Learning Goals

- Definition of cuts and their capacities
- Cut capacities are upper bounds on flow values
- Max Flow Min Cut Theorem and its proof
- Correctness of Ford-Fulkerson
- Ford-Fulkerson as an algorithm to find min cuts
- Properties of Ford-Fulkerson

Proof of Correctness for Ford-Fulkerson

We already know:

- After each augmentation, we have a new flow with more value.
- For integral capacities, the algorithm terminates after at most C rounds, where $C = \sum_{e \in \delta_{\text{out}}(s)} c_e$.

Proof of Correctness for Ford-Fulkerson

We already know:

- After each augmentation, we have a new flow with more value.
- For integral capacities, the algorithm terminates after at most C rounds, where $C = \sum_{e \in \delta_{\text{out}}(s)} c_e$.
- Remains to show: the flow returned is maximum.

Proof of Correctness for Ford-Fulkerson

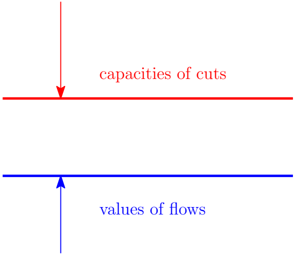
We already know:

- After each augmentation, we have a new flow with more value.
- For integral capacities, the algorithm terminates after at most C rounds, where $C = \sum_{e \in \delta_{\text{out}}(s)} c_e$.
- Remains to show: the flow returned is maximum.
- Proof strategy: show that the value of the flow returned is equal to a quantity which is an upper bound on the value of any flow.

Proof of Correctness for Ford-Fulkerson

We already know:

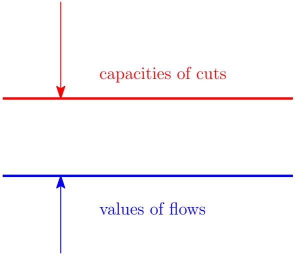
- After each augmentation, we have a new flow with more value.
- For integral capacities, the algorithm terminates after at most C rounds, where $C = \sum_{e \in \delta_{\text{out}}(s)} c_e$.
- Remains to show: the flow returned is maximum.
- Proof strategy: show that the value of the flow returned is equal to a quantity which is an upper bound on the value of any flow.

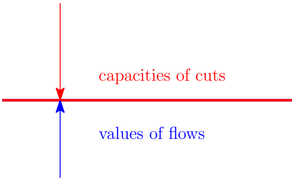


Proof of Correctness for Ford-Fulkerson

We already know:

- After each augmentation, we have a new flow with more value.
- For integral capacities, the algorithm terminates after at most C rounds, where $C = \sum_{e \in \delta_{\text{out}}(s)} c_e$.
- Remains to show: the flow returned is maximum.
- Proof strategy: show that the value of the flow returned is equal to a quantity which is an upper bound on the value of any flow.





Cuts

- Given a graph $G = (V, E)$, a *cut* (A, B) is a partition of V into two sets A and B, i.e., $A \cap B = \emptyset$, $A \cup B = V$.

Cuts

- Given a graph $G = (V, E)$, a *cut* (A, B) is a partition of V into two sets A and B, i.e., $A \cap B = \emptyset$, $A \cup B = V$.
- Given a flow network, an *s-t cut* is a cut (A, B) such that the source s is in A and the sink t is in B.

Cuts

- Given a graph $G = (V, E)$, a *cut* (A, B) is a partition of V into two sets A and B, i.e., $A \cap B = \emptyset$, $A \cup B = V$.
- Given a flow network, an *s-t cut* is a cut (A, B) such that the source s is in A and the sink t is in B.
- Notation: For a set of nodes $S \subseteq V$, let $\delta_{\text{out}}(S)$ denote the set of edges going out of S, and $\delta_{\text{in}}(S)$ the set of edges going into S.

Cuts

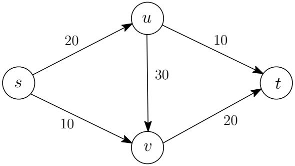
- Given a graph $G = (V, E)$, a *cut* (A, B) is a partition of V into two sets A and B, i.e., $A \cap B = \emptyset$, $A \cup B = V$.
- Given a flow network, an *s-t cut* is a cut (A, B) such that the source s is in A and the sink t is in B.
- Notation: For a set of nodes $S \subseteq V$, let $\delta_{\text{out}}(S)$ denote the set of edges going out of S, and $\delta_{\text{in}}(S)$ the set of edges going into S.
- The capacity of an s-t cut (A, B) is

$$c(A, B) := \sum_{e \in \delta_{\text{out}}(A)} c_e.$$

Cuts

- Given a graph $G = (V, E)$, a *cut* (A, B) is a partition of V into two sets A and B, i.e., $A \cap B = \emptyset$, $A \cup B = V$.
- Given a flow network, an *s-t cut* is a cut (A, B) such that the source s is in A and the sink t is in B.
- Notation: For a set of nodes $S \subseteq V$, let $\delta_{\text{out}}(S)$ denote the set of edges going out of S, and $\delta_{\text{in}}(S)$ the set of edges going into S.
- The capacity of an s-t cut (A, B) is

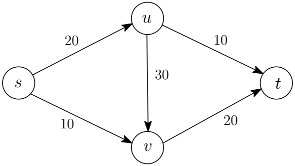
$$c(A, B) := \sum_{e \in \delta_{\text{out}}(A)} c_e.$$

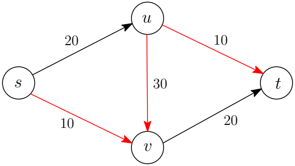


Cuts

- Given a graph $G = (V, E)$, a *cut* (A, B) is a partition of V into two sets A and B, i.e., $A \cap B = \emptyset$, $A \cup B = V$.
- Given a flow network, an *s-t cut* is a cut (A, B) such that the source s is in A and the sink t is in B.
- Notation: For a set of nodes $S \subseteq V$, let $\delta_{\text{out}}(S)$ denote the set of edges going out of S, and $\delta_{\text{in}}(S)$ the set of edges going into S.
- The capacity of an s-t cut (A, B) is

$$c(A, B) := \sum_{e \in \delta_{\text{out}}(A)} c_e.$$

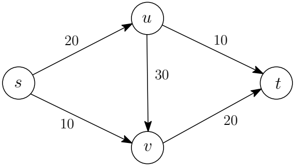


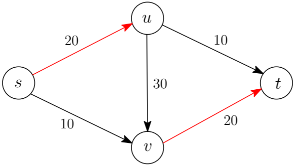


Cuts

- Given a graph $G = (V, E)$, a *cut* (A, B) is a partition of V into two sets A and B, i.e., $A \cap B = \emptyset$, $A \cup B = V$.
- Given a flow network, an *s-t cut* is a cut (A, B) such that s is in A and t is in B.
- Notation: For a set of nodes $S \subseteq V$, let $\delta_{\text{out}}(S)$ denote the set of edges going out of S, and $\delta_{\text{in}}(S)$ the set of edges going into S.
- The capacity of an s-t cut (A, B) is

$$c(A, B) := \sum_{e \in \delta_{\text{out}}(A)} c_e.$$





$c(\{s, v\}, \{u, t\}) = 40.$

Flow across cuts

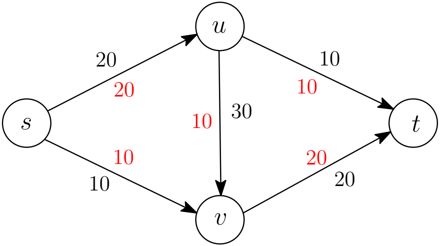
Given a flow f and an s-t cut (A, B), define

$$f^{\mathrm{out}}(A) := \sum_{e \in \delta_{\mathrm{out}}(A)} f(e), \quad f^{\mathrm{in}}(A) := \sum_{e \in \delta_{\mathrm{in}}(A)} f(e).$$

Flow across cuts

Given a flow f and an s-t cut (A, B), define

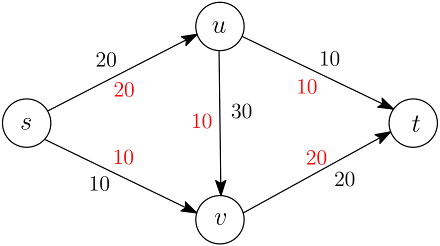
$$f^{\mathrm{out}}(A) := \sum_{e \in \delta_{\mathrm{out}}(A)} f(e), \quad f^{\mathrm{in}}(A) := \sum_{e \in \delta_{\mathrm{in}}(A)} f(e).$$



Flow across cuts

Given a flow f and an s-t cut (A, B), define

$$f^{\text{out}}(A) := \sum_{e \in \delta_{\text{out}}(A)} f(e), \quad f^{\text{in}}(A) := \sum_{e \in \delta_{\text{in}}(A)} f(e).$$

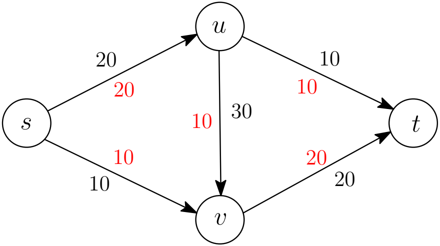


$$f^{\text{out}}(\{s, v\}) = 40;$$
$$f^{\text{in}}(\{s, v\}) = 10;$$

Flow across cuts

Given a flow f and an s-t cut (A, B), define

$$f^{\text{out}}(A) := \sum_{e \in \delta_{\text{out}}(A)} f(e), \quad f^{\text{in}}(A) := \sum_{e \in \delta_{\text{in}}(A)} f(e).$$



$$f^{\text{out}}(\{s, v\}) = 40;$$
$$f^{\text{in}}(\{s, v\}) = 10;$$

$$f^{\text{out}}(\{s, u\}) = 30;$$
$$f^{\text{in}}(\{s, u\}) = 0.$$

Relating Flows to Cuts

Lemma

For any s-t cut (A, B) and any flow f, the value of f, $|f|$, is $f^{out}(A) - f^{in}(A)$.

Relating Flows to Cuts

Lemma

For any s-t cut (A, B) *and any flow* f*, the value of* f*,* $|f|$*, is* $f^{out}(A) - f^{in}(A)$.

Corollary

For any s-t cut (A, B) *and any flow* f*,* $|f| \leq c(A, B)$*. If* $|f| = c(A, B)$*, then* $f^{\mathrm{in}}(A) = 0$*, and* $f(e) = c_e$ *for each* $e \in \delta_{\mathrm{out}}(A)$.

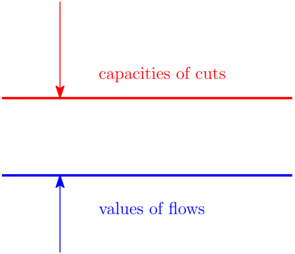
Relating Flows to Cuts

Lemma

For any s-t cut (A, B) and any flow f, the value of f, $|f|$, is $f^{out}(A) - f^{in}(A)$.

Corollary

For any s-t cut (A, B) and any flow f, $|f| \leq c(A, B)$. If $|f| = c(A, B)$, then $f^{\text{in}}(A) = 0$, and $f(e) = c_e$ for each $e \in \delta_{\text{out}}(A)$.



Relating Flows to Cuts

Lemma

For any s-t cut (A, B) and any flow f, the value of f, $|f|$, is $f^{out}(A) - f^{in}(A)$.

Corollary

For any s-t cut (A, B) and any flow f, $|f| \leq c(A, B)$. If $|f| = c(A, B)$, then $f^{\text{in}}(A) = 0$, and $f(e) = c_e$ for each $e \in \delta_{\text{out}}(A)$.

Proof of Corollary.

$$|f| = f^{\text{out}}(A) - f^{\text{in}}(A) \leq f^{\text{out}}(A) = \sum_{e \in \delta_{\text{out}}(A)} f(e) \leq \sum_{e \in \delta_{\text{out}}(A)} c_e = c(A, B).$$

Relating Flows to Cuts

Lemma

For any s-t cut (A, B) and any flow f, the value of f, $|f|$, is $f^{out}(A) - f^{in}(A)$.

Corollary

For any s-t cut (A, B) and any flow f, $|f| \leq c(A, B)$. If $|f| = c(A, B)$, then $f^{\text{in}}(A) = 0$, and $f(e) = c_e$ for each $e \in \delta_{\text{out}}(A)$.

Proof of Corollary.

$$|f| = f^{\text{out}}(A) - f^{\text{in}}(A) \leq f^{\text{out}}(A) = \sum_{e \in \delta_{\text{out}}(A)} f(e) \leq \sum_{e \in \delta_{\text{out}}(A)} c_e = c(A, B).$$

If $|f| = c(A, B)$, all inequalities above must be tight. □

Lemma

For any s-t cut (A, B) and any flow f, the value of f, $|f|$, is $f^{\text{out}}(A) - f^{\text{in}}(A)$.

Proof of Lemma.

By induction on $|A|$. IH: Lemma statement. Base case: $|A| = 1$, $A = \{s\}$, $f^{\text{in}}(A) = 0$, $f^{\text{out}}(A) = \sum_{e \in \delta_{\text{out}}(s)} f(e) = |f|$ by definition.

Lemma

For any s-t cut (A, B) *and any flow* f*, the value of* f*,* $|f|$*, is* $f^{\text{out}}(A) - f^{\text{in}}(A)$.

Proof of Lemma.

By induction on $|A|$. IH: Lemma statement. Base case: $|A| = 1$, $A = \{s\}$, $f^{\text{in}}(A) = 0$, $f^{\text{out}}(A) = \sum_{e \in \delta_{\text{out}}(s)} f(e) = |f|$ by definition.
Inductive step: Take $v \in A$, $v \neq s$. By IH,

$$|f| = f^{\text{out}}(A \setminus \{v\}) - f^{\text{in}}(A \setminus \{v\}).$$

Lemma

For any s-t cut (A, B) and any flow f, the value of f, $|f|$, is $f^{\text{out}}(A) - f^{\text{in}}(A)$.

Proof of Lemma.

By induction on $|A|$. IH: Lemma statement. Base case: $|A| = 1$, $A = \{s\}$, $f^{\text{in}}(A) = 0$, $f^{\text{out}}(A) = \sum_{e \in \delta_{\text{out}}(s)} f(e) = |f|$ by definition.
Inductive step: Take $v \in A$, $v \neq s$. By IH,

$$|f| = f^{\text{out}}(A \setminus \{v\}) - f^{\text{in}}(A \setminus \{v\}).$$

$$f^{\text{out}}(A) = f^{\text{out}}(A \setminus \{v\}) + \sum_{e \text{ from } v \text{ to } \overline{A}} f(e) - \sum_{e \text{ from } A \setminus \{v\} \text{ to } v} f(e);$$

Lemma

For any s-t cut (A, B) and any flow f, the value of f, $|f|$, is $f^{\text{out}}(A) - f^{\text{in}}(A)$.

Proof of Lemma.

By induction on $|A|$. IH: Lemma statement. Base case: $|A| = 1$, $A = \{s\}$, $f^{\text{in}}(A) = 0$, $f^{\text{out}}(A) = \sum_{e \in \delta_{\text{out}}(s)} f(e) = |f|$ by definition.
Inductive step: Take $v \in A$, $v \neq s$. By IH,

$$|f| = f^{\text{out}}(A \setminus \{v\}) - f^{\text{in}}(A \setminus \{v\}).$$

$$f^{\text{out}}(A) = f^{\text{out}}(A \setminus \{v\}) + \sum_{e \text{ from } v \text{ to } \overline{A}} f(e) - \sum_{e \text{ from } A \setminus \{v\} \text{ to } v} f(e);$$

$$f^{\text{in}}(A) = f^{\text{in}}(A \setminus \{v\}) + \sum_{e \text{ from } \overline{A} \text{ to } v} f(e) - \sum_{e \text{ from } v \text{ to } A \setminus \{v\}} f(e).$$

Lemma

For any s-t cut (A, B) *and any flow* f*, the value of* f*,* $|f|$*, is* $f^{\text{out}}(A) - f^{\text{in}}(A)$*.*

Proof of Lemma.

$$f^{\text{out}}(A) = f^{\text{out}}(A \setminus \{v\}) + \sum_{e \text{ from } v \text{ to } \overline{A}} f(e) - \sum_{e \text{ from } A\setminus\{v\} \text{ to } v} f(e);$$

$$f^{\text{in}}(A) = f^{\text{in}}(A \setminus \{v\}) + \sum_{e \text{ from } \overline{A} \text{ to } v} f(e) - \sum_{e \text{ from } v \text{ to } A\setminus\{v\}} f(e).$$

□

Lemma

For any s-t cut (A, B) and any flow f, the value of f, $|f|$, is $f^{\text{out}}(A) - f^{\text{in}}(A)$.

Proof of Lemma.

$$f^{\text{out}}(A) = f^{\text{out}}(A \setminus \{v\}) + \sum_{e \text{ from } v \text{ to } \overline{A}} f(e) - \sum_{e \text{ from } A \setminus \{v\} \text{ to } v} f(e);$$

$$f^{\text{in}}(A) = f^{\text{in}}(A \setminus \{v\}) + \sum_{e \text{ from } \overline{A} \text{ to } v} f(e) - \sum_{e \text{ from } v \text{ to } A \setminus \{v\}} f(e).$$

$$f^{\text{out}}(A) - f^{\text{in}}(A) = f^{\text{out}}(A \setminus \{v\}) - f^{\text{in}}(A \setminus \{v\}) + \left(\sum_{e \in \delta_{\text{out}}(v)} f(e) - \sum_{e \in \delta_{\text{in}}(v)} f(e) \right) = |f|.$$

□

The Max Flow Min Cut

Theorem (Max-Flow Min-Cut)

The following statements are equivalent:

1. f *is a maximum flow on a flow network* G;
2. *There is an s-t cut* (A, B) *with* $c(A, B) = |f|$;
3. *There exists no augmenting path in the residual graph* G_f.

The Max Flow Min Cut

Theorem (Max-Flow Min-Cut)

The following statements are equivalent:

1. f *is a maximum flow on a flow network* G;
2. *There is an* s-t *cut* (A, B) *with* $c(A, B) = |f|$;
3. *There exists no augmenting path in the residual graph* G_f.

Corollary

When the Ford-Fulkerson algorithm terminates, the flow it returns is a maximum flow.

Proof of Max Flow Min Cut Theorem

Theorem (Max-Flow Min-Cut Theorem)

The following statements are equivalent:

1. *f is a maximum flow on a flow network G;*
2. *There is an s-t cut (A, B) with $c(A, B) = |f|$;*
3. *There exists no augmenting path in the residual graph G_f.*

Proof.

$2 \Rightarrow 1$: For any other flow f', by previous corollary, $|f'| \leq c(A, B) = |f|$.

Proof of Max Flow Min Cut Theorem

Theorem (Max-Flow Min-Cut Theorem)

The following statements are equivalent:

1. *f is a maximum flow on a flow network G;*
2. *There is an s-t cut (A, B) with $c(A, B) = |f|$;*
3. *There exists no augmenting path in the residual graph G_f.*

Proof.

$2 \Rightarrow 1$: For any other flow f', by previous corollary, $|f'| \leq c(A, B) = |f|$.
$1 \Rightarrow 3$: If there were an augmenting path P, augmenting along P gives rise to another flow f' with $|f'| > |f|$, contradicting f's maximality.

Proof of Max Flow Min Cut Theorem

Theorem (Max-Flow Min-Cut Theorem)

The following statements are equivalent:

1. *f is a maximum flow on a flow network G;*
2. *There is an s-t cut (A, B) with $c(A, B) = |f|$;*
3. *There exists no augmenting path in the residual graph G_f.*

Proof.

$2 \Rightarrow 1$: For any other flow f', by previous corollary, $|f'| \leq c(A, B) = |f|$.
$1 \Rightarrow 3$: If there were an augmenting path P, augmenting along P gives rise to another flow f' with $|f'| > |f|$, contradicting f's maximality.
$3 \Rightarrow 2$: Let S be the set of nodes reachable from s in G_f.
Claim: $c(S, \overline{S}) = |f|$.

Proof of Theorem ($3 \Rightarrow 2$)

Claim (Restatement)

If for a flow f, there is no augmenting path in the residual graph G_f, let S be the set of nodes reachable from s in G_f, then $c(S, \overline{S}) = |f|$.

Proof of Theorem ($3 \Rightarrow 2$)

Claim (Restatement)

If for a flow f, there is no augmenting path in the residual graph G_f, let S be the set of nodes reachable from s in G_f, then $c(S, \overline{S}) = |f|$.

Proof.

There can be no edge from S to $\overline{S}$ in G_f, by definition of S. Hence:

Proof of Theorem ($3 \Rightarrow 2$)

Claim (Restatement)

If for a flow f, there is no augmenting path in the residual graph G_f, let S be the set of nodes reachable from s in G_f, then $c(S, \overline{S}) = |f|$.

Proof.

There can be no edge from S to $\overline{S}$ in G_f, by definition of S. Hence:

1. For every edge e in G going from S to $\overline{S}$, $f(e) = c_e$, otherwise there would be a forward edge from S to $\overline{S}$ in G_f;

Proof of Theorem ($3 \Rightarrow 2$)

Claim (Restatement)

If for a flow f, there is no augmenting path in the residual graph G_f, let S be the set of nodes reachable from s in G_f, then $c(S, \overline{S}) = |f|$.

Proof.

There can be no edge from S to $\overline{S}$ in G_f, by definition of S. Hence:

1. For every edge e in G going from S to $\overline{S}$, $f(e) = c_e$, otherwise there would be a forward edge from S to $\overline{S}$ in G_f;
2. For every edge e in G going from $\overline{S}$ to S, $f(e) = 0$, otherwise there would be a backward edge in G_f from S to $\overline{S}$.

Proof of Theorem ($3 \Rightarrow 2$)

Claim (Restatement)

If for a flow f, there is no augmenting path in the residual graph G_f, let S be the set of nodes reachable from s in G_f, then $c(S, \overline{S}) = |f|$.

Proof.

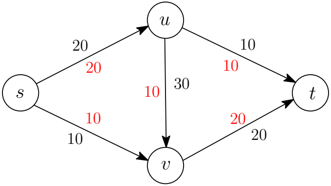
There can be no edge from S to $\overline{S}$ in G_f, by definition of S. Hence:

1. For every edge e in G going from S to $\overline{S}$, $f(e) = c_e$, otherwise there would be a forward edge from S to $\overline{S}$ in G_f;
2. For every edge e in G going from $\overline{S}$ to S, $f(e) = 0$, otherwise there would be a backward edge in G_f from S to $\overline{S}$.

Therefore $f^{\text{out}}(S) = \sum_{e \in \delta_{\text{out}}(S)} c_e = c(S, \overline{S})$, $f^{\text{in}}(S) = 0$, and $|f| = f^{\text{out}}(S) - f^{\text{in}}(S) = c(S, \overline{S})$.

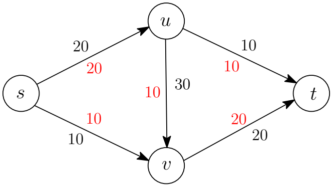
□

Illustration of 3 $\Rightarrow$ 2

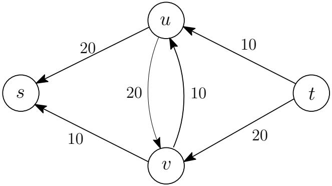


Max flow

Illustration of 3 $\Rightarrow$ 2

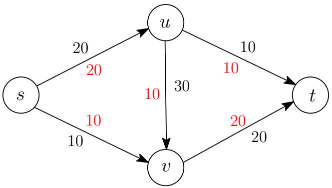


Max flow

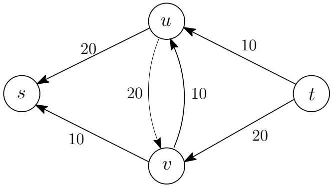


Residual graph

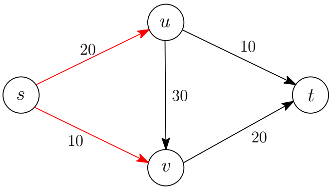
Illustration of $3 \Rightarrow 2$



Max flow

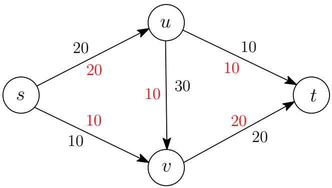


Residual graph

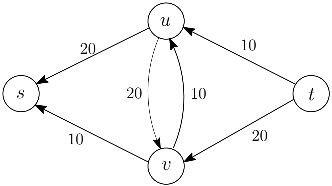


$S = \{s\}$.

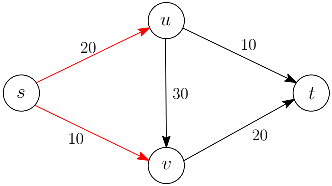
Illustration of 3 $\Rightarrow$ 2



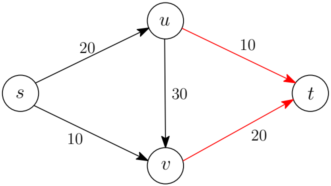
Max flow



Residual graph



$S = \{s\}$.



Another min cut.

The Min Cut Problem

Problem

- *Input: A flow network* $G = (V, E)$.
- *Output: An s-t cut* (A, B) *such that* $c(A, B)$ *is minimum among all s-t cuts.*

The Min Cut Problem

Problem

- *Input: A flow network* $G = (V, E)$*.*
- *Output: An s-t cut* (A, B) *such that* $c(A, B)$ *is minimum among all s-t cuts.*

- If a cut's capacity is equal to the value of a flow, its capacity must be minimum.

The Min Cut Problem

Problem

- *Input: A flow network* $G = (V, E)$.
- *Output: An s-t cut* (A, B) *such that* $c(A, B)$ *is minimum among all s-t cuts.*

- If a cut's capacity is equal to the value of a flow, its capacity must be minimum.
- Algorithm: Run a max flow algorithm and find a max flow f, build its residual graph G_f, let A be the set of nodes reachable from s in G_f, let B be $\overline{A}$.

The Min Cut Problem

Problem

- *Input: A flow network* $G = (V, E)$.
- *Output: An s-t cut* (A, B) *such that* $c(A, B)$ *is minimum among all s-t cuts.*

- If a cut's capacity is equal to the value of a flow, its capacity must be minimum.
- Algorithm: Run a max flow algorithm and find a max flow f, build its residual graph G_f, let A be the set of nodes reachable from s in G_f, let B be $\overline{A}$.
- Such a cut is called a *min cut*.

Last Remarks

- For flow networks with integer capacities, there is always an integer-valued maximum flow.

Last Remarks

- For flow networks with integer capacities, there is always an integer-valued maximum flow.
  - Such an integral flow can be found by the Ford-Fulkerson algorithm in time $O(Cm)$.

Last Remarks

- For flow networks with integer capacities, there is always an integer-valued maximum flow.
  - Such an integral flow can be found by the Ford-Fulkerson algorithm in time $O(Cm)$.
- For all flow networks (even where capacities are not integers), a maximum flow exists, and can be found in polynomial time. (More on this later.)

Last Remarks

- For flow networks with integer capacities, there is always an integer-valued maximum flow.
  - Such an integral flow can be found by the Ford-Fulkerson algorithm in time $O(Cm)$.
- For all flow networks (even where capacities are not integers), a maximum flow exists, and can be found in polynomial time. (More on this later.)
- Given a flow, verifying whether it is a max flow takes only $O(m)$ time.