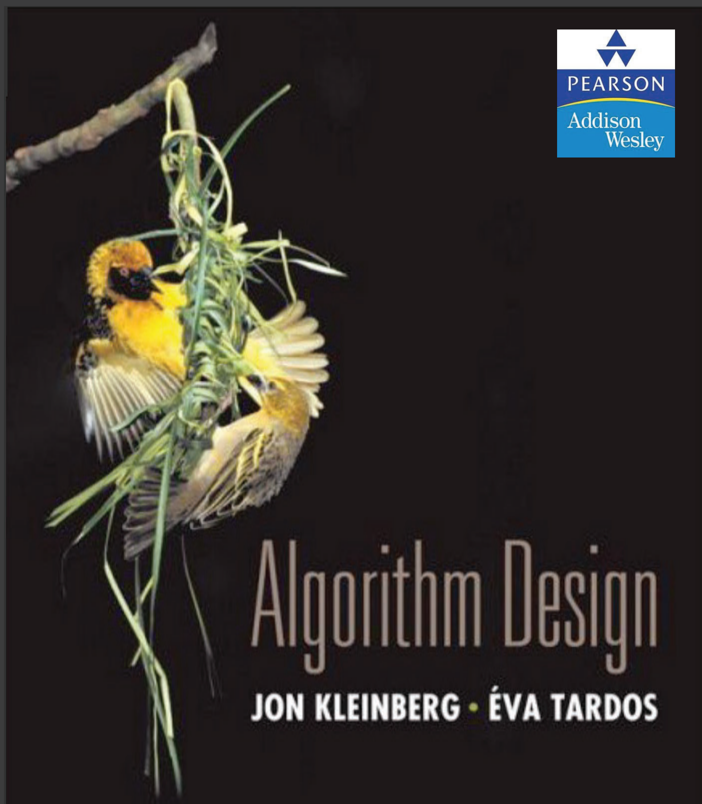


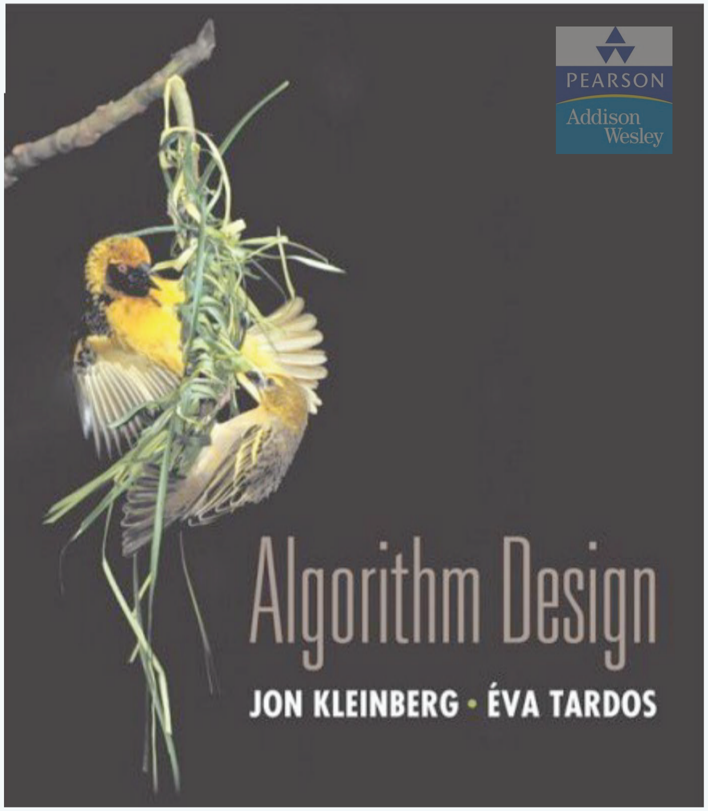
7. NETWORK FLOW I

- *max-flow and min-cut problems*
- *Ford-Fulkerson algorithm*
- *max-flow min-cut theorem*
- *capacity-scaling algorithm*
- *shortest augmenting paths*
- *blocking-flow algorithm*
- *unit-capacity simple networks*

Lecture slides by Kevin Wayne

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

Last updated on Apr 21, 2013 7:35 PM



SECTION 7.1

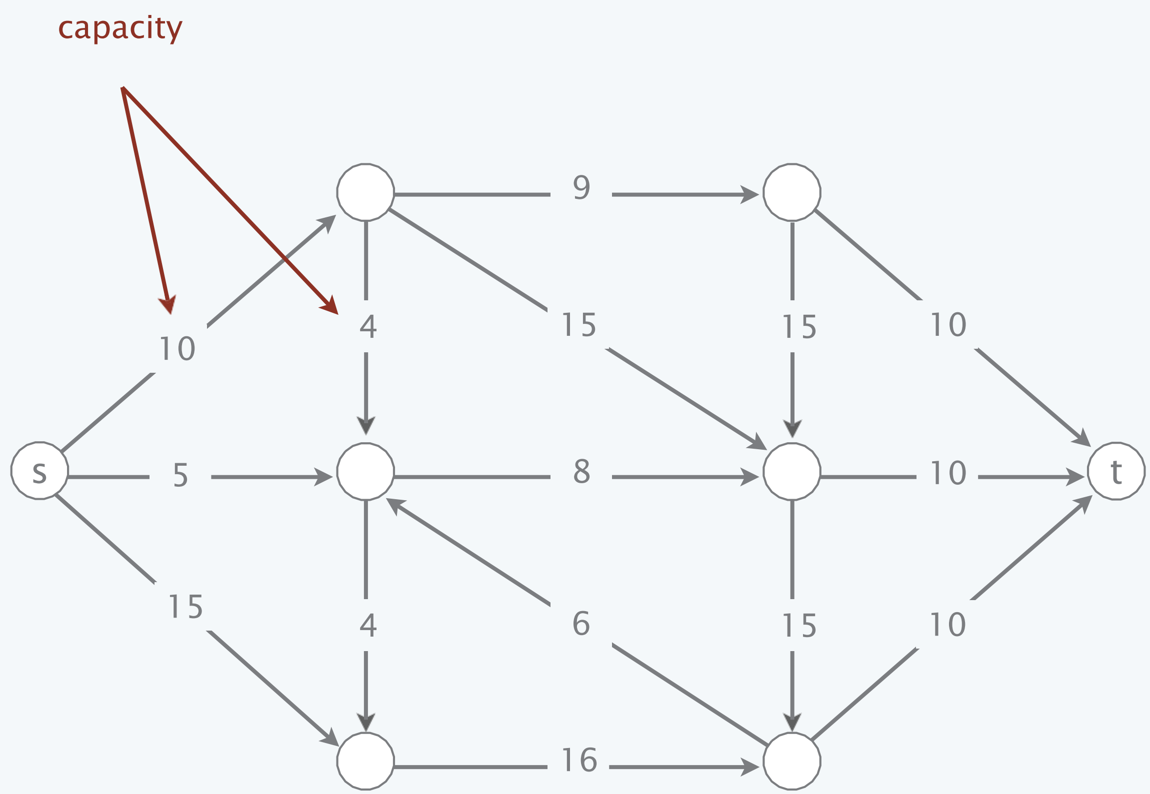
7. NETWORK FLOW I

- *max-flow and min-cut problems*
- *Ford-Fulkerson algorithm*
- *max-flow min-cut theorem*
- *capacity-scaling algorithm*
- *shortest augmenting paths*
- *blocking-flow algorithm*
- *unit-capacity simple networks*

Flow network

- Abstraction for material flowing through the edges.
- Digraph $G = (V, E)$ with source $s \in V$ and sink $t \in V$.
- Nonnegative integer capacity $c(e)$ for each $e \in E$.

no parallel edges
no edge enters s
no edge leaves t

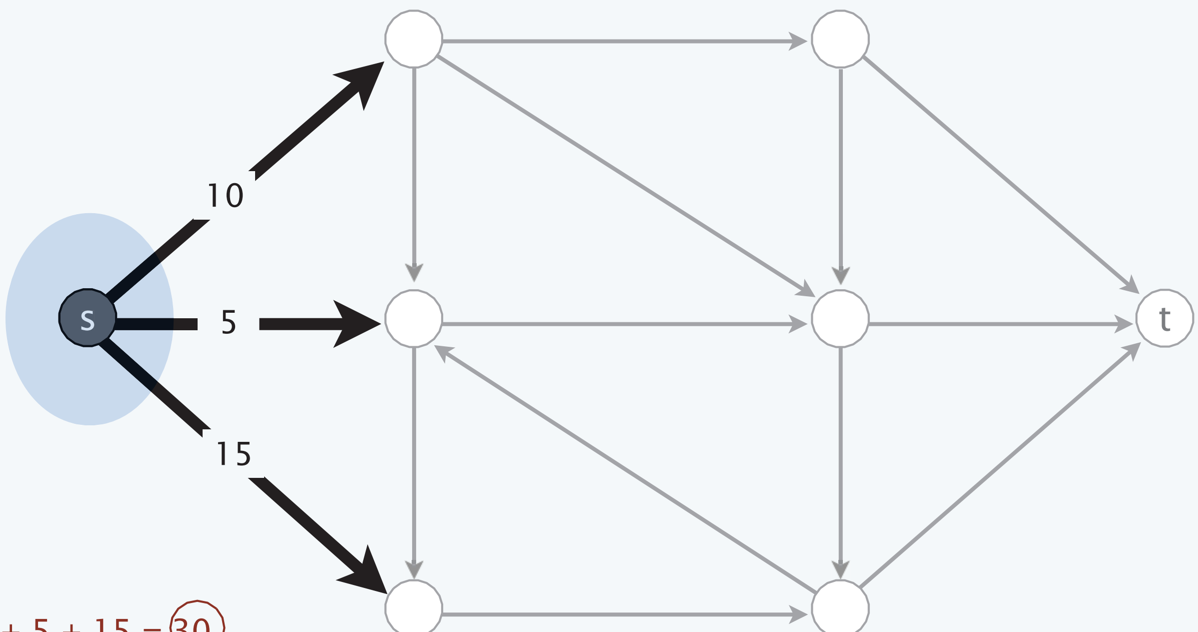


Minimum cut problem

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

Def. Its capacity is the sum of the capacities of the edges from A to B.

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



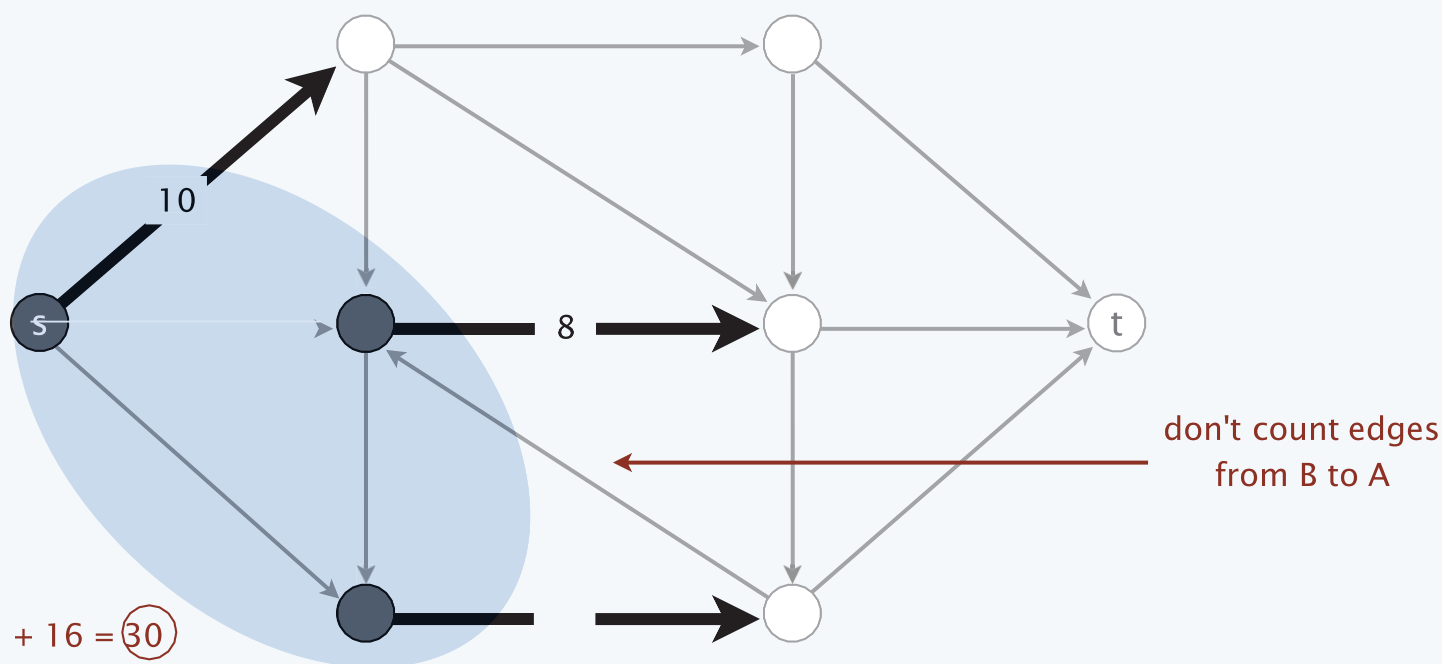
capacity = 10 + 5 + 15 = 30

Minimum cut problem

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

Def. Its capacity is the sum of the capacities of the edges from A to B.

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



capacity = 10 + 5 + 16 = 30

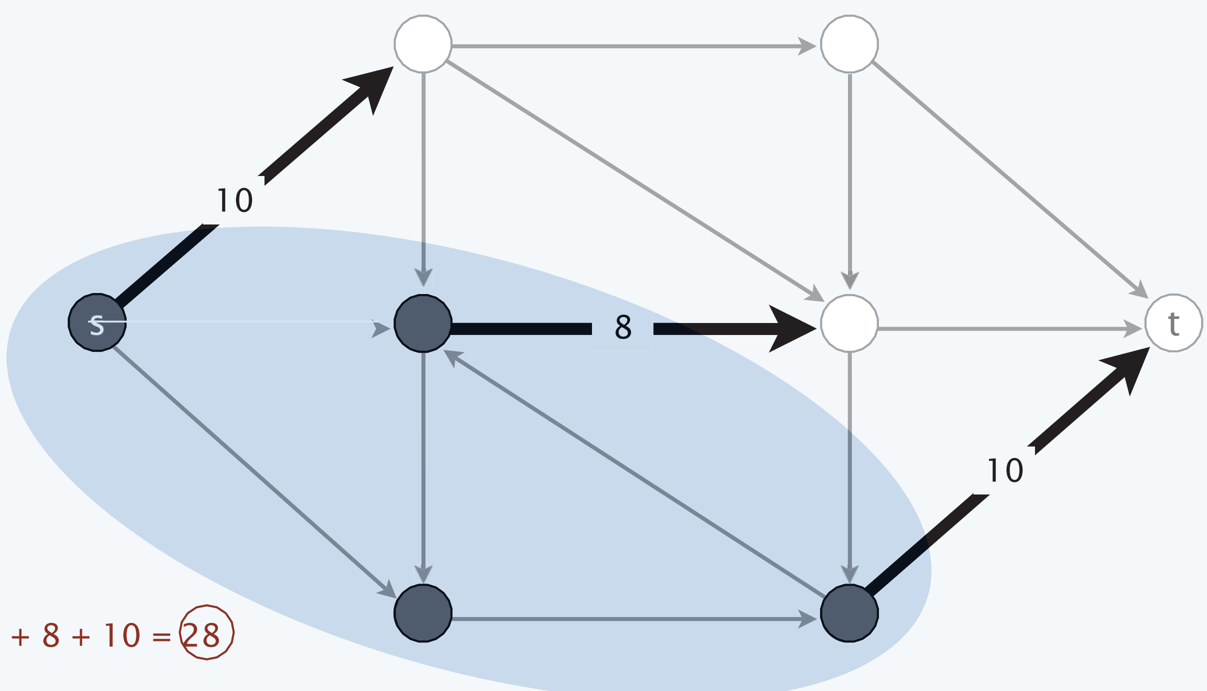
Minimum cut problem

Def. A st-cut (cut) is a partition (A, B) of the vertices with $s \in A$ and $t \in B$.

Def. Its capacity is the sum of the capacities of the edges from A to B.

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

Min-cut problem. Find a cut of minimum capacity.

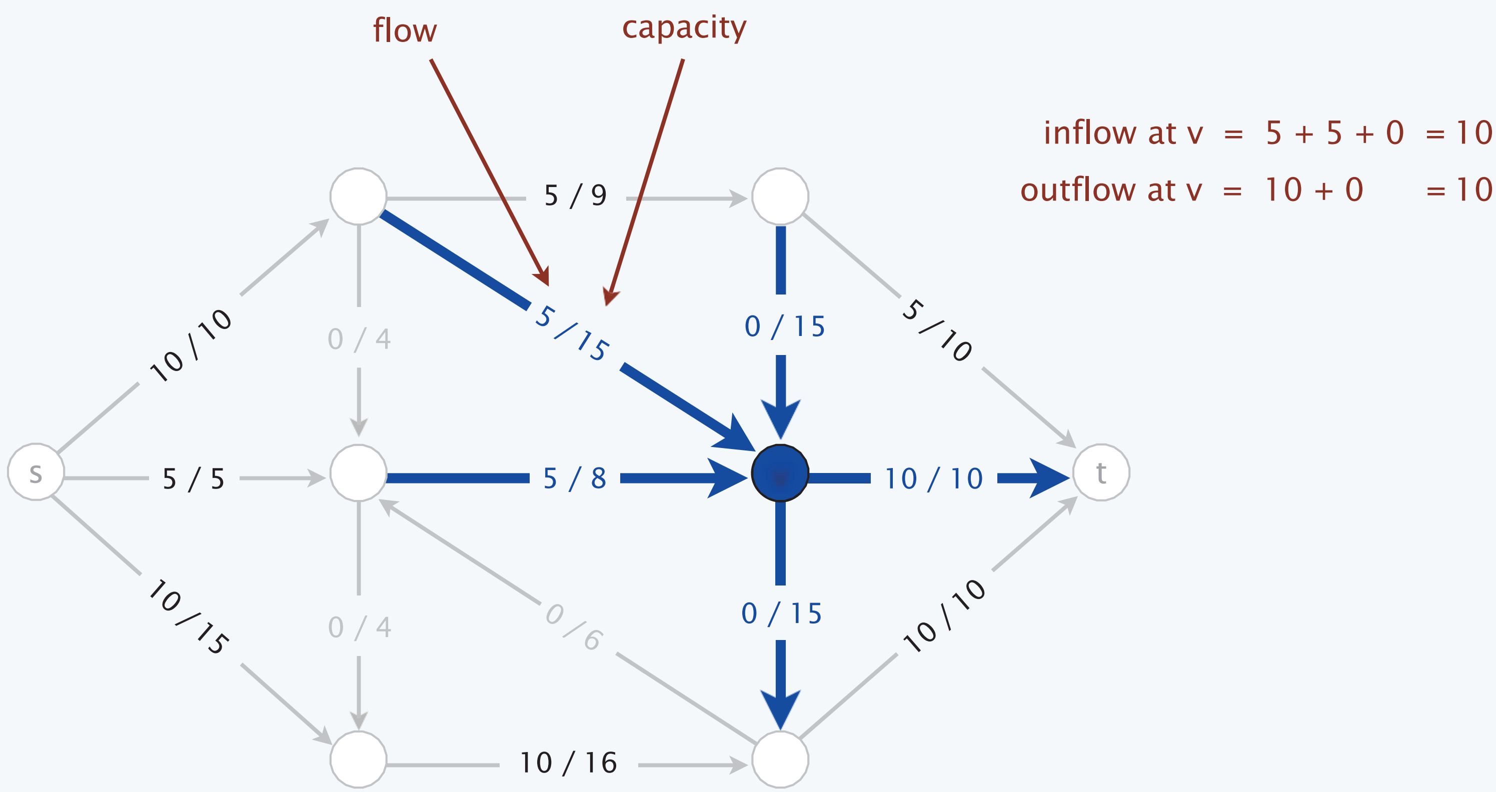


capacity = 10 + 8 + 10 = 28

Maximum flow problem

Def. An st-flow (flow) f is a function that satisfies:

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

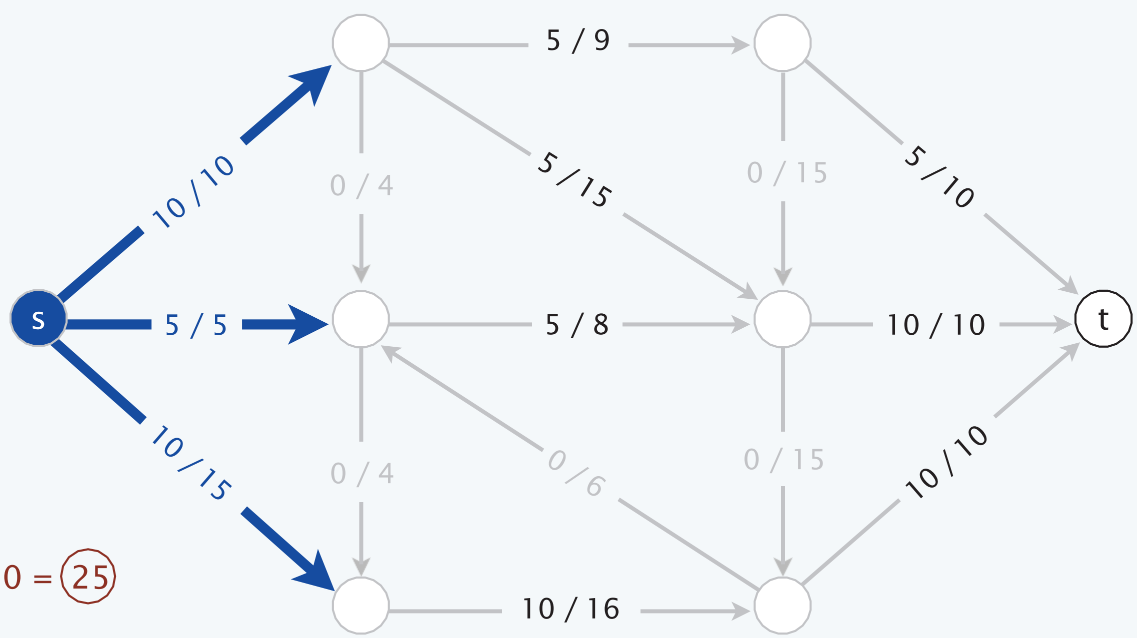


Maximum flow problem

Def. An *st*-flow (flow) f is a function that satisfies:

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

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



value = 5 + 10 + 10 = 25

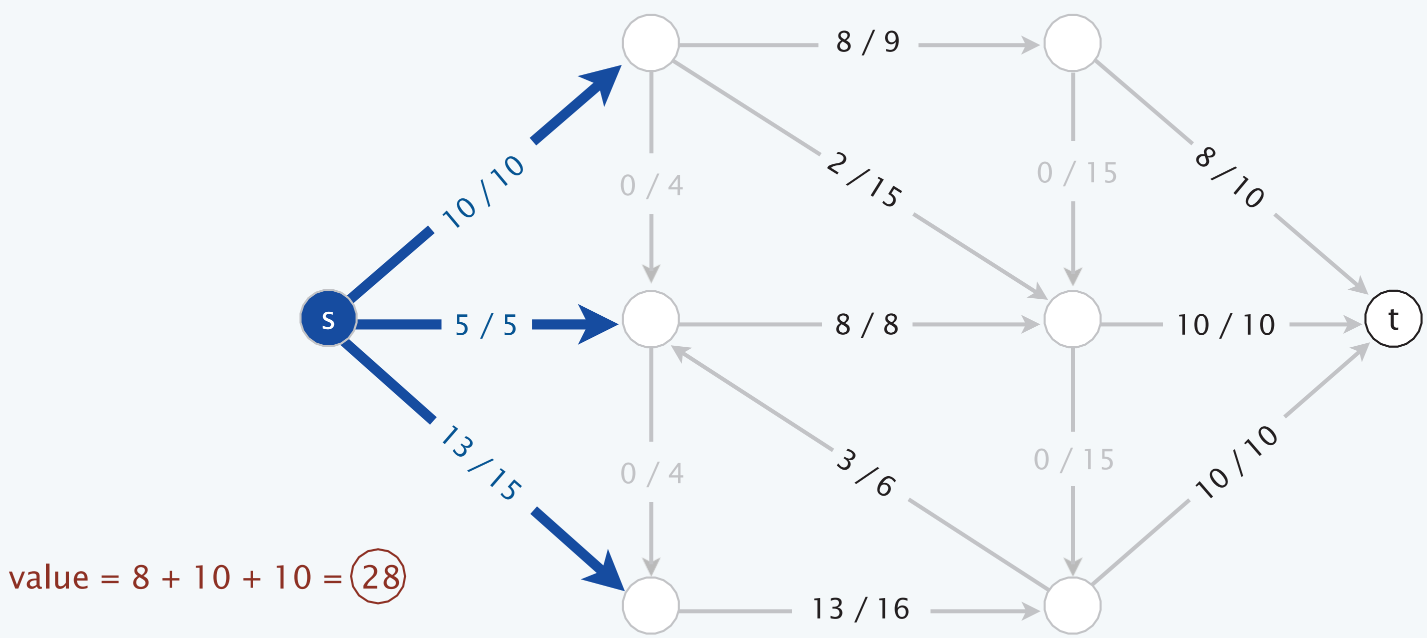
Maximum flow problem

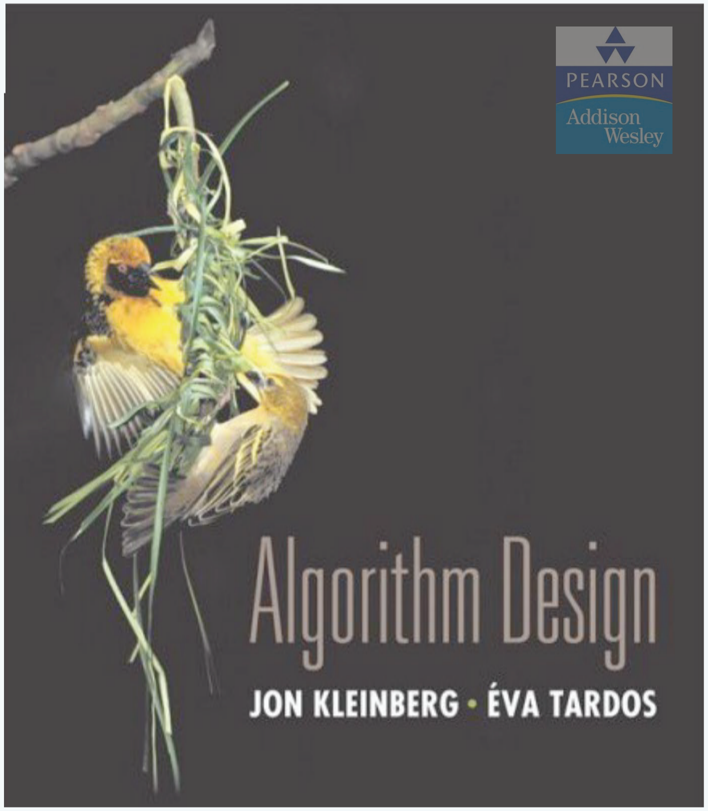
Def. An *st*-flow (flow) f is a function that satisfies:

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

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

Max-flow problem. Find a flow of maximum value.





SECTION 7.1

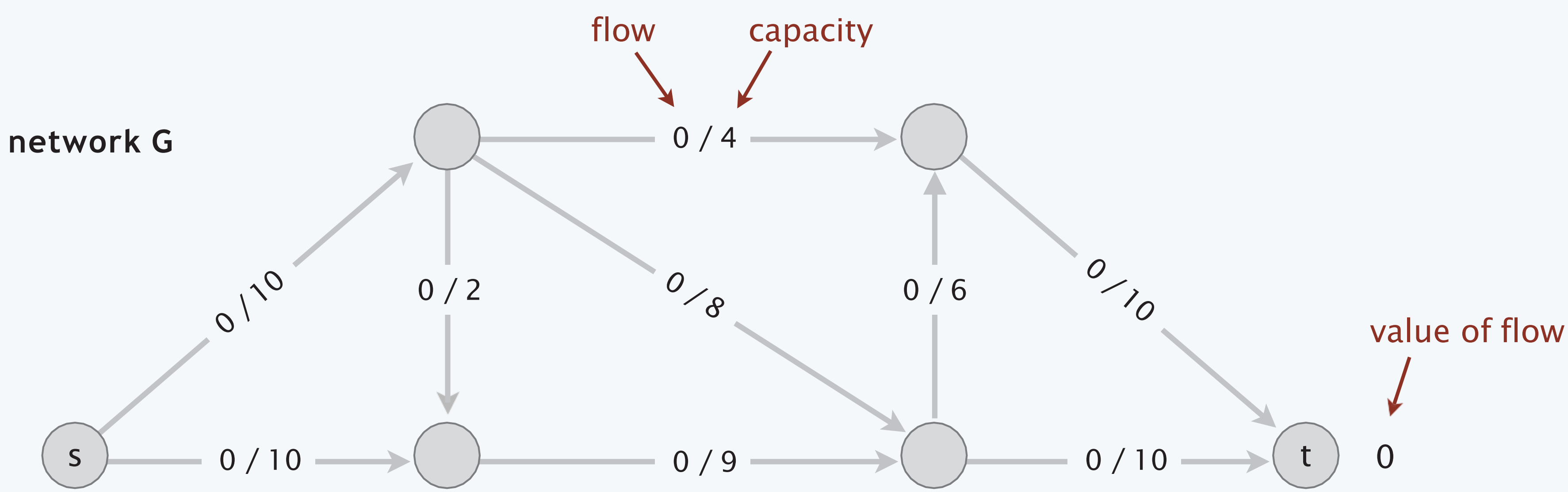
7. NETWORK FLOW I

- max-flow and min-cut problems
- Ford-Fulkerson algorithm
- max-flow min-cut theorem
- capacity-scaling algorithm
- shortest augmenting paths
- blocking-flow algorithm
- unit-capacity simple networks

Towards a max-flow algorithm

Greedy algorithm.

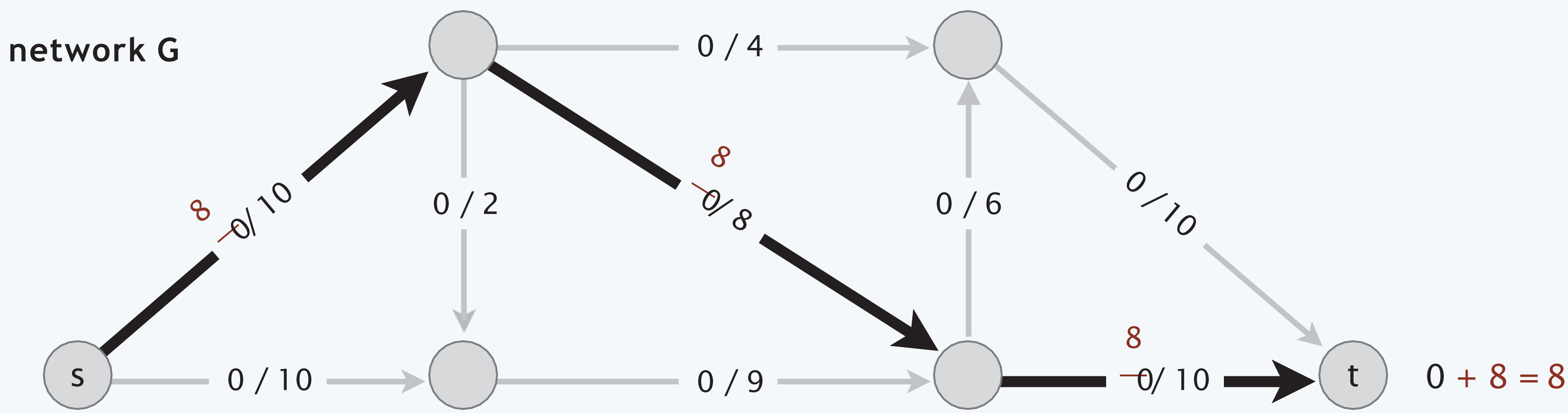
- Start with $f(e) = 0$ for all edge $e \in E$.
- Find an $s \rightsquigarrow t$ path P where each edge has $f(e) < c(e)$.
- Augment flow along path P.
- Repeat until you get stuck.



Towards a max-flow algorithm

Greedy algorithm.

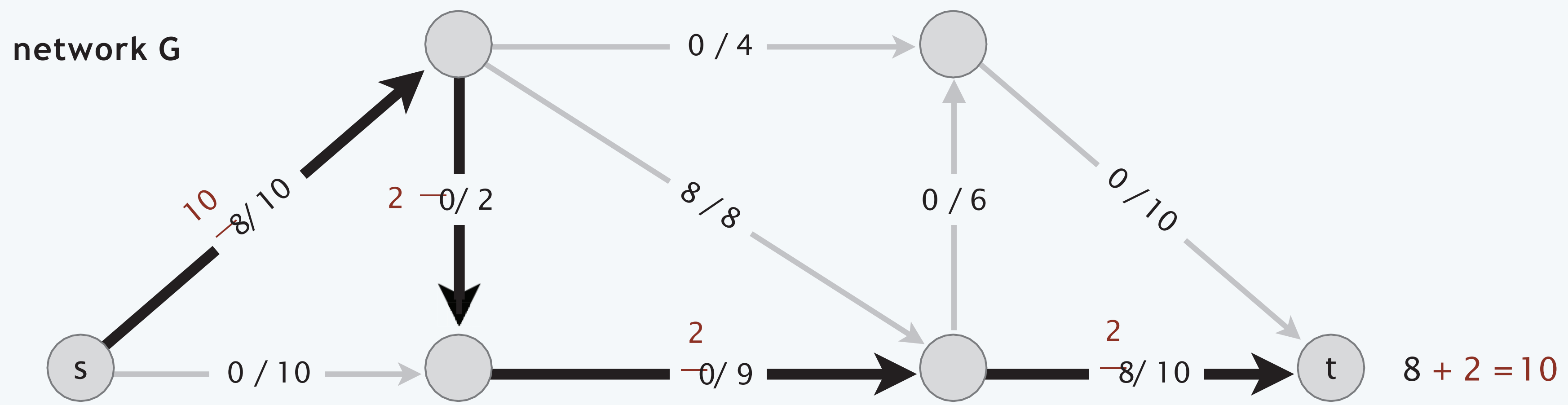
- Start with $f(e) = 0$ for all edge $e \in E$.
- Find an $s \rightsquigarrow t$ path P where each edge has $f(e) < c(e)$.
- Augment flow along path P.
- Repeat until you get stuck.



Towards a max-flow algorithm

Greedy algorithm.

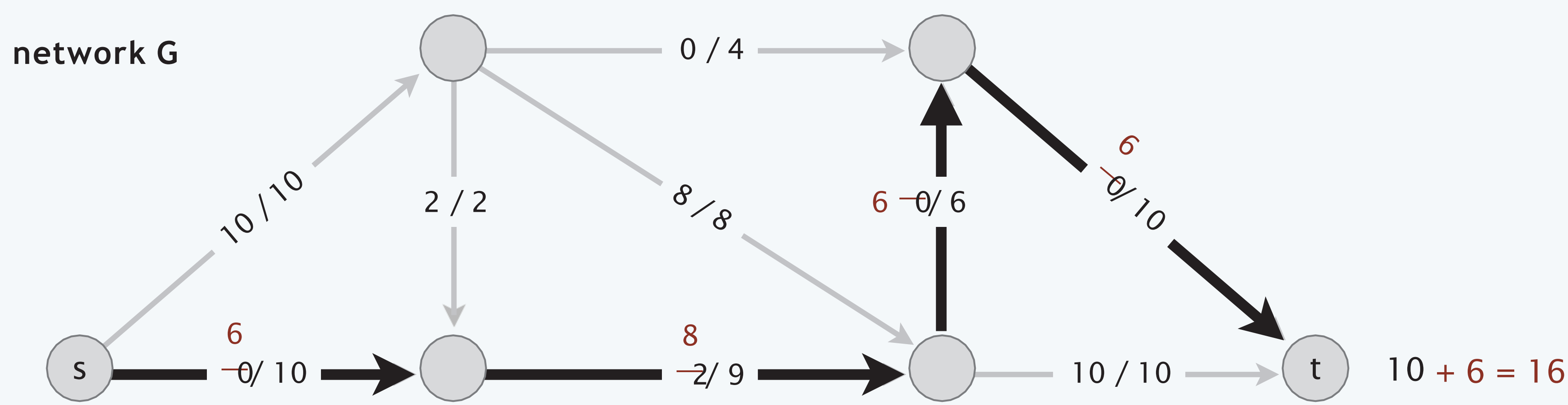
- Start with $f(e) = 0$ for all edge $e \in E$.
- Find an $s \rightsquigarrow t$ path P where each edge has $f(e) < c(e)$.
- Augment flow along path P.
- Repeat until you get stuck.



Towards a max-flow algorithm

Greedy algorithm.

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

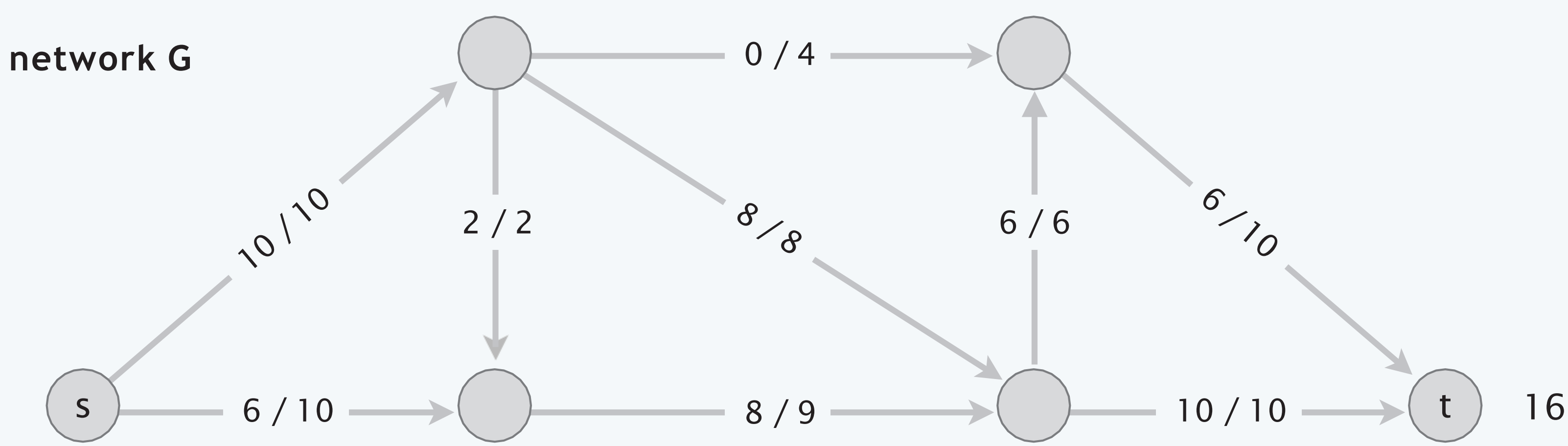


Towards a max-flow algorithm

Greedy algorithm.

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

ending flow value = 16



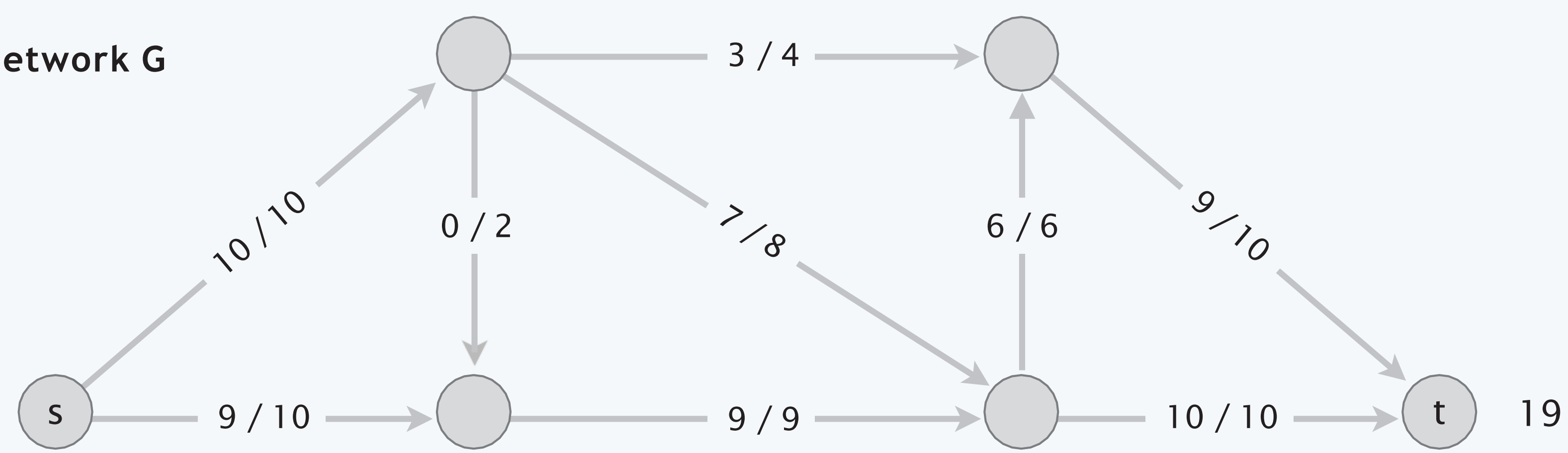
Towards a max-flow algorithm

Greedy algorithm.

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

but max-flow value = 19

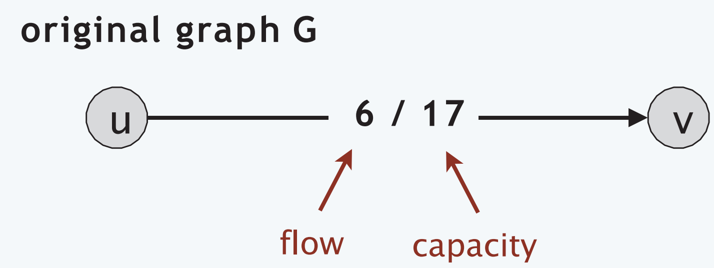
network G



Residual graph

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

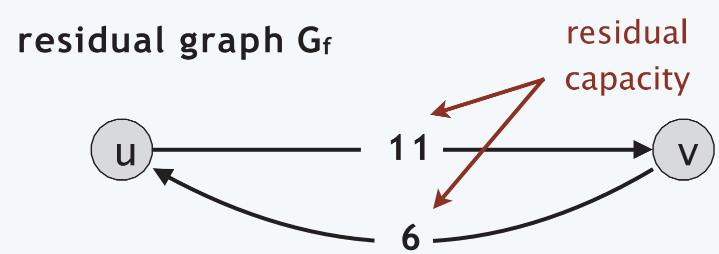
- Flow $f(e)$.
- Capacity $c(e)$.



Residual edge.

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

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



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

- Residual edges with positive residual capacity.
- $E_f = \{e : f(e) < c(e)\} \cup \{e^R : f(e) > 0\}$.
- Key property: f' is a flow in G_f iff $f + f'$ is a flow in G.

where flow on a reverse edge negates flow on a forward edge

Augmenting path

Def. An augmenting path is a simple $s \rightsquigarrow t$ path P in the residual graph G_f.

Def. The bottleneck capacity of an augmenting P is the minimum residual capacity of any edge in P.

Key property. Let f be a flow and let P be an augmenting path in G_f.
Define $f' = Augment(f,c,P)$.
Then f' is a flow and $val(f') = val(f) + bottleneck(G_f, P)$.

AUGMENT (f, c, P)

$b \leftarrow$ bottleneck capacity of path P.
FOREACH edge $e \in P$
  IF $(e \in E)$ $f(e) \leftarrow f(e) + b$.
  ELSE $f(e^R) \leftarrow f(e^R) - b$.
RETURN f.

Ford-Fulkerson algorithm

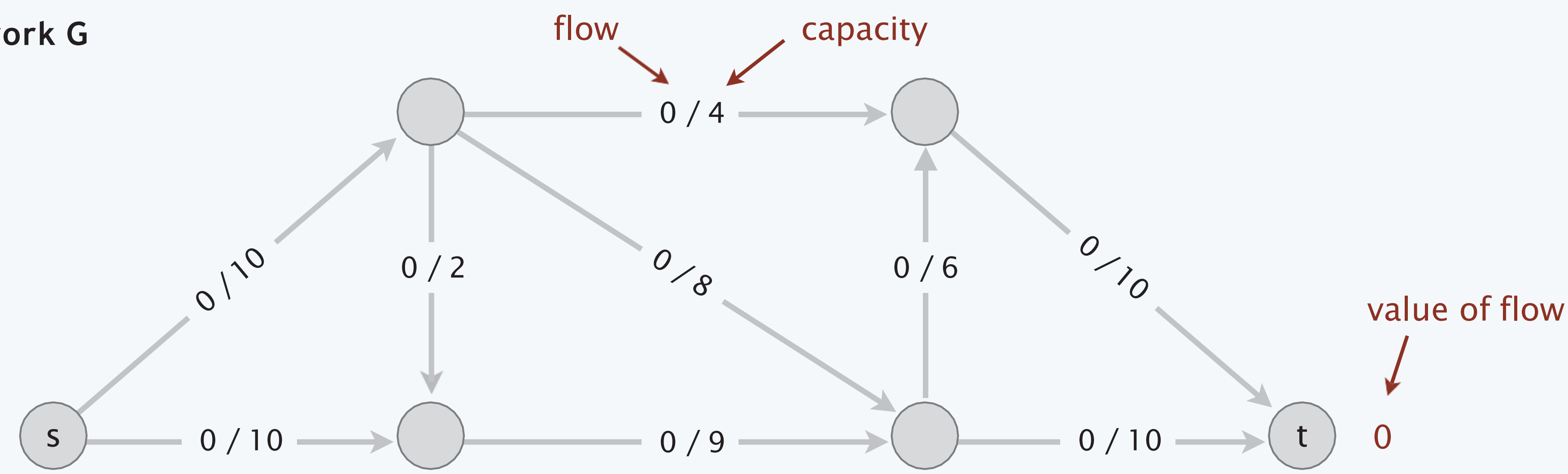
Ford-Fulkerson augmenting path algorithm.

- Start with $f(e) = 0$ for all edge $e \in E$.
- Find an augmenting path P in the residual graph G_f.
- Augment flow along path P.
- Repeat until you get stuck.

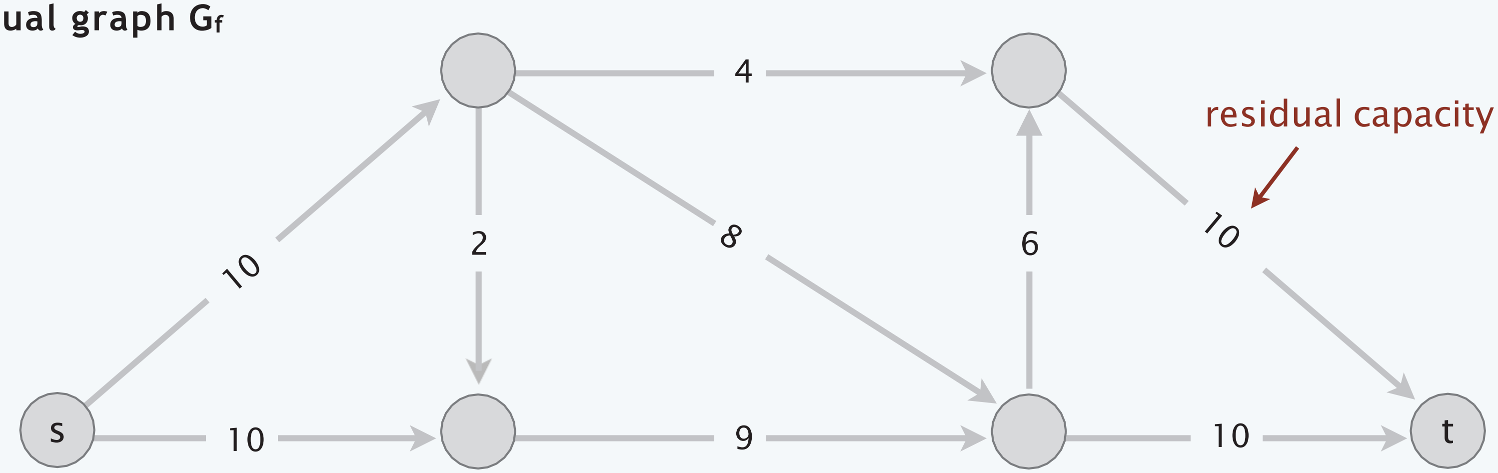
```
FORD-FULKERSON (G, s, t, c)
─────────────────────────────────────────
FOREACH edge e ∈ E : f (e) ← 0.
   G_f ← residual graph.
   WHILE (there exists an augmenting path P in G_f )
      f ← AUGMENT (f, c, P).
      Update G_f.
   RETURN f.
}
─────────────────────────────────────────
```

Ford-Fulkerson algorithm demo

network G

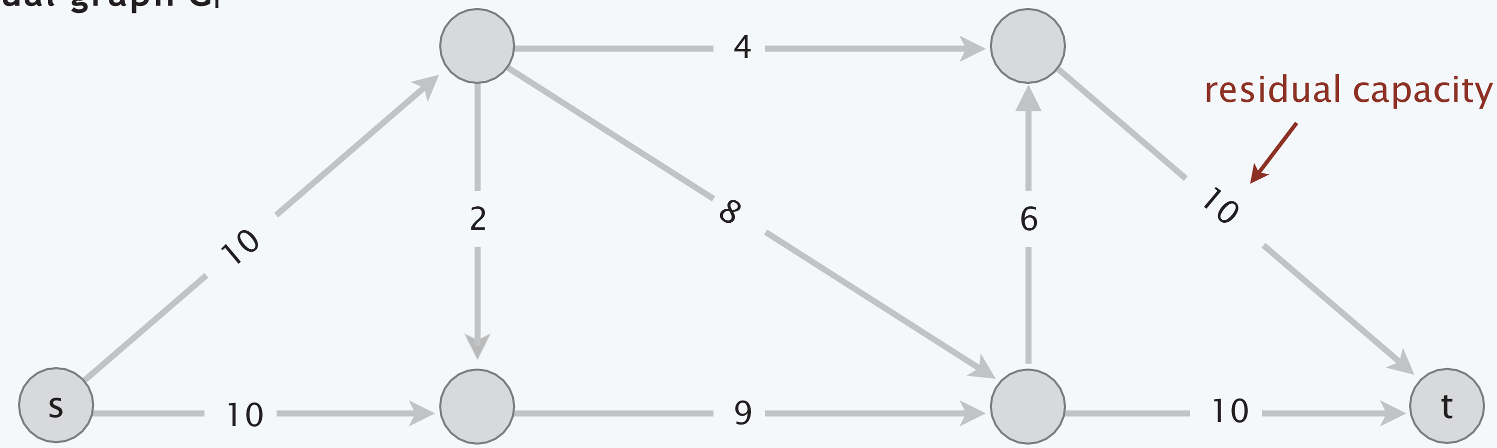


residual graph G_f

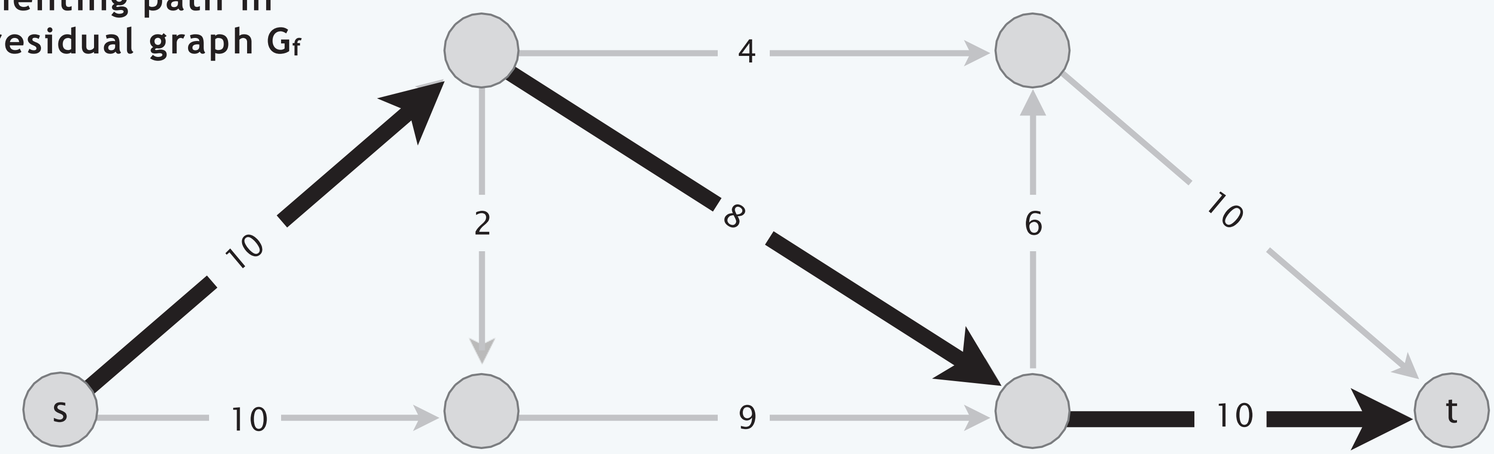


Ford-Fulkerson algorithm demo

residual graph G_f

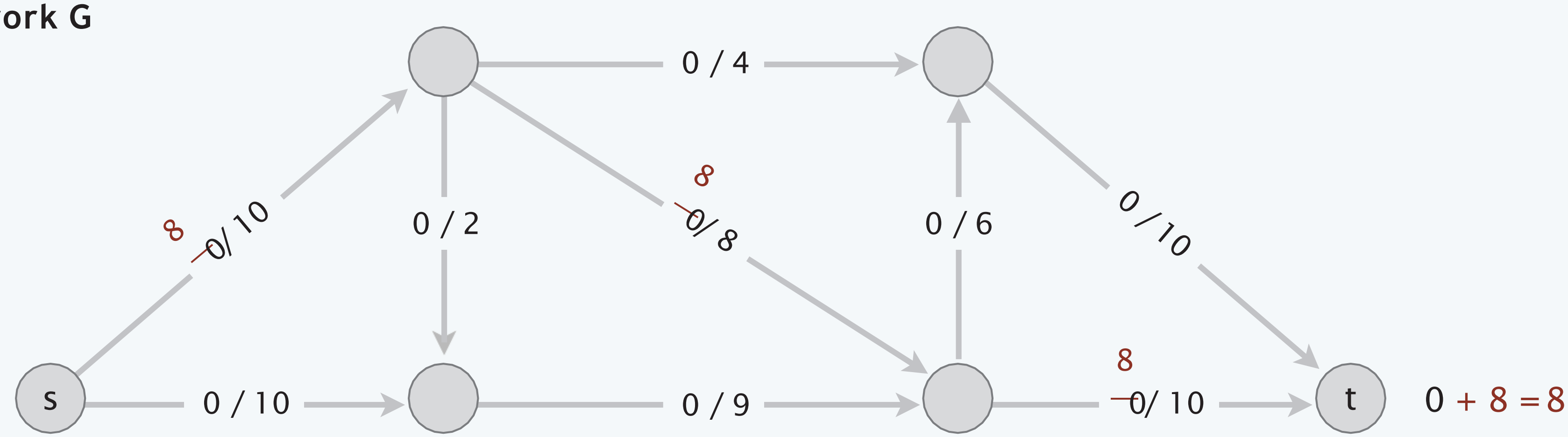


Augmenting path in the residual graph G_f

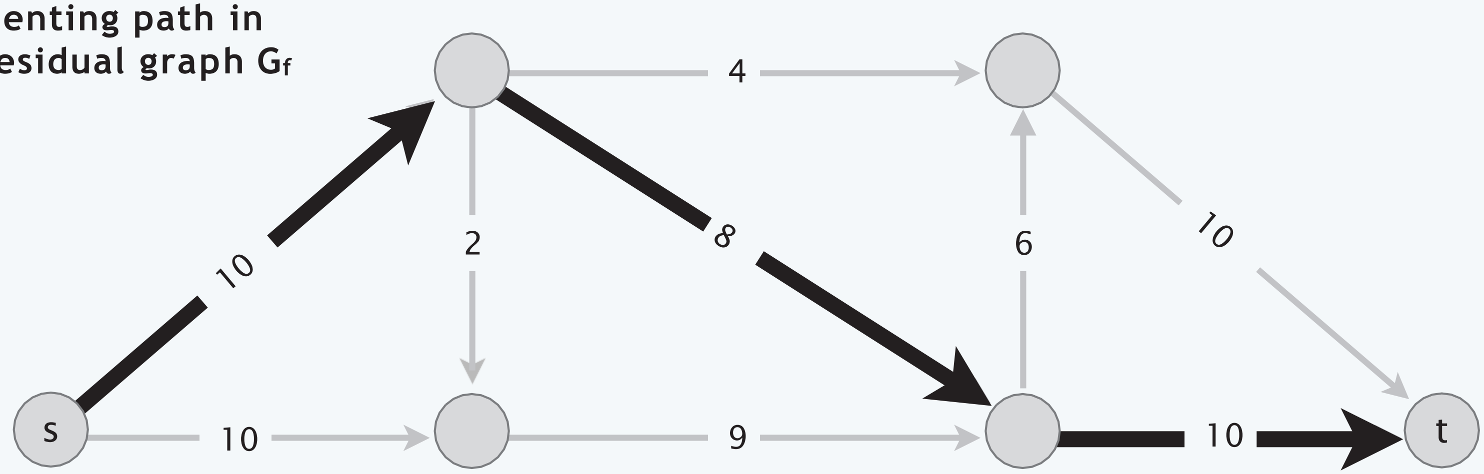


Ford-Fulkerson algorithm demo

network G

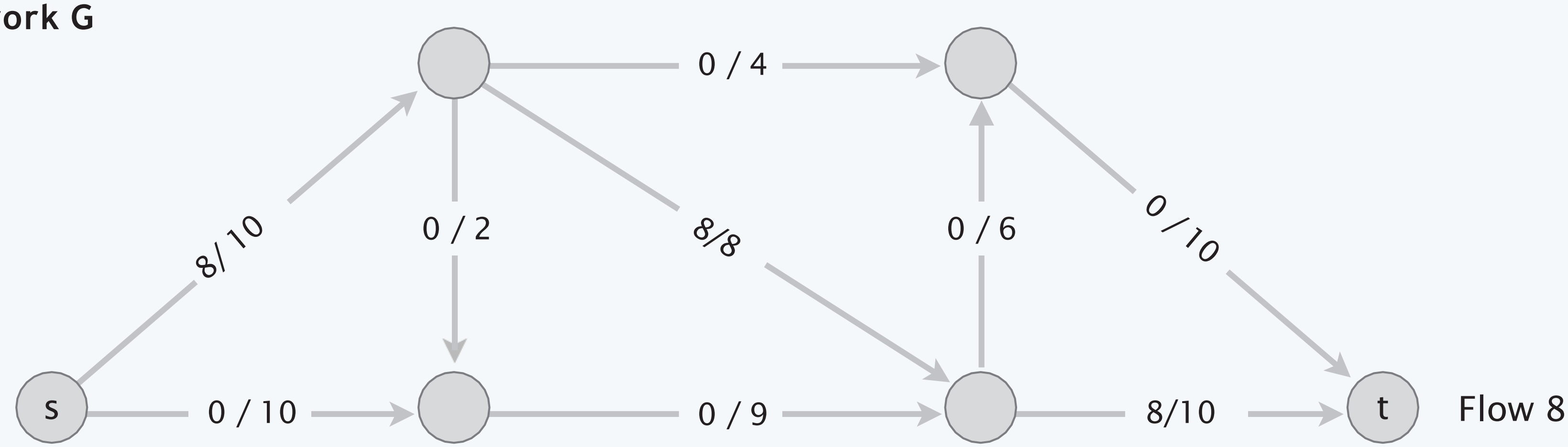


Augmenting path in the residual graph G_f

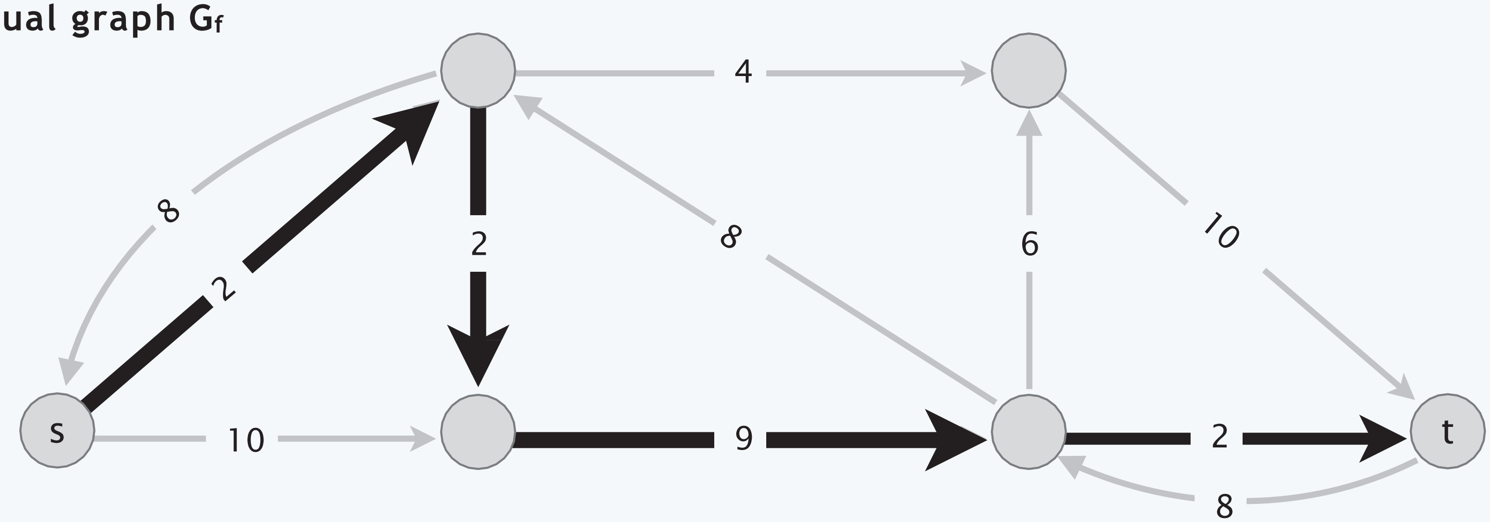


Ford-Fulkerson algorithm demo

network G

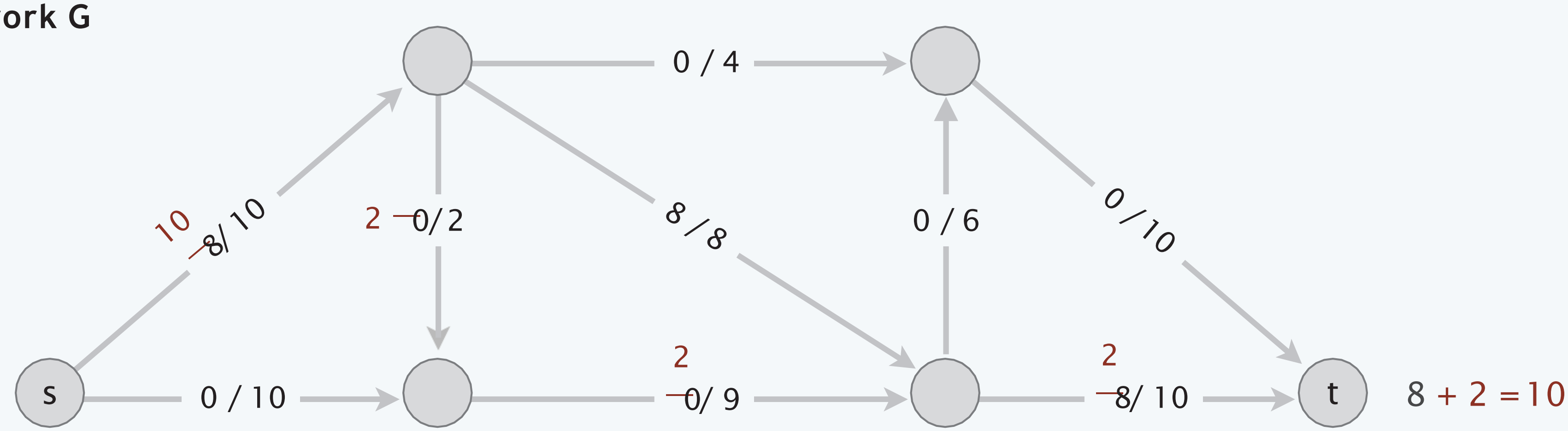


residual graph G_f

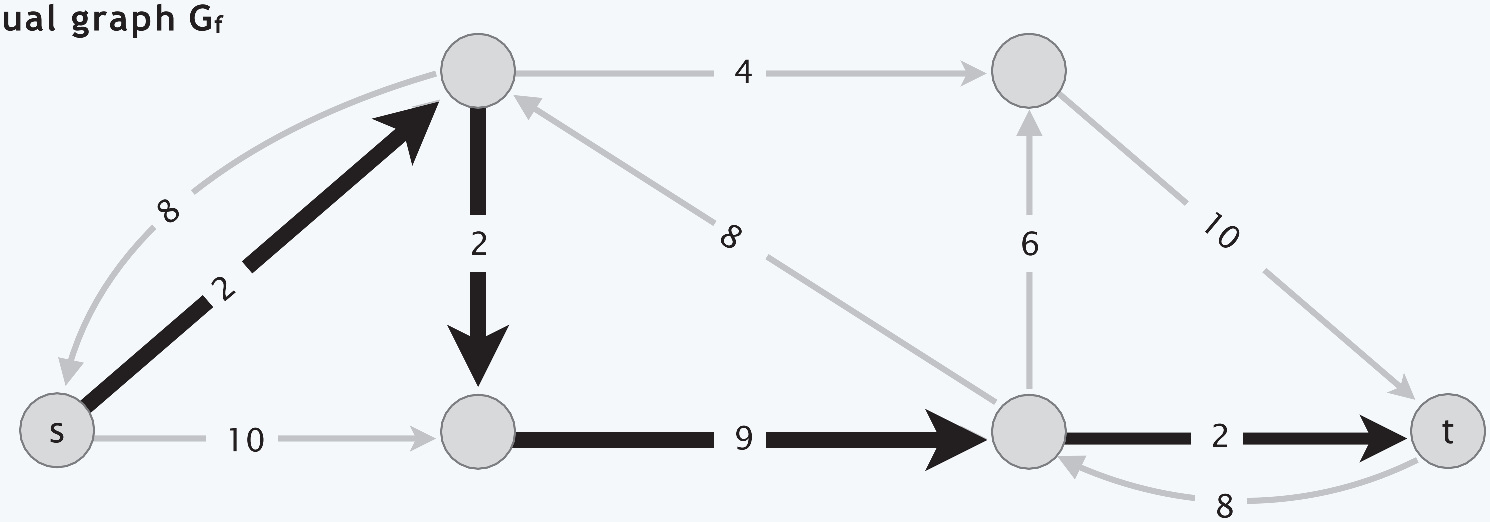


Ford-Fulkerson algorithm demo

network G

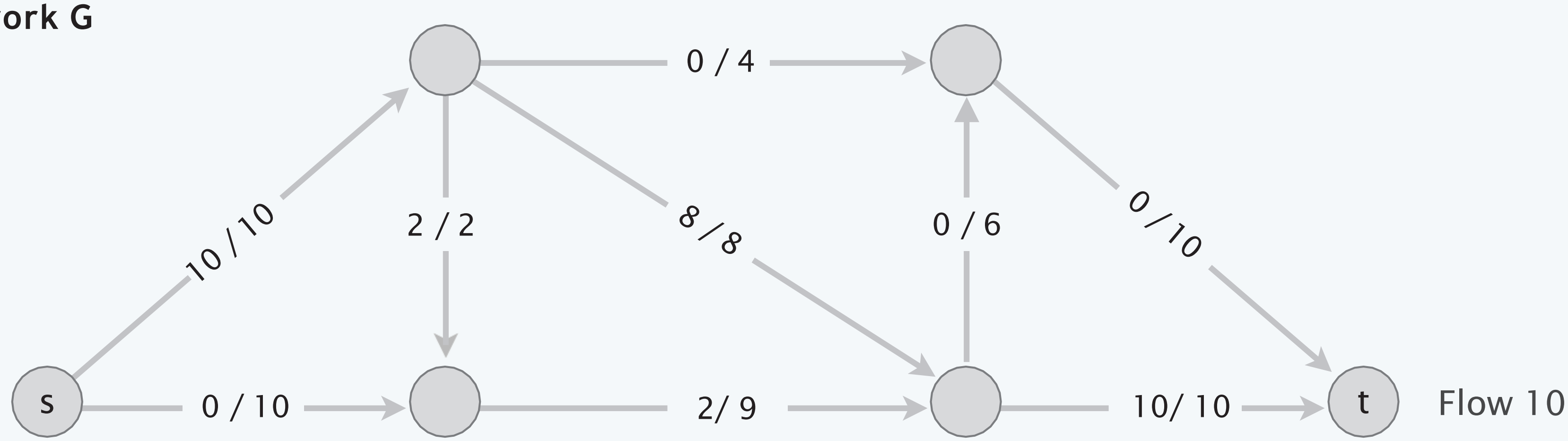


residual graph G_f

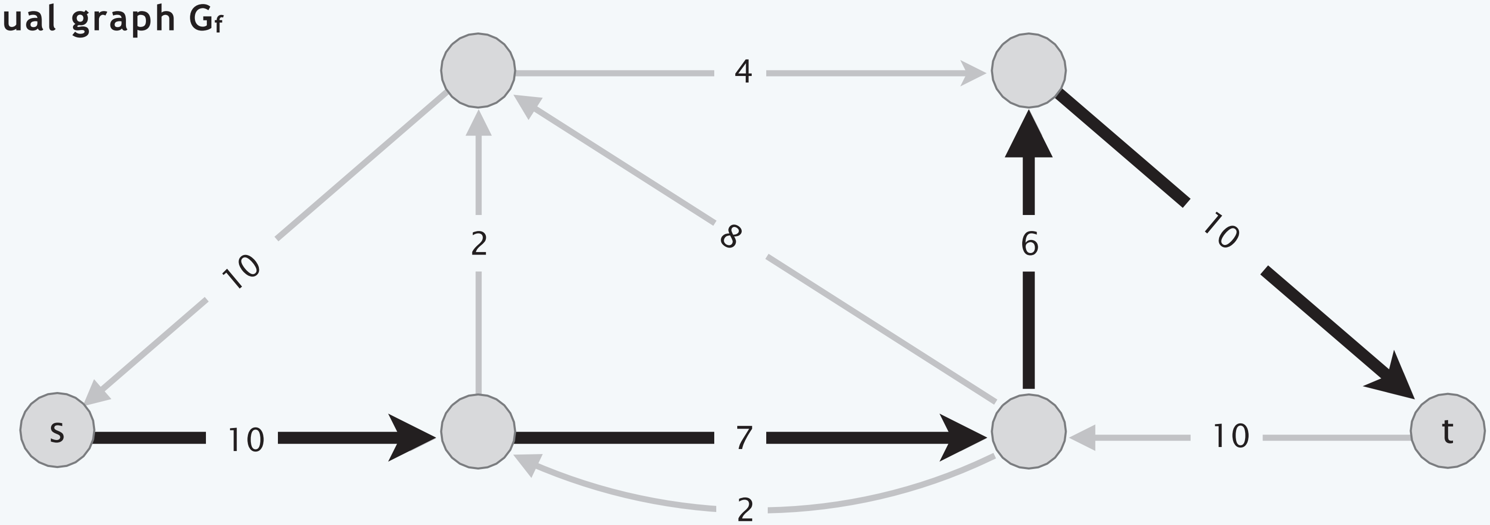


Ford-Fulkerson algorithm demo

network G

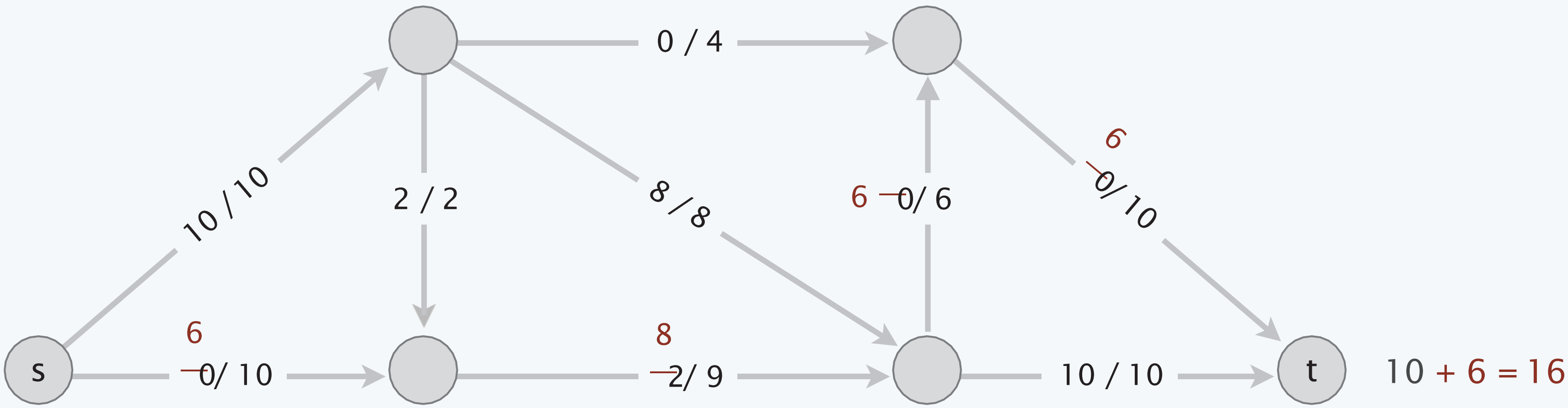


residual graph G_f

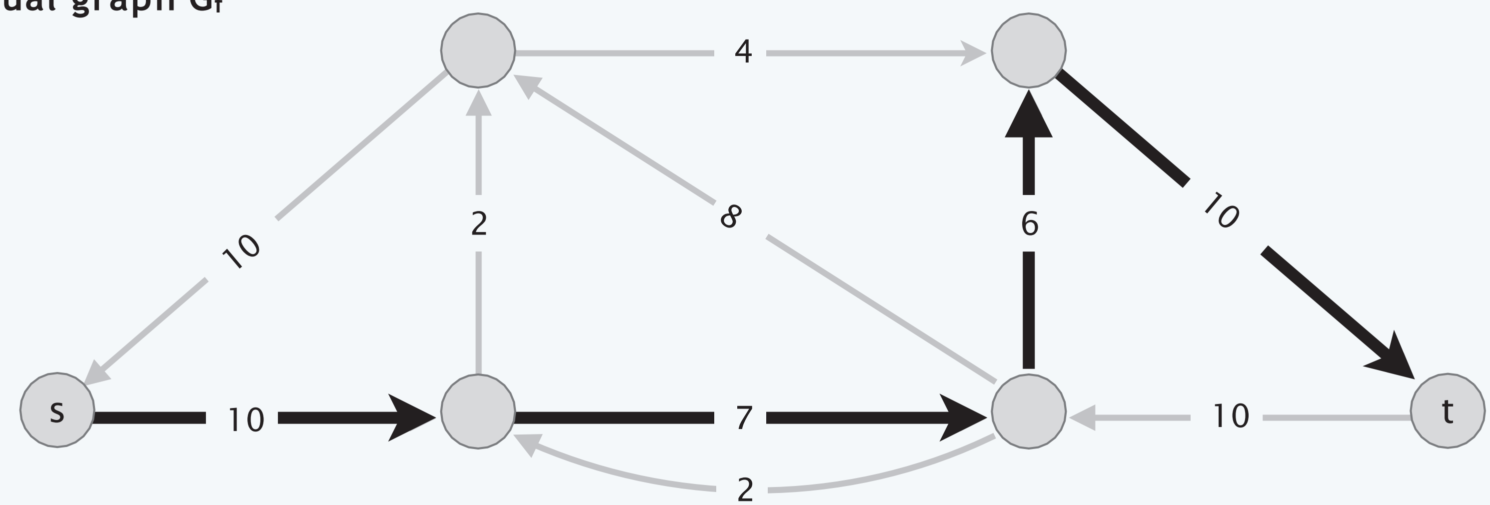


Ford-Fulkerson algorithm demo

network G

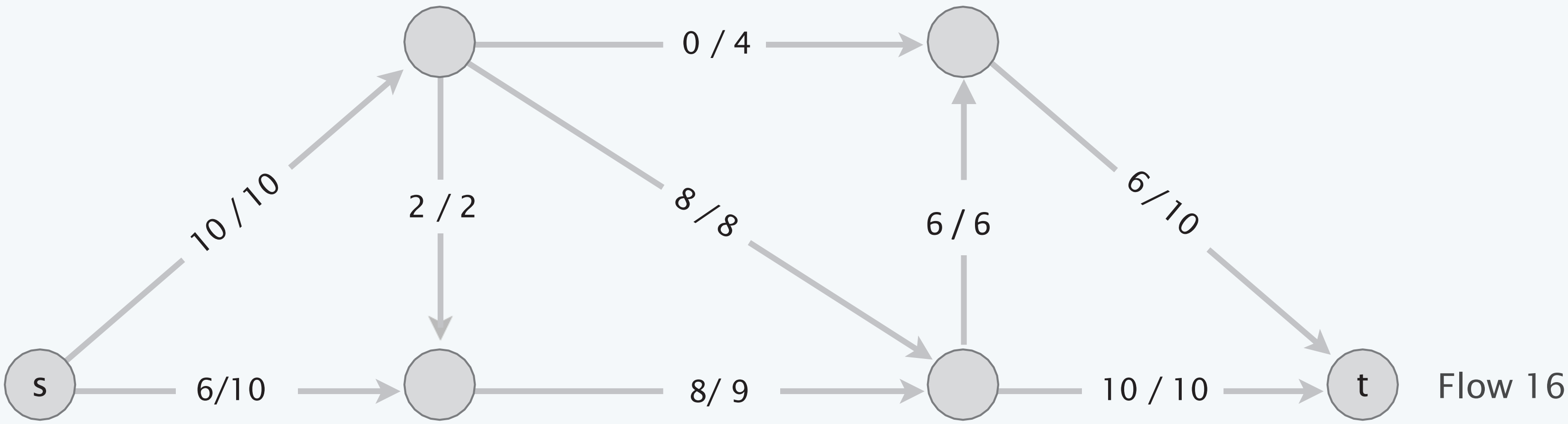


residual graph G_f

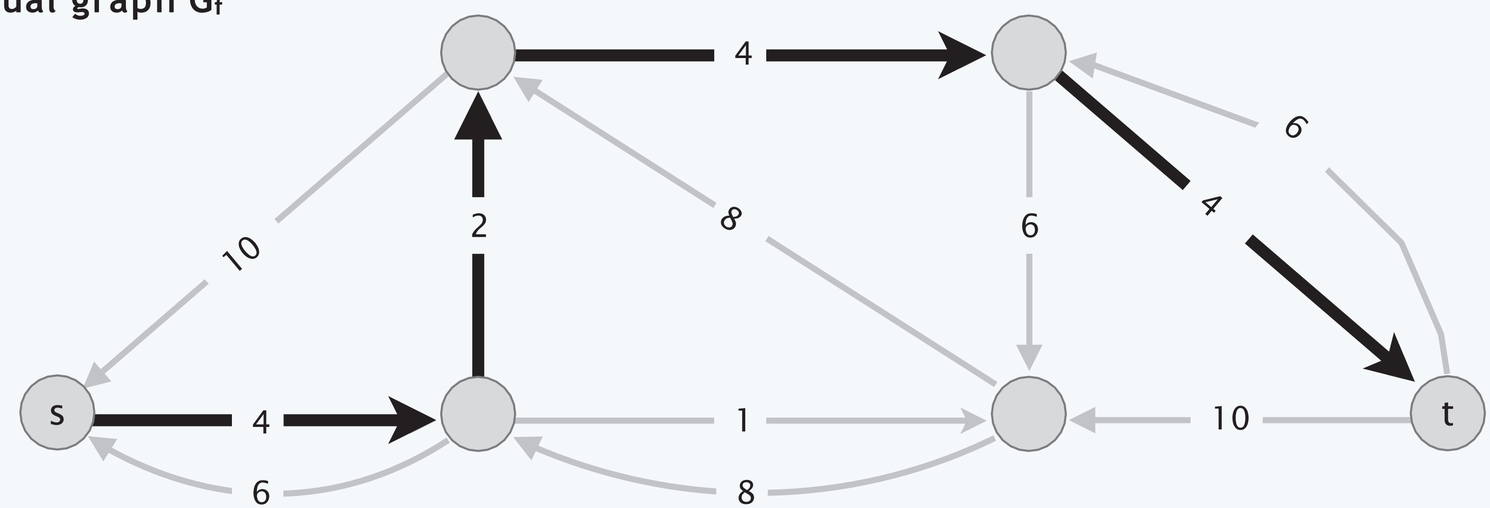


Ford-Fulkerson algorithm demo

network G

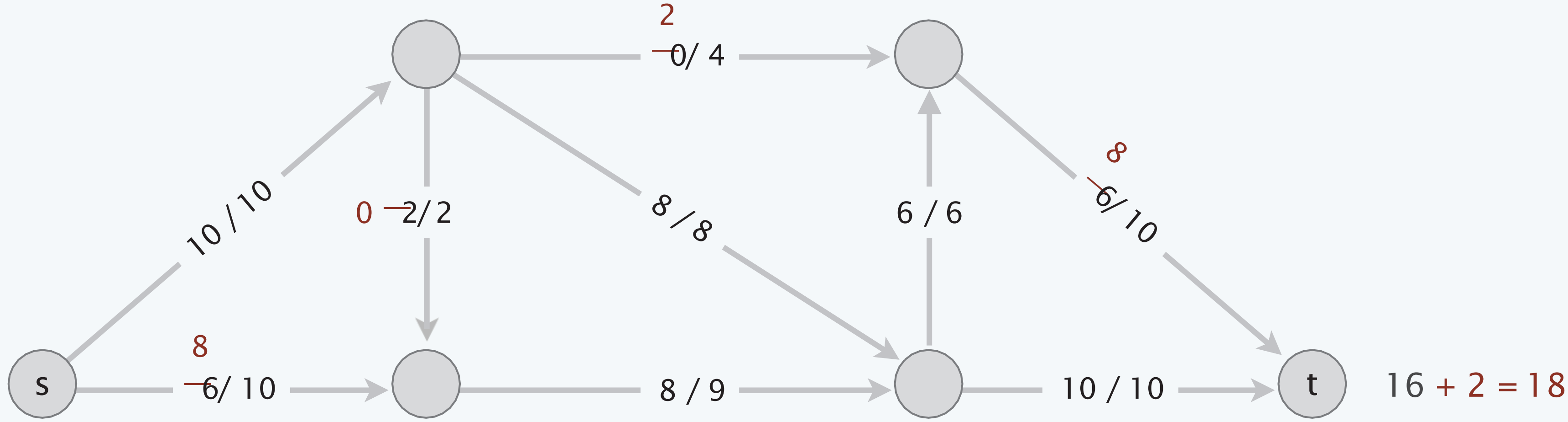


residual graph G_f

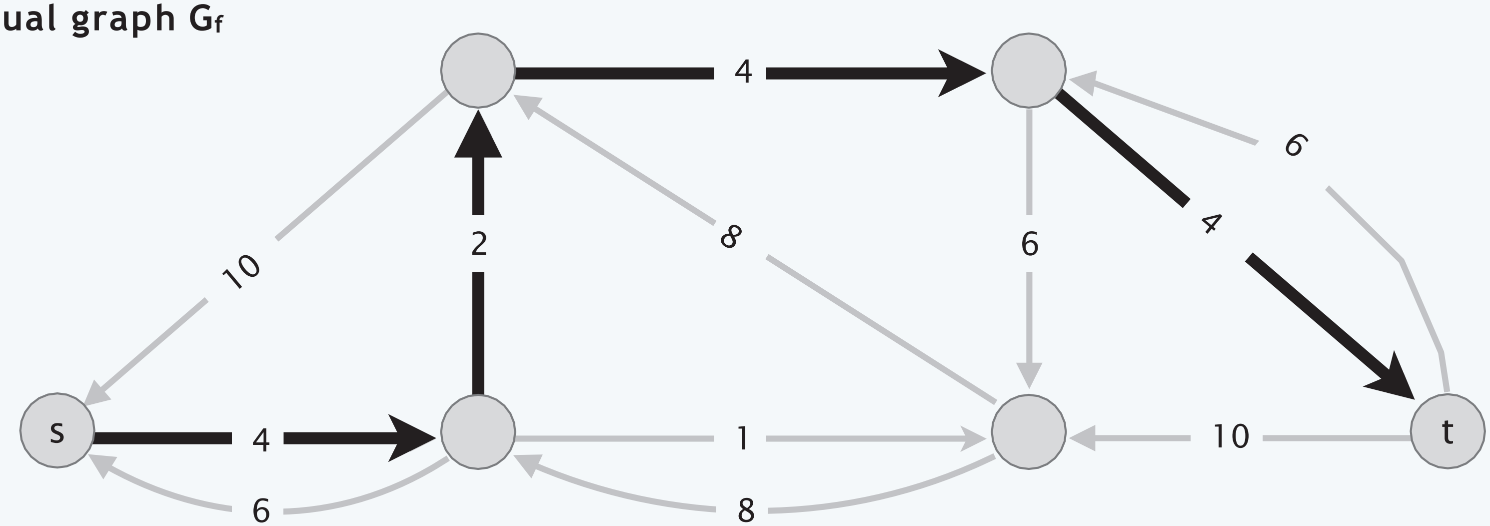


Ford-Fulkerson algorithm demo

network G

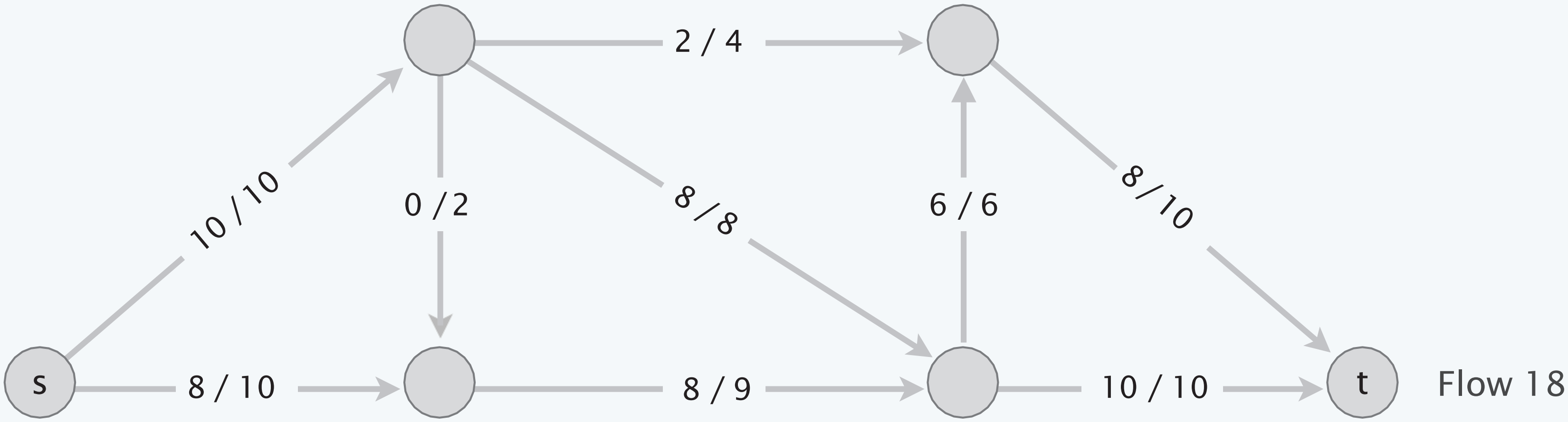


residual graph G_f

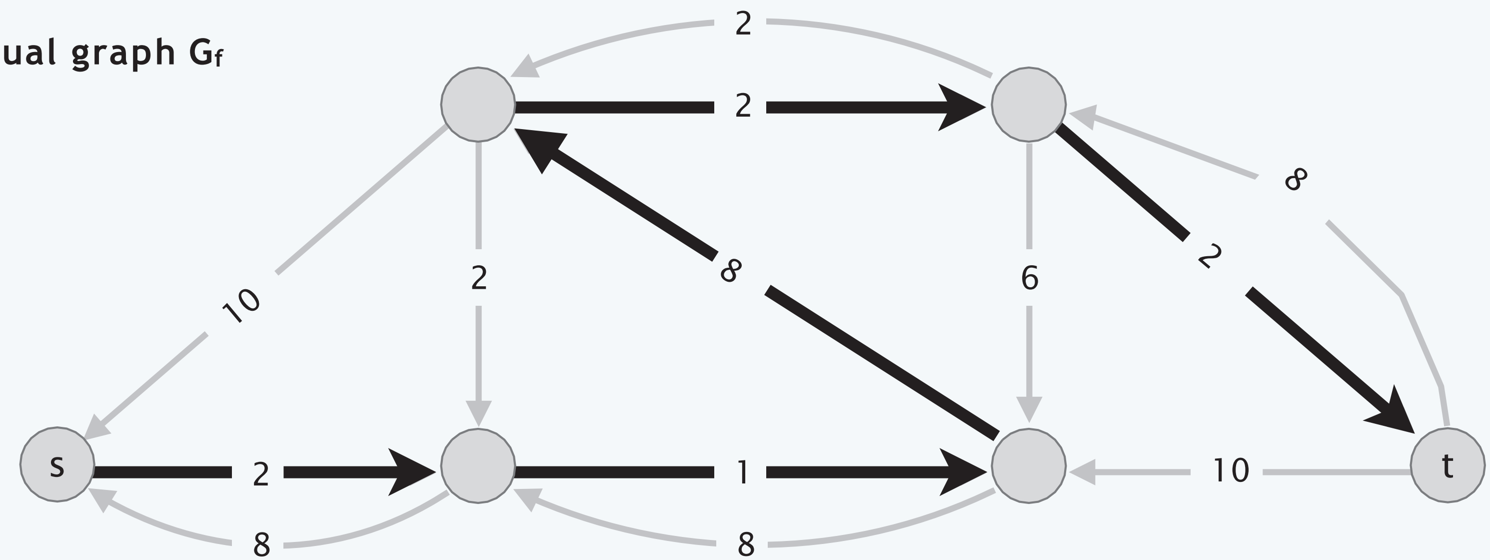


Ford-Fulkerson algorithm demo

network G

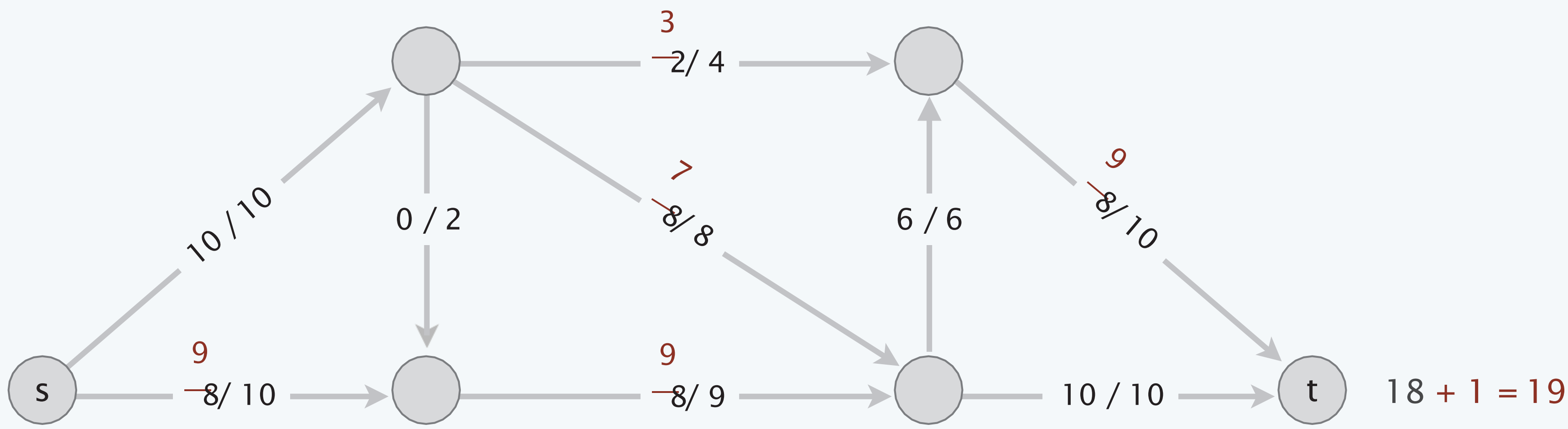


residual graph G_f



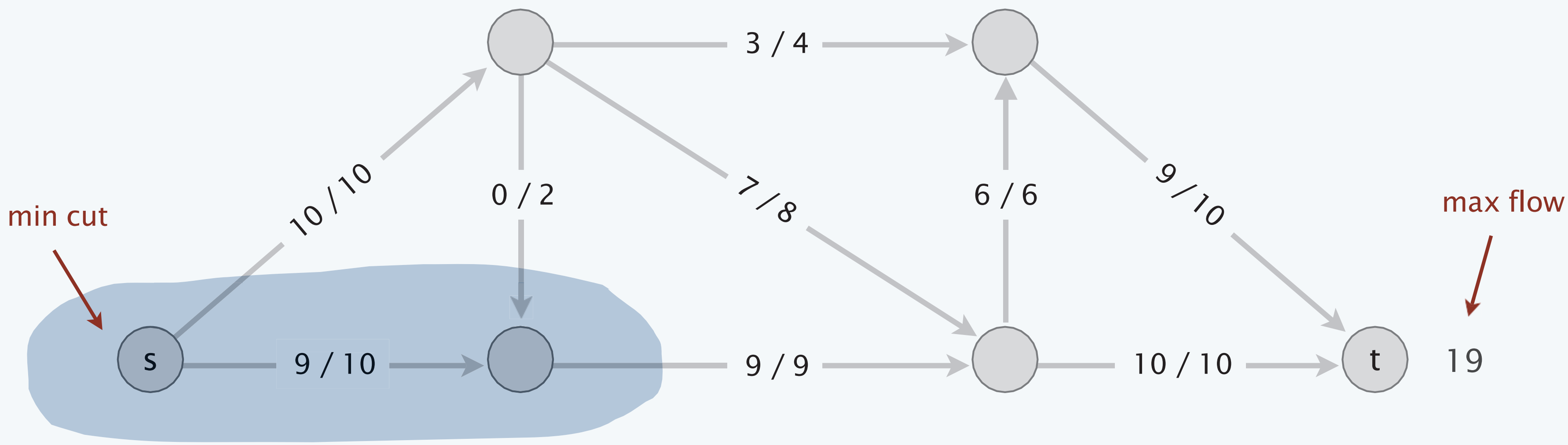
Ford-Fulkerson algorithm demo

network G

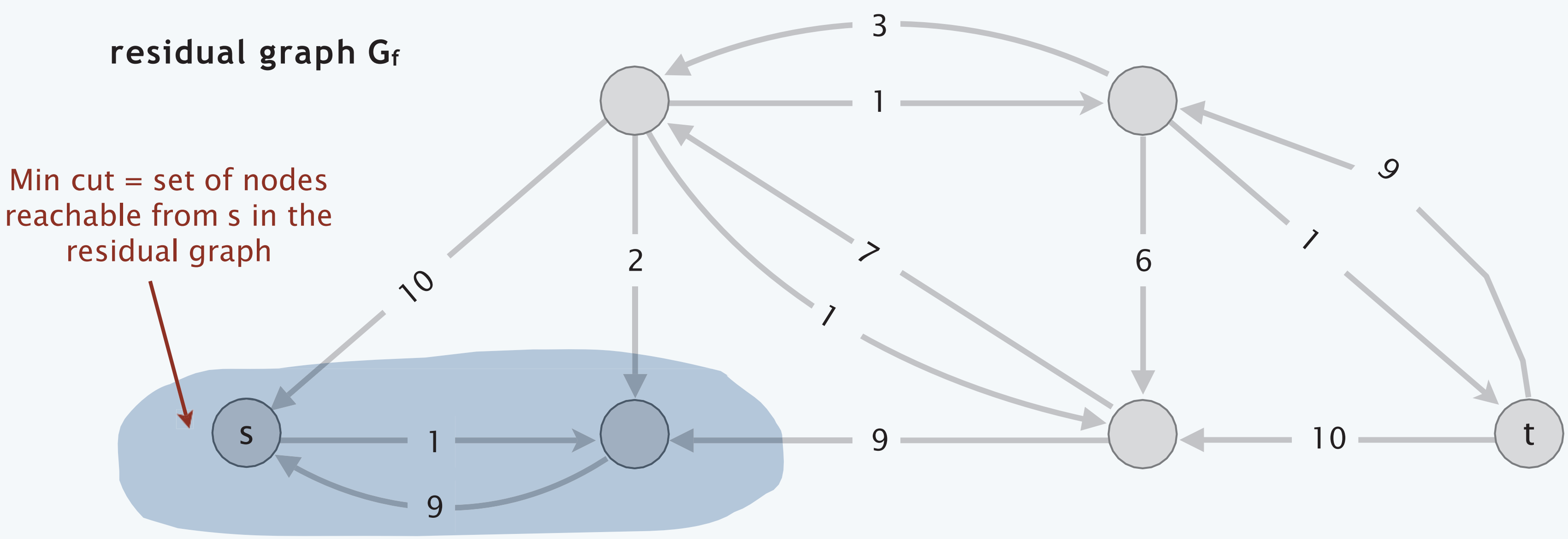


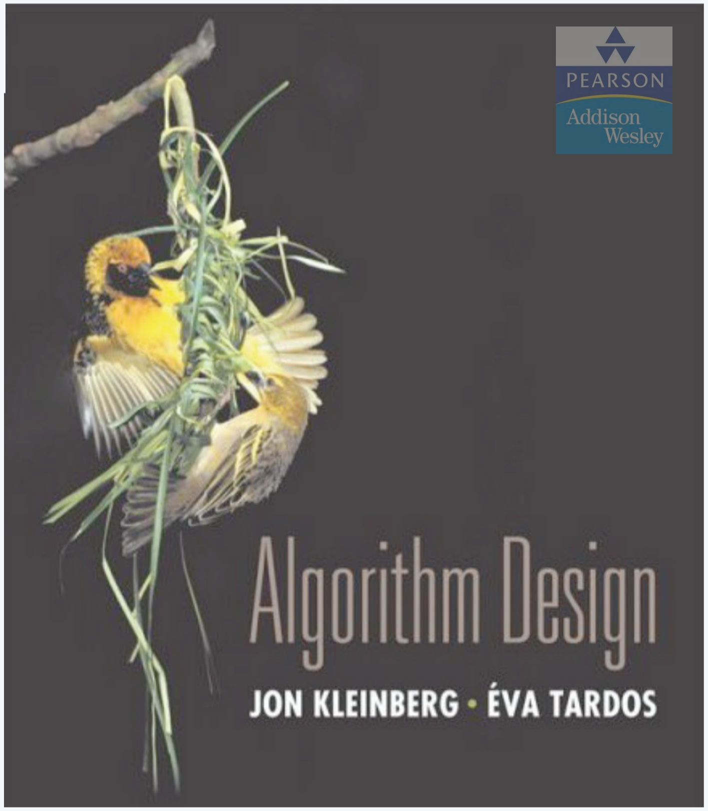
Ford-Fulkerson algorithm demo

network G



residual graph G_f





SECTION 7.2

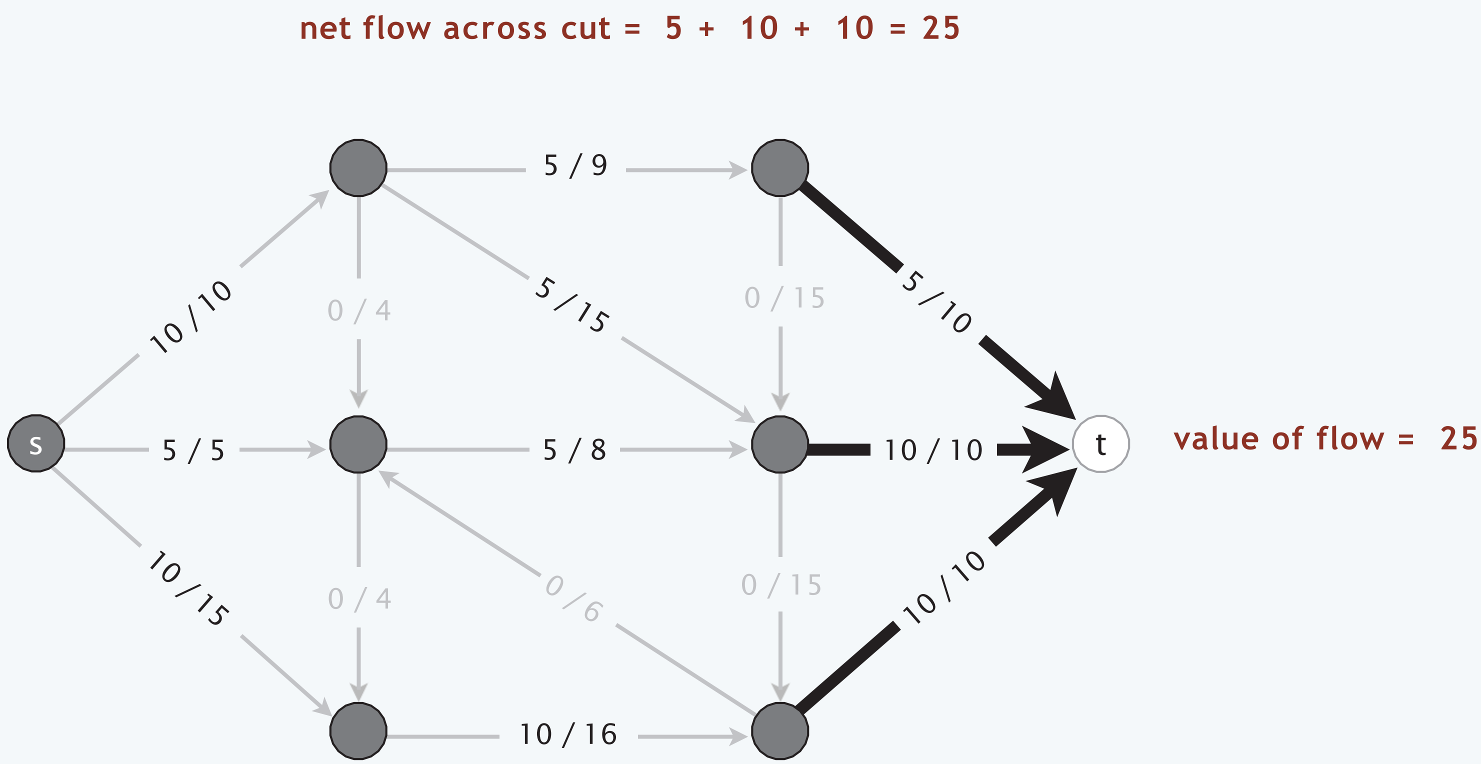
7. NETWORK FLOW I

- *max-flow and min-cut problems*
- *Ford-Fulkerson algorithm*
- *max-flow min-cut theorem*
- *capacity-scaling algorithm*
- *shortest augmenting paths*
- *blocking-flow algorithm*
- *unit-capacity simple networks*

Relationship between flows and cuts

Flow value lemma. Let f be any flow and let (A, B) be any cut. Then, the net flow across (A, B) equals the value of f.

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

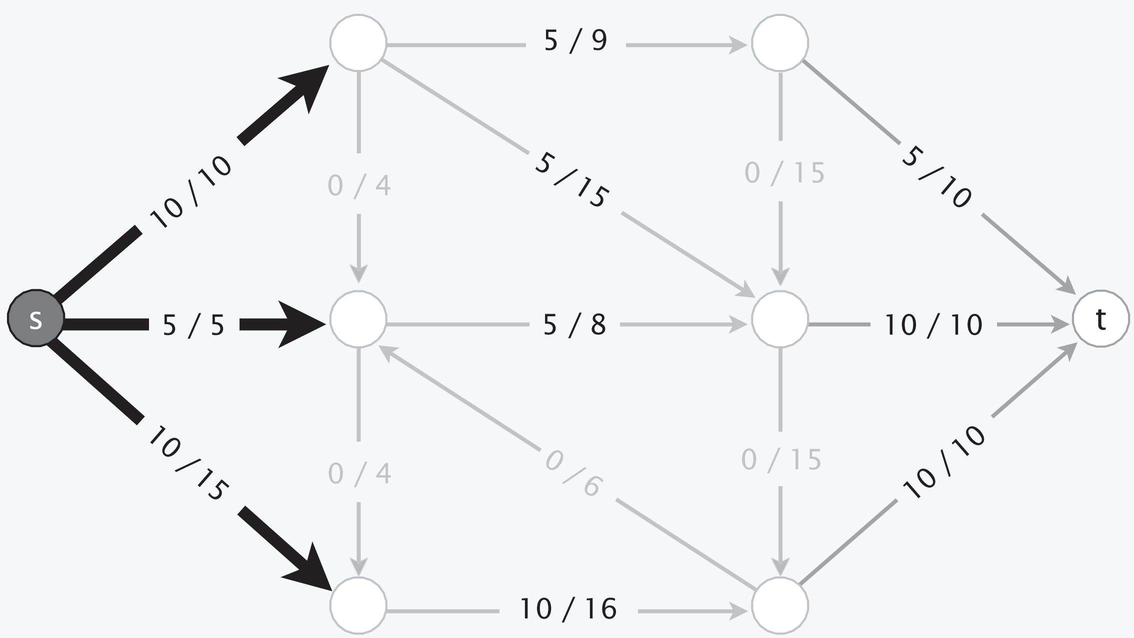


Relationship between flows and cuts

Flow value lemma. Let f be any flow and let (A, B) be any cut. Then, the net flow across (A, B) equals the value of f.

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

net flow across cut = 10 + 5 + 10 = 25



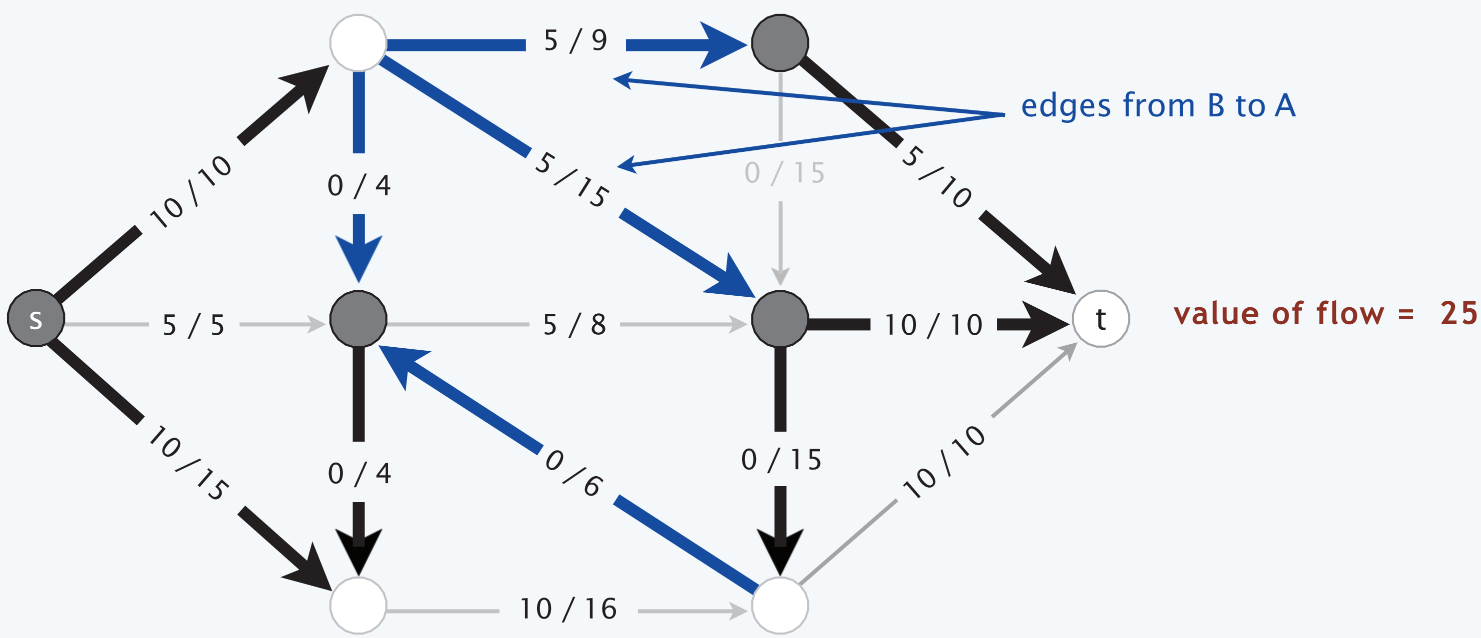
value of flow = 25

Relationship between flows and cuts

Flow value lemma. Let f be any flow and let (A, B) be any cut. Then, the net flow across (A, B) equals the value of f.

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

net flow across cut = (10 + 10 + 5 + 10 + 0 + 0) - (5 + 5 + 0 + 0) = 25



Relationship between flows and cuts

Flow value lemma. Let f be any flow and let (A, B) be any cut. Then, the net flow across (A, B) equals the value of f.

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

Pf.

$$
\begin{aligned}
val(f) &= \sum_{e \text{ out of } s} f(e) \;-\; \sum_{e \text{ in to } s} f(e) \\
&= \sum_{v \in A} \left(\sum_{e \text{ out of } v} f(e) \;-\; \sum_{e \text{ in to } v} f(e) \right) \\
&= \sum_{e \text{ out of } A} f(e) \;-\; \sum_{e \text{ in to } A} f(e) \;\blacksquare
\end{aligned}
$$

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

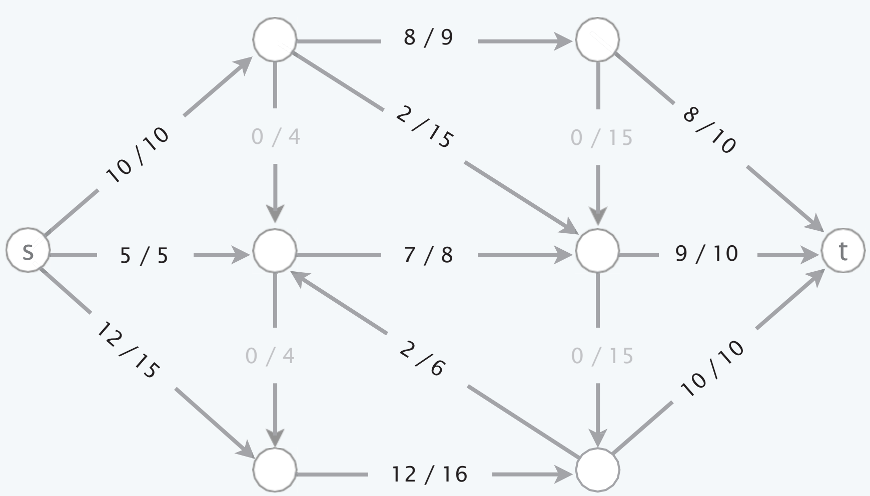
Relationship between flows and cuts

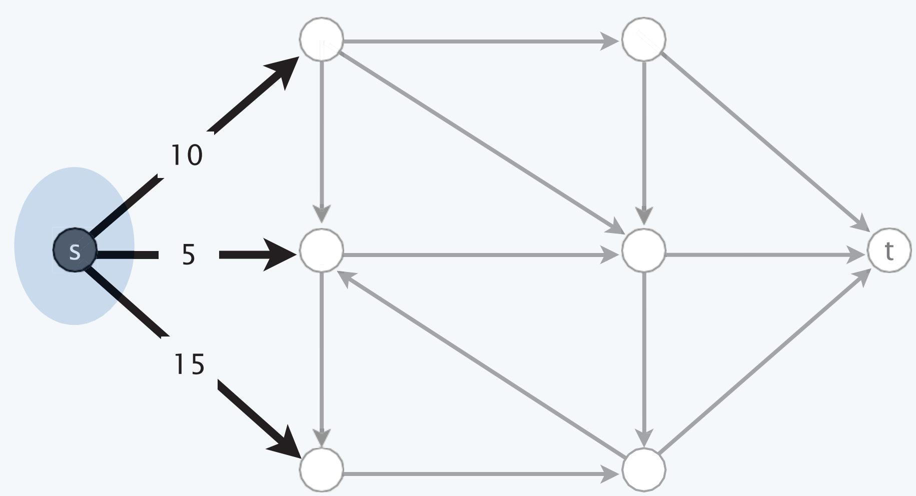
Weak duality. Let f be any flow and (A, B) be any cut. Then, $v(f) \leq cap(A, B)$.

Pf.

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

flow-value lemma





value of flow = 27 ≤ capacity of cut = 30

Max-flow min-cut theorem

Augmenting path theorem. A flow f is a max-flow iff no augmenting paths.

Max-flow min-cut theorem. Value of the max-flow = capacity of min-cut.

Pf. The following three conditions are equivalent for any flow f:

i. There exists a cut (A, B) such that $cap(A, B) = val(f)$.

ii. f is a max-flow.

iii. There is no augmenting path with respect to f.

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

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

Def. Its capacity is the sum of the capacities of the edges from A to B.

Recall:

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

Weak duality. Let f be any flow and (A, B) be any cut. Then, $v(f) \leq cap(A, B)$.

Flow value lemma. Let f be any flow and let (A, B) be any cut. Then, the net flow across (A, B) equals the value of f.

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

Max-flow min-cut theorem

Augmenting path theorem. A flow f is a max-flow iff no augmenting paths.

Max-flow min-cut theorem. Value of the max-flow = capacity of min-cut.

Pf. The following three conditions are equivalent for any flow f:

i. There exists a cut (A, B) such that $cap(A, B) = val(f)$.

ii. f is a max-flow.

iii. There is no augmenting path with respect to f.

[i ⇒ ii]

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

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

Def. Its capacity is the sum of the capacities of the edges from A to B.

Recall:

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

Weak duality. Let f be any flow and (A, B) be any cut. Then, $v(f) \leq cap(A, B)$.

Flow value lemma. Let f be any flow and let (A, B) be any cut. Then, the net flow across (A, B) equals the value of f.

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

Max-flow min-cut theorem

Augmenting path theorem. A flow f is a max-flow iff no augmenting paths.

Max-flow min-cut theorem. Value of the max-flow = capacity of min-cut.

Pf. The following three conditions are equivalent for any flow f:

i. There exists a cut (A, B) such that $cap(A, B) = val(f)$.

ii. f is a max-flow.

iii. There is no augmenting path with respect to f.

[i ⇒ ii]

- Suppose that (A, B) is a cut such that $cap(A, B) = val(f)$.
- Then, for any flow f', $\underbrace{val(f') \leq cap(A, B)}_{\text{weak duality}} \underbrace{=}_{\text{by assumption}} val(f)$.
- Thus, f is a max-flow. ▪

Max-flow min-cut theorem

Augmenting path theorem. A flow f is a max-flow iff no augmenting paths.

Max-flow min-cut theorem. Value of the max-flow = capacity of min-cut.

Pf. The following three conditions are equivalent for any flow f:

i. There exists a cut (A, B) such that $cap(A, B) = val(f)$.

ii. f is a max-flow.

iii. There is no augmenting path with respect to f.

[ii ⇒ iii]

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

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

Def. Its capacity is the sum of the capacities of the edges from A to B.

Recall:

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

Weak duality. Let f be any flow and (A, B) be any cut. Then, $v(f) \leq cap(A, B)$.

Flow value lemma. Let f be any flow and let (A, B) be any cut. Then, the net flow across (A, B) equals the value of f.

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

Max-flow min-cut theorem

Augmenting path theorem. A flow f is a max-flow iff no augmenting paths.

Max-flow min-cut theorem. Value of the max-flow = capacity of min-cut.

Pf. The following three conditions are equivalent for any flow f:

i. There exists a cut (A, B) such that $cap(A, B) = val(f)$.

ii. f is a max-flow.

iii. There is no augmenting path with respect to f.

[ii ⇒ iii] We prove contrapositive: ~iii ⇒ ~ii.

- Suppose that there is an augmenting path with respect to f.
- Can improve flow f by sending flow along this path.
- Thus, f is not a max-flow. ▪

Max-flow min-cut theorem

Augmenting path theorem. A flow f is a max-flow iff no augmenting paths.

Max-flow min-cut theorem. Value of the max-flow = capacity of min-cut.

Pf. The following three conditions are equivalent for any flow f:

i. There exists a cut (A, B) such that $cap(A, B) = val(f)$.

ii. f is a max-flow.

iii. There is no augmenting path with respect to f.

[iii $\Rightarrow$ i]

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

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

Def. Its capacity is the sum of the capacities of the edges from A to B.

Recall:

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

Weak duality. Let f be any flow and (A, B) be any cut. Then, $v(f) \leq cap(A, B)$.

Flow value lemma. Let f be any flow and let (A, B) be any cut. Then, the net flow across (A, B) equals the value of f.

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

Max-flow min-cut theorem

[iii ⇒ i]

- Let f be a flow with no augmenting paths.
- Let A be set of nodes reachable from s in residual graph G_f.
- By definition of cut A, $s \in A$.
- By definition of flow f, $t \notin A$.

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

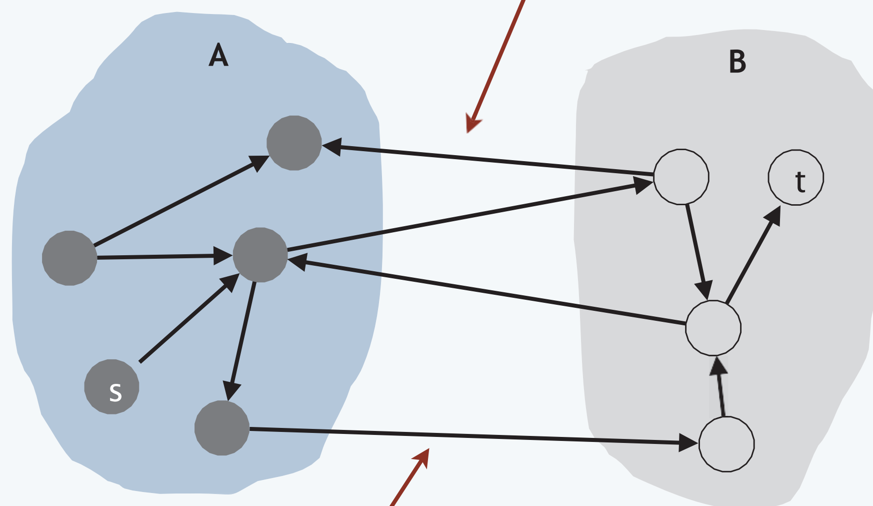
flow-value lemma

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

$$= cap(A, B) \quad \blacksquare$$

edge e = (v, w) with v ∈B, w ∈A
must have f(e) = 0

original network G



edge e = (v, w) with v ∈A, w ∈B
must have f(e) = c(e)