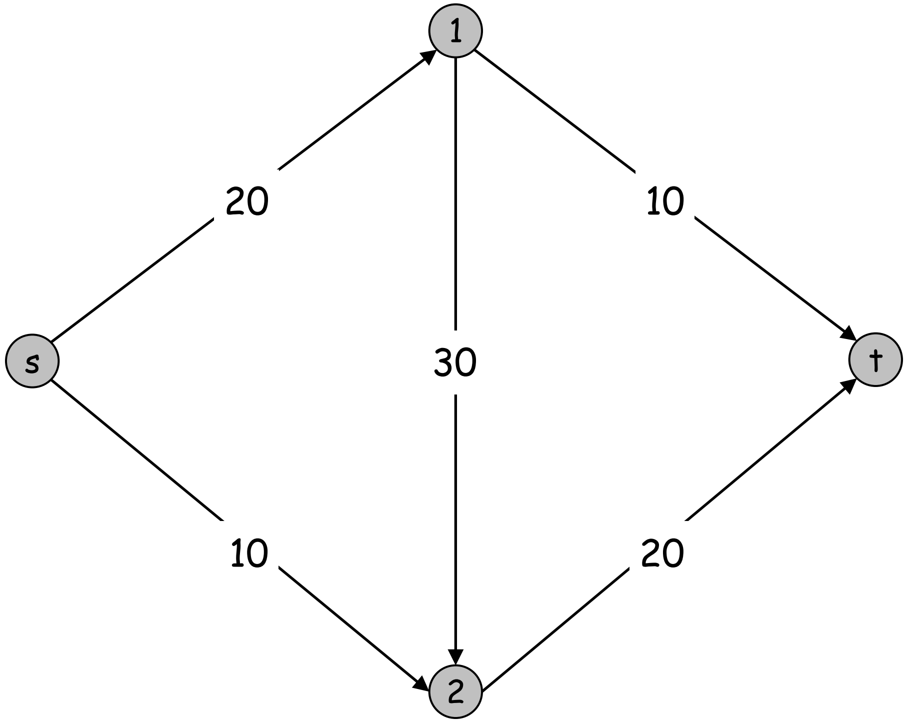
Corollary. Let f be any flow, and let (A, B) be any cut.
If v(f) = cap(A, B), then f is a max flow.

Value of flow = 10+4+14 = 28
Cut capacity = 10+8+10 = 28 ⇒ Flow value ≤ 28



Towards a Max Flow Algorithm

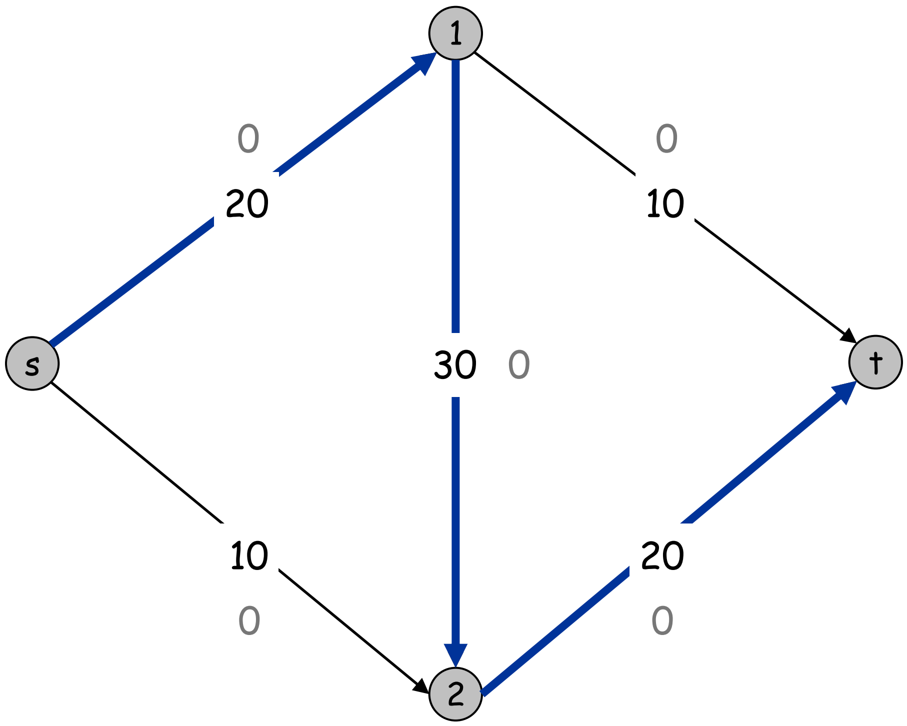
Q. How to find such a max flow? (1 min)



Towards a Max Flow Algorithm

Greedy algorithm.

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



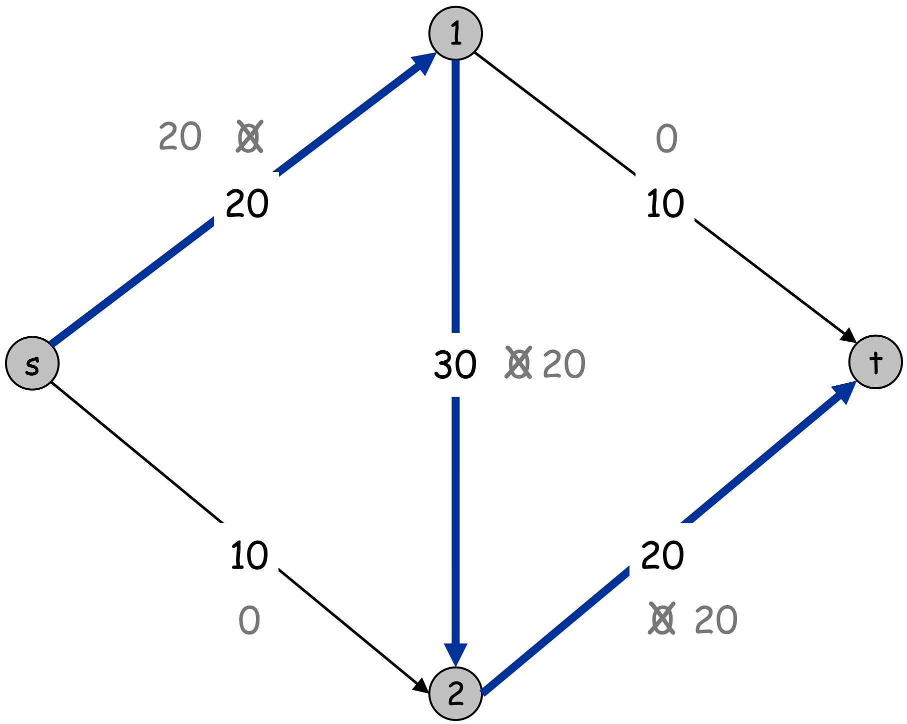
Flow value = 0

Towards a Max Flow Algorithm

Greedy algorithm.

- Start with f(e) = 0 for all edges e ∈ 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.

Q. Can the flow below be improved in this way (or are we stuck)?



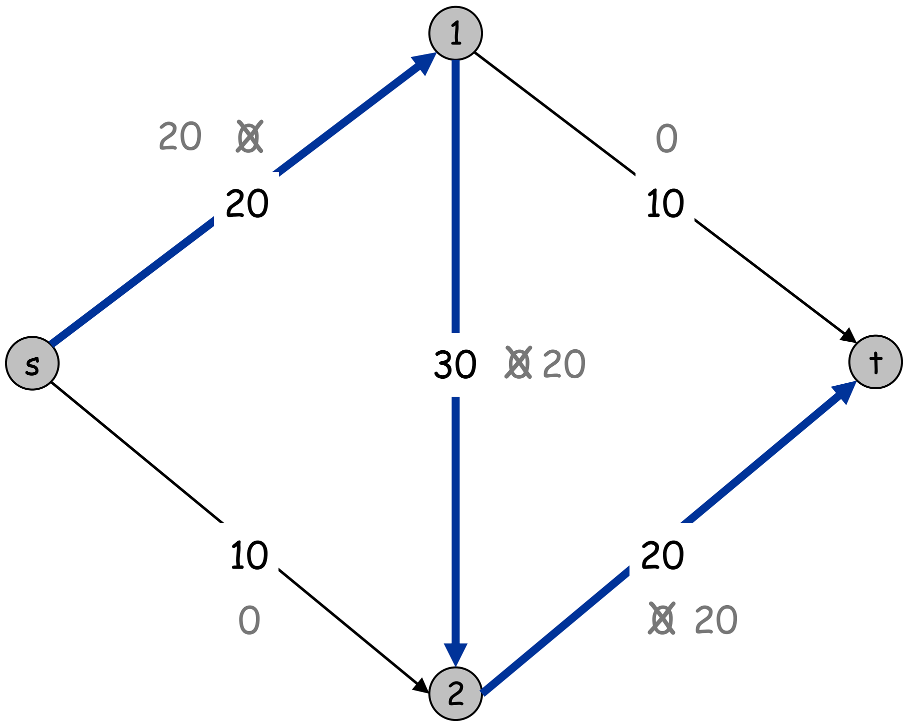
Flow value = 20

Towards a Max Flow Algorithm

Greedy algorithm.

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

Q. Is the flow below optimal?



Flow value = 20

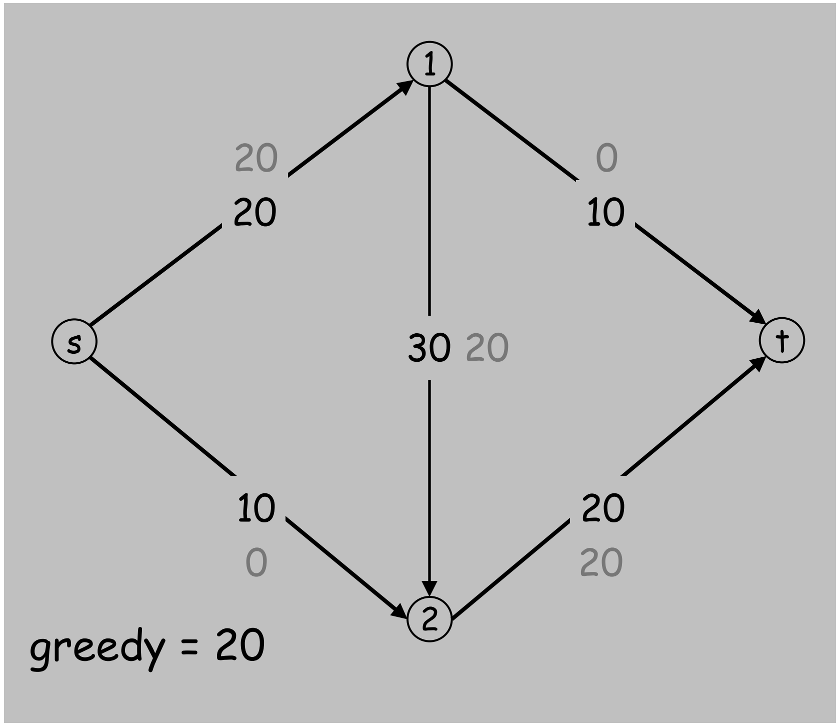
Towards a Max Flow Algorithm

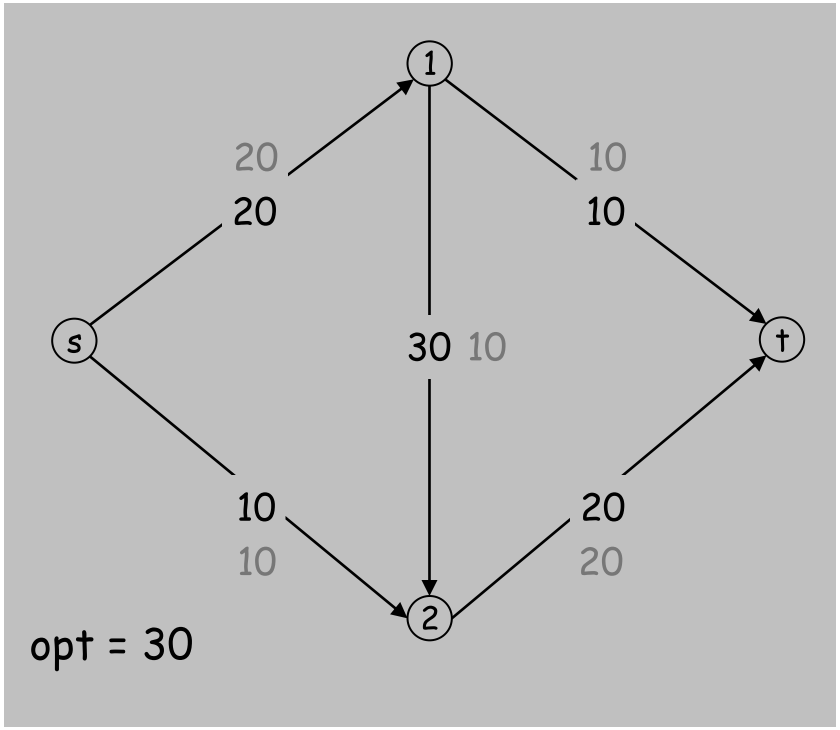
Greedy algorithm.

- Start with f(e) = 0 for all edges e ∈ 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.

Q. How can we fix this? (1 min)

locally optimality ⇏ global optimality

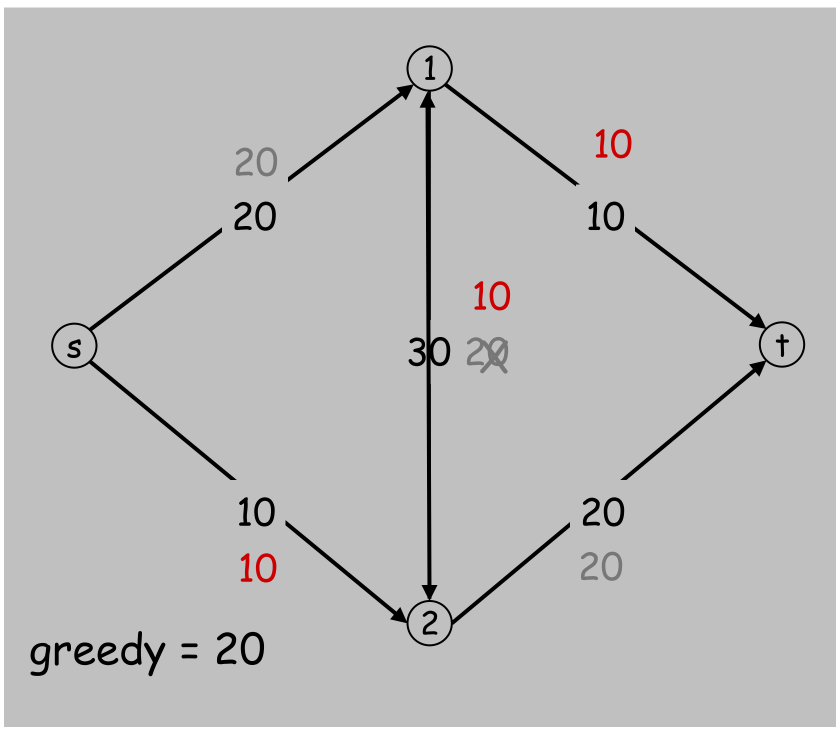


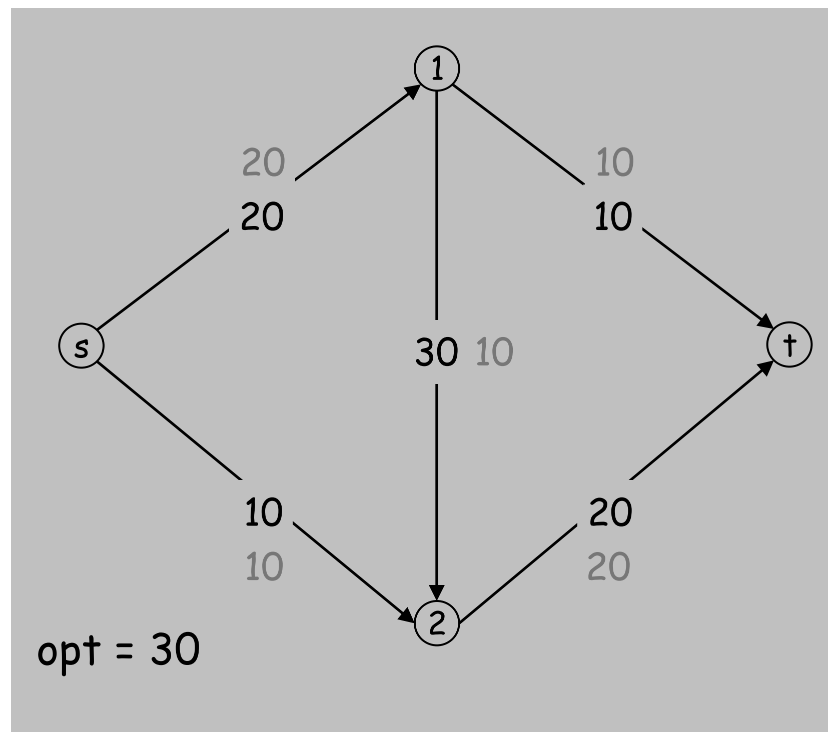


Towards a Max Flow Algorithm

Greedy algorithm.

- Start with $f(e) = 0$ for all edges $e \in E$.
- Find an s-t path P where each edge has $f(e) \leq c(e)$.
- Augment flow along path P.
- Repeat until you get stuck.
- Also allow decreasing the flow on an edge… ("undo")

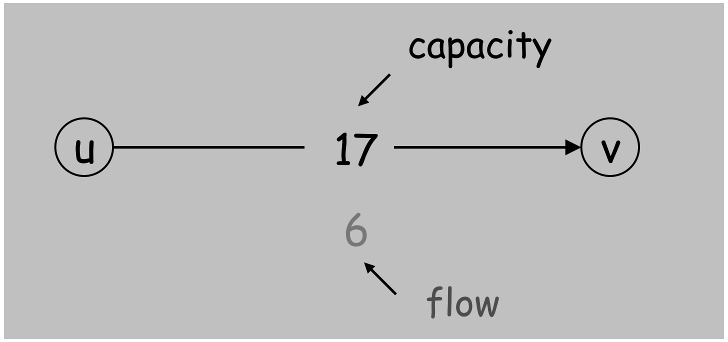




Residual Graph

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

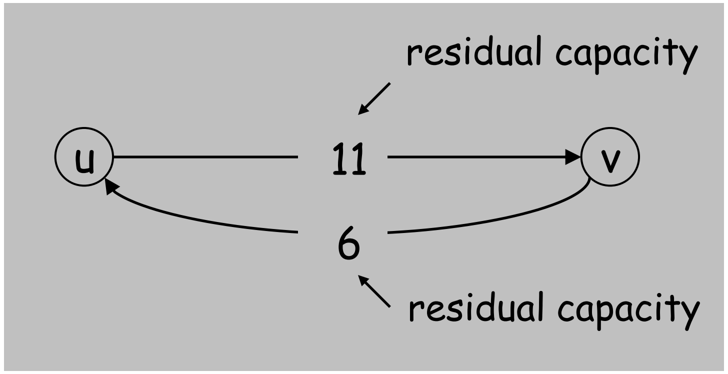
- Flow f(e), capacity c(e).



Residual edge.

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

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

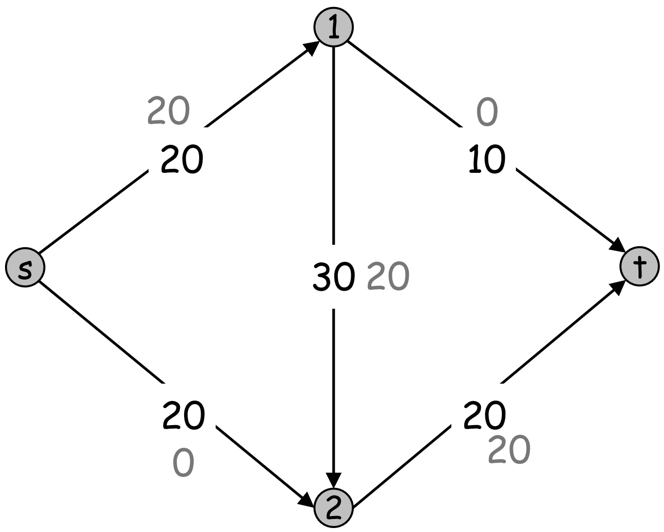


Residual graph: $G_f = (V, E_f)$.

- Residual edges with positive residual capacity.
- $E_f = \{e \in E : f(e) < c(e)\} \cup \{e^R : e \in E \text{ and } f(e) > 0\}$.

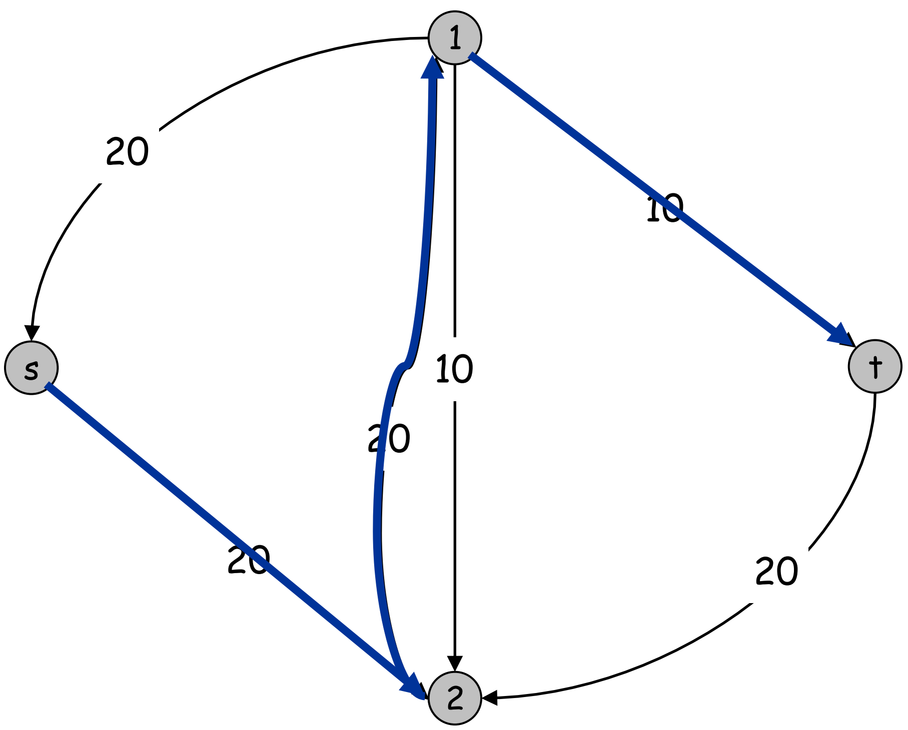
Residual graph (for Ford-Fulkerson)

original graph with flow:



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

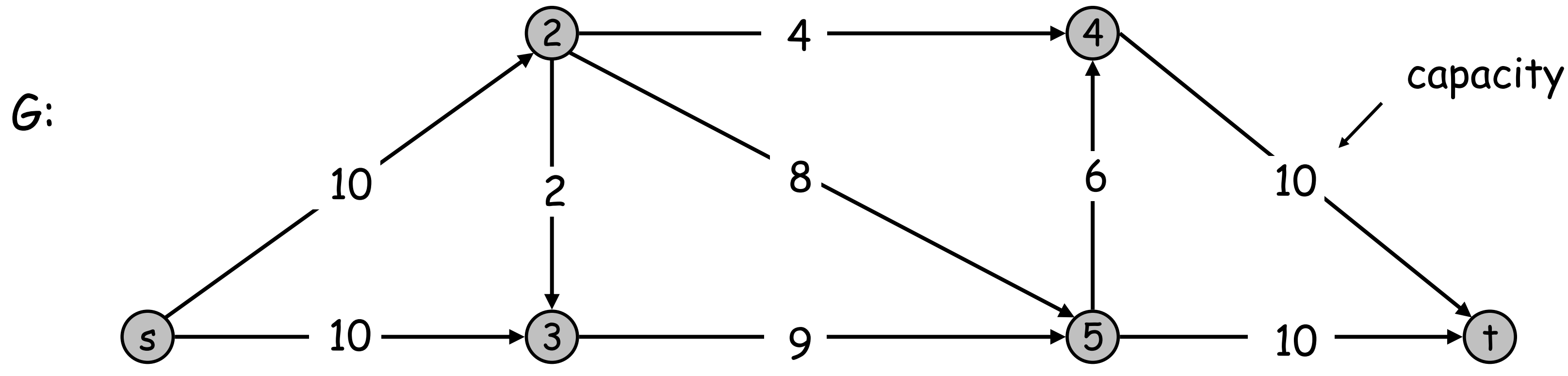
residual graph:



Ford-Fulkerson Algorithm.

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

Ford-Fulkerson Algorithm

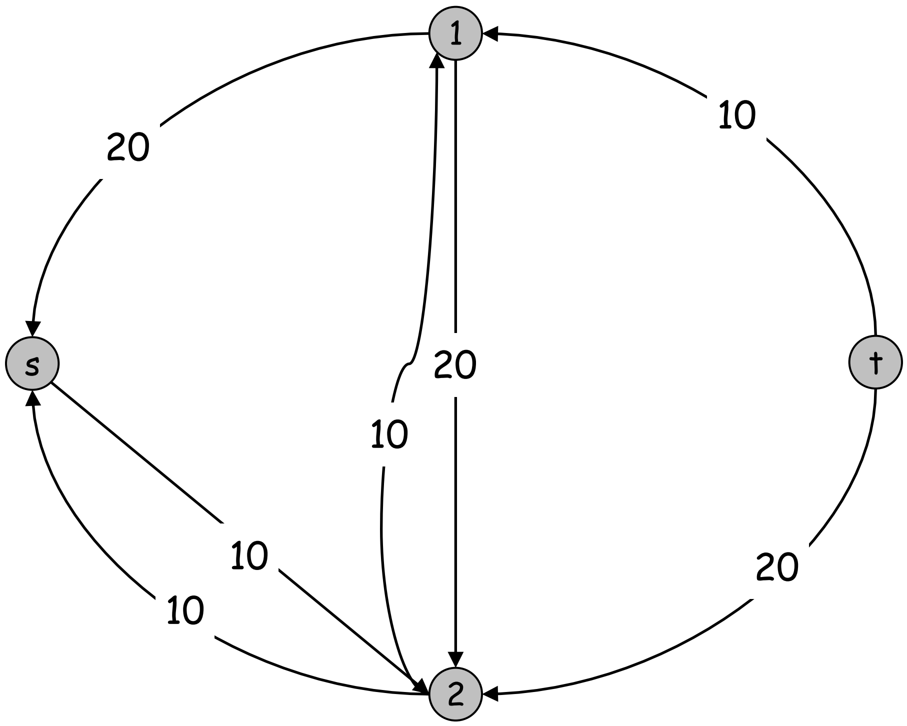


Q. How to find an augmenting path?

A. Depth-first search or breadth-first search from s

Residual graph

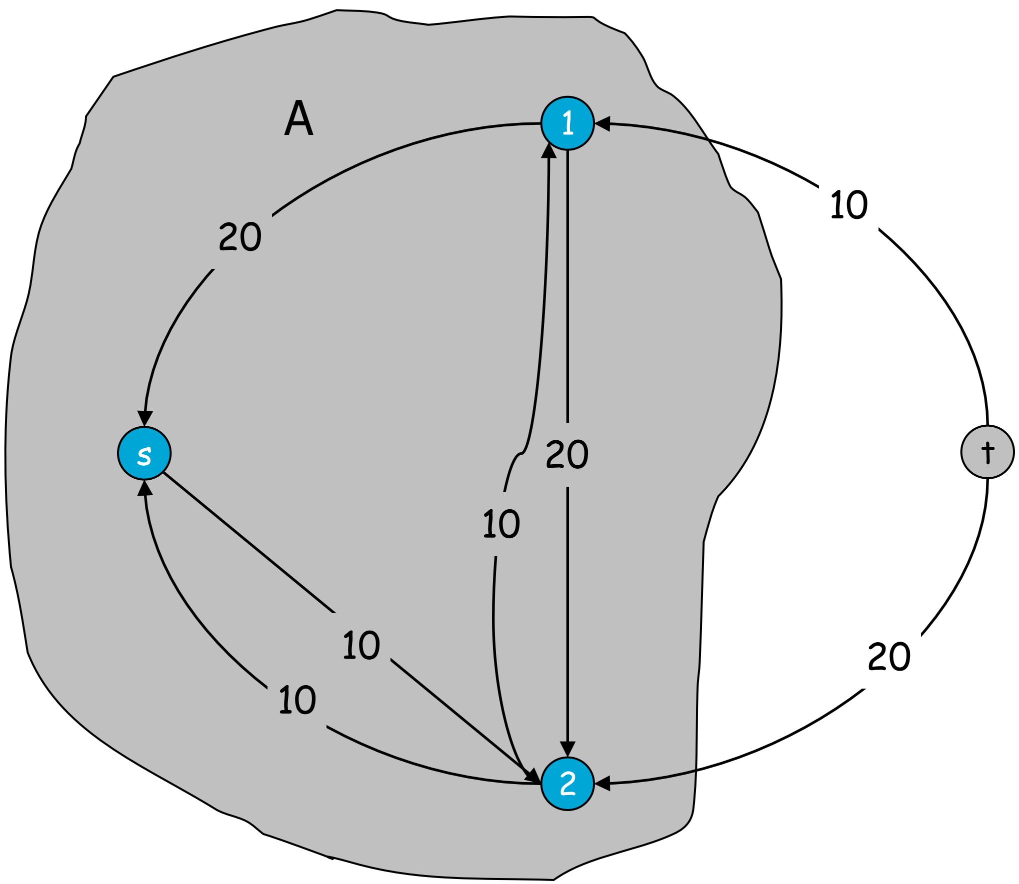
Q. How can we find the minimum cut (A,B)?



Residual graph

Q. How can we find the minimum cut (A,B)?

A. Take A = all nodes reachable in the residual graph and B = the rest.



Augmenting Path Algorithm

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

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

   while (there exists augmenting path P from s to t) {
      f ← Augment(f, c, P)
      update Gf
   }
   return f
}
```

Q. Is this algorithm correct?

TU Delft

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. [Ford-Fulkerson 1956] The value of the max flow is equal to the capacity of the min cut.

the following are equivalent

Proof strategy. We prove both simultaneously by showing TFAE:

(i) There exists a 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.

(i) $\Rightarrow$ (ii) Q. Where did we see this one before?

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. [Ford-Fulkerson 1956] The value of the max flow is equal to the capacity of the min cut.

the following are equivalent

Proof strategy. We prove both simultaneously by showing TFAE:

(i) There exists a 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.

(i) $\Rightarrow$ (ii) This was the corollary to the weak duality lemma.

(ii) $\Rightarrow$ (iii) We show the contrapositive, i.e. $\neg$(iii) $\Rightarrow$ $\neg$(ii)

- Let f be a flow. If there exists an augmenting path, then we can improve f by sending flow along this path. Thus f is not a max flow.

Q. What do we still need to proof?

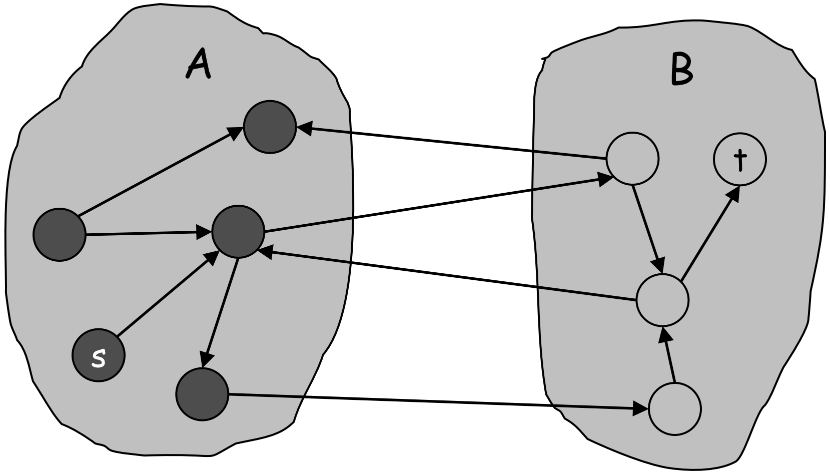
Proof of Max-Flow Min-Cut Theorem

(iii) $\Rightarrow$ (i), i.e. if (iii) there is no augmenting path relative to f
then (i) a cut (A, B) exists such that v(f) = cap(A, B).

Pf.

- Let f be a flow with no augmenting paths.

Q. Which cut (A,B) should we take to show that v(f) = cap(A, B)?



original network

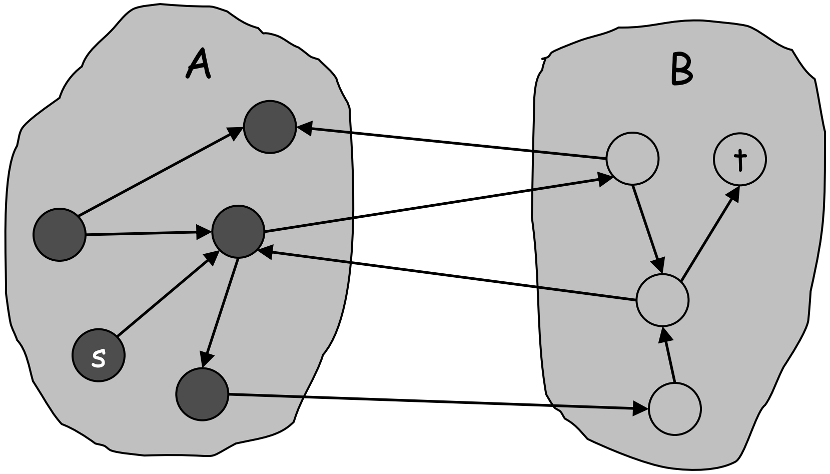
Proof of Max-Flow Min-Cut Theorem

(iii) $\Rightarrow$ (i), i.e. if (iii) there is no augmenting path relative to f
then (i) a cut (A, B) exists such that v(f) = cap(A, B).

Pf.

- Let f be a flow with no augmenting paths.
- Let A be the set of vertices reachable from s in residual graph.
- (A,B) is a cut, because s $\in$ A and because no path to t in G_f, t $\notin$ A.

Q. What do we know about v(f) then?



original network

$$v(f) = $$
$$= \vdots$$
$$= cap(A,B)$$

Proof of Max-Flow Min-Cut Theorem

(iii) $\Rightarrow$ (i), i.e. if (iii) there is no augmenting path relative to f
then (i) a cut (A, B) exists such that v(f) = cap(A, B).

Pf.

- Let f be a flow with no augmenting paths.
- Let A be the set of vertices reachable from s in residual graph.
- (A,B) is a cut, because s $\in$ A and because no path to t in G_f, t $\notin$ A.

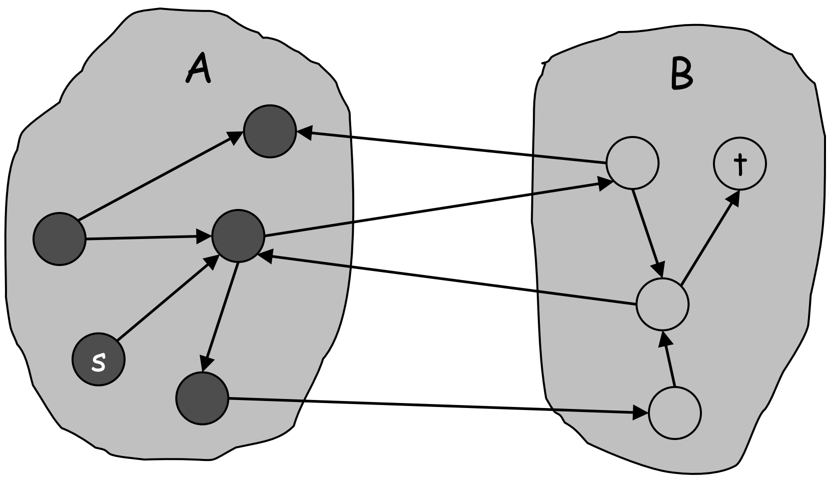
Q. What do we know about $\sum_{e \text{ out of } A} f(e)$?

flow value lemma

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

$$= \vdots$$

$$= cap(A,B)$$



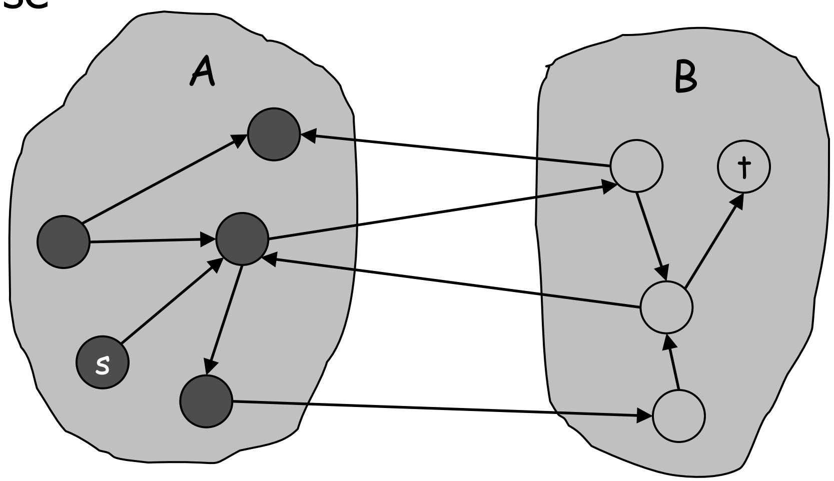
original network

Proof of Max-Flow Min-Cut Theorem

(iii) $\Rightarrow$ (i), i.e. if (iii) there is no augmenting path relative to f then (i) a cut (A, B) exists such that v(f) = cap(A, B).

Pf.

- Let f be a flow with no augmenting paths.
- Let A be the set of vertices reachable from s in residual graph.
- (A,B) is a cut, because s $\in$ A and because no path to t in G_f, t $\notin$ A.
- flow f(u,v) out of A is c(u,v), otherwise v reachable in residual graph
- so $\sum_{e \text{ out of } A} f(e) = \sum_{e \text{ out of } A} c(e)$



original network

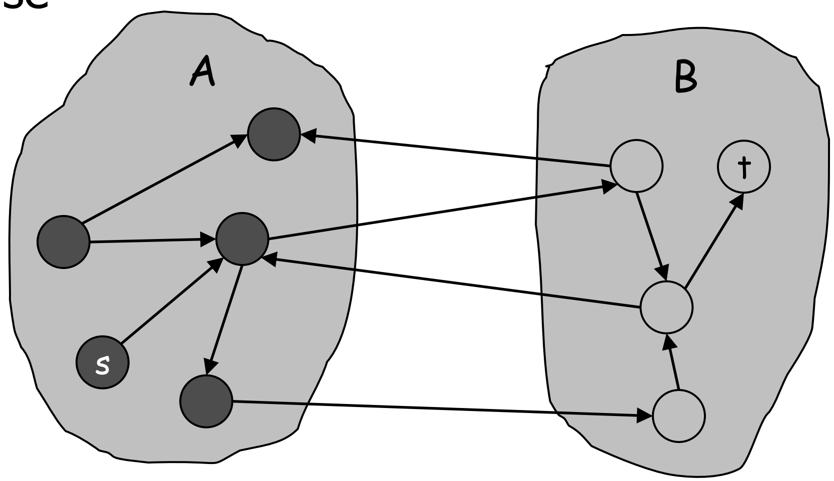
$$\begin{aligned} v(f) &= \sum_{e \text{ out of } A} f(e) - \sum_{e \text{ in to } A} f(e) \\ &= \sum_{e \text{ out of } A} c(e) - \ldots \\ &= cap(A,B) \end{aligned}$$

Proof of Max-Flow Min-Cut Theorem

(iii) $\Rightarrow$ (i), i.e. if (iii) there is no augmenting path relative to f
then (i) a cut (A, B) exists such that v(f) = cap(A, B).

Pf.

- Let f be a flow with no augmenting paths.
- Let A be the set of vertices reachable from s in residual graph.
- (A,B) is a cut, because s $\in$ A and because no path to t in G_f, t $\notin$ A.
- flow f(u,v) out of A is c(u,v), otherwise
  v reachable in residual graph
- so $\sum_{e \text{ out of } A} f(e) = \sum_{e \text{ out of } A} c(e)$



original network

Q. What do we know about $\sum_{e \text{ in to } A} f(e)$?

$$
\begin{aligned}
v(f) &= \sum_{e \text{ out of } A} f(e) - \sum_{e \text{ in to } A} f(e) \\
&= \sum_{e \text{ out of } A} c(e) - \ldots \\
&= cap(A,B)
\end{aligned}
$$

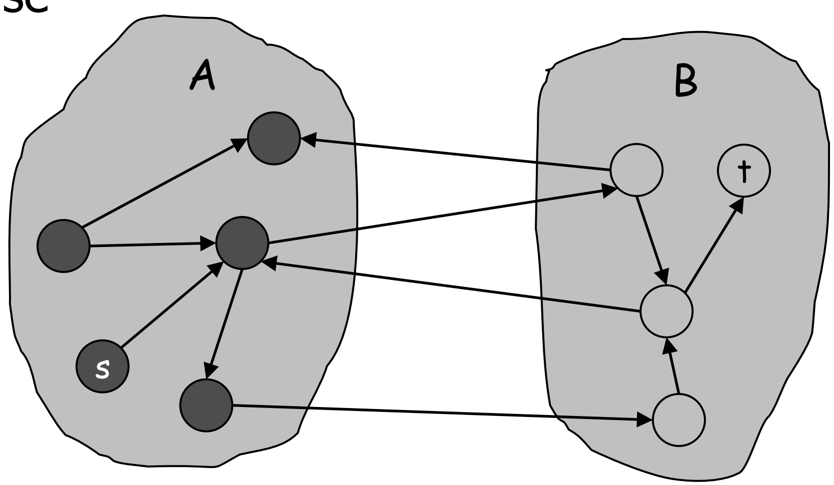
Proof of Max-Flow Min-Cut Theorem

(iii) $\Rightarrow$ (i), i.e. if (iii) there is no augmenting path relative to f
then (i) a cut (A, B) exists such that v(f) = cap(A, B).

Pf.

- Let f be a flow with no augmenting paths.
- Let A be the set of vertices reachable from s in residual graph.
- (A,B) is a cut, because s $\in$ A and because no path to t in G_f, t $\notin$ A.
- flow f(u,v) out of A is c(u,v), otherwise v reachable in residual graph
- so $\sum_{e \text{ out of } A} f(e) = \sum_{e \text{ out of } A} c(e)$
- flow f(u,v) into A is zero, otherwise u reachable in residual graph
- so

$$
\begin{aligned}
v(f) &= \sum_{e \text{ out of } A} f(e) - \sum_{e \text{ in to } A} f(e) \\
&= \sum_{e \text{ out of } A} c(e) - 0 \\
&= cap(A,B)
\end{aligned}
$$



original network

This proof can be found on page 348-349.

Max-Flow Min-Cut Theorem

Proof (summary). We have now shown that:

- (i) $\Rightarrow$ (ii)
- (ii) $\Rightarrow$ (iii)
- (iii) $\Rightarrow$ (i)
- So, TFAE: (the following are equivalent)

  (i) There exists a 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.

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

Pf. (ii) $\Leftrightarrow$ (iii)

Max-flow min-cut theorem. [Ford-Fulkerson 1956] The value of the max flow is equal to the capacity of the min cut.

Pf. (i) $\Leftrightarrow$ (ii), so cap(A, B) = v(f) is max flow. Corollary: (A, B) is min cut.

This proof can be found on page 350.

Augmenting Path Algorithm

```
Augment(f, c, P) {
   b ← bottleneck(P,c)
   foreach e ∈ P {
      if (e ∈ E) f(e) ← f(e) + b       forward edge
      else       f(e^R) ← f(e) - b     reverse edge
   }
   return f
}
```

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

   while (there exists augmenting path P from s to t) {
      f ← Augment(f, c, P)
      update G_f
   }
   return f
}
```

Q. What is the run-time complexity of one iteration of the while?

Augmenting Path Algorithm

```
Augment(f, c, P) {
   b ← bottleneck(P,c)
   foreach e ∈ P {
      if (e ∈ E) f(e) ← f(e) + b        forward edge
      else       f(e^R) ← f(e) - b      reverse edge
   }
   return f
}
```

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

   while (there exists augmenting path P from s to t) {
      f ← Augment(f, c, P)
      update Gf
   }
   return f
}
```

Q. What is the run-time complexity of one iteration of the while?

A. O(m+n) for finding a path, O(n) to augment, so O(m)

Running Time

Q. How many iterations until maximum flow? What does it depend upon?
A. The value of the maximum flow, which depends on the capacities.

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.

Integrality theorem. If all capacities are integers, then there exists a max flow f for which every flow value f(e) is an integer.
Pf. Since algorithm terminates, theorem follows from invariant. ▪

Q. What is the value of the maximum possible flow?
A. Maximum possible flow is nc*, since at most n neighbors of s.

Q. What is the time complexity of Ford-Fulkerson?

Running Time

Q. What is the time complexity of Ford-Fulkerson?

Theorem. F-F terminates in at most $v(f^*) \leq nc^*$ iterations, so $O(mnc^*)$.

Pf. Maximum possible flow is nc^*, since at most n neighbors of s.
Each augmentation increase value by at least 1.
O(m) per augmenting path. ▪

Q. What is the run time if $c^* = 1$?

Running Time

Q. What is the time complexity of Ford-Fulkerson?

Theorem. F-F terminates in at most $v(f^*) \leq nc^*$ iterations, so $O(mnc^*)$.

Pf. Maximum possible flow is nc^*, since at most n neighbors of s.
Each augmentation increase value by at least 1.
$O(m)$ per augmenting path. ▪

Corollary. If $c^* = 1$, Ford-Fulkerson runs in $O(nm)$ time.

TU Delft