Lecture 20: Max-Flow Min-Cut

Jessica Sorrell

November 6, 2025
601.433/633 Introduction to Algorithms
Slides by Michael Dinitz

Introduction

Flow Network:

- Directed graph $G = (V, E)$
- *Capacities* $c : E \to \mathbb{R}_{\geq 0}$ (simplify notation: $c(x, y) = 0$ if $(x, y) \notin E$)
- Source $s \in V$, sink $t \in V$

Today: flows and cuts

- Flow: "sending stuff" from s to t
- Cut: separating t from s

Turn out to be very related!

Today: some algorithms but not efficient. Mostly structure. Better algorithms Tuesday.

Flows

Intuition: send "stuff" from s to t

- Water in a city water system, traffic along roads, trains along tracks, ...

Flows

Intuition: send "stuff" from s to t

- Water in a city water system, traffic along roads, trains along tracks, ...

Definition

An *(s,t)-flow* is a function $f : E \to \mathbb{R}_{\geq 0}$ such that

$$\sum_{u:(u,v)\in E} f(u,v) = \sum_{u:(v,u)\in E} f(v,u)$$

for all $v \in V \setminus \{s,t\}$. This constraint is known as *flow conservation*.

Flows

Intuition: send "stuff" from s to t

- Water in a city water system, traffic along roads, trains along tracks, ...

Definition

An *(s,t)-flow* is a function $f : E \to \mathbb{R}_{\geq 0}$ such that

$$\sum_{u:(u,v)\in E} f(u,v) = \sum_{u:(v,u)\in E} f(v,u)$$

for all $v \in V \setminus \{s,t\}$. This constraint is known as *flow conservation*.

Value of flow $|f|$: "total amount of stuff sent from s to t"

Flows

Intuition: send "stuff" from s to t

- Water in a city water system, traffic along roads, trains along tracks, ...

Definition

An *(s,t)-flow* is a function $f : E \to \mathbb{R}_{\geq 0}$ such that

$$\sum_{u:(u,v)\in E} f(u,v) = \sum_{u:(v,u)\in E} f(v,u)$$

for all $v \in V \setminus \{s,t\}$. This constraint is known as *flow conservation*.

Value of flow $|f|$: "total amount of stuff sent from s to t"

$$|f| = \sum_{u:(s,u)\in E} f(s,u) - \sum_{u:(u,s)\in E} f(u,s) = \sum_{u:(u,t)\in E} f(u,t) - \sum_{u:(t,u)\in E} f(t,u)$$

Feasible Flows

Capacity constraints: $\boldsymbol{0 \leq f(u,v) \leq c(u,v)}$ for all $\boldsymbol{(u,v) \in V \times V}$

Feasible Flows

Capacity constraints: $0 \le f(u,v) \le c(u,v)$ for all $(u,v) \in V \times V$

Definitions:

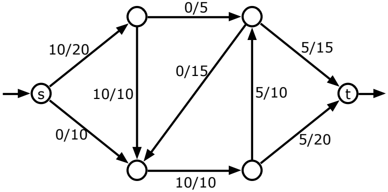
- An (s,t)-flow satisfying capacity constraints is a *feasible* flow.
- If $f(e) = c(e)$ then f *saturates* e.
- If $f(e) = 0$ then f *avoids* e.

Feasible Flows

Capacity constraints: $0 \le f(u,v) \le c(u,v)$ for all $(u,v) \in V \times V$

Definitions:

- An (s,t)-flow satisfying capacity constraints is a *feasible* flow.
- If $f(e) = c(e)$ then f *saturates* e.
- If $f(e) = 0$ then f *avoids* e.



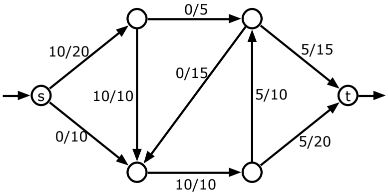
An (s,t)-flow with value 10. Each edge is labeled with its flow/capacity.

Feasible Flows

Capacity constraints: $\mathbf{0} \leq \boldsymbol{f}(\boldsymbol{u}, \boldsymbol{v}) \leq \boldsymbol{c}(\boldsymbol{u}, \boldsymbol{v})$ for all $(\boldsymbol{u}, \boldsymbol{v}) \in \boldsymbol{V} \times \boldsymbol{V}$

Definitions:

- An $(\boldsymbol{s}, \boldsymbol{t})$-flow satisfying capacity constraints is a *feasible* flow.
- If $\boldsymbol{f}(\boldsymbol{e}) = \boldsymbol{c}(\boldsymbol{e})$ then $\boldsymbol{f}$ *saturates* $\boldsymbol{e}$.
- If $\boldsymbol{f}(\boldsymbol{e}) = \mathbf{0}$ then $\boldsymbol{f}$ *avoids* $\boldsymbol{e}$.



An (s, t)-flow with value 10. Each edge is labeled with its flow/capacity.

Problem we'll talk about: find feasible flow of maximum value (max flow)

Cuts

Definition

- An (s,t)-*cut* is a partition of V into $(S, \bar{S})$ such that $s \in S$, $t \notin S$

Cuts

Definition

- An (s,t)*-cut* is a partition of V into $(S, \bar{S})$ such that $s \in S$, $t \notin S$
- The *capacity* of an (s,t)-cut $(S, \bar{S})$ is

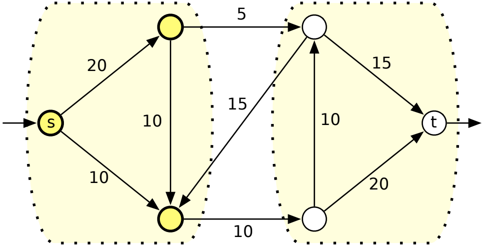
$$cap(S, \bar{S}) = \sum_{(u,v) \in E : u \in S, v \in \bar{S}} c(u,v) = \sum_{u \in S} \sum_{v \in \bar{S}} c(u,v)$$

Cuts

Definition

- An ***(s, t)-cut*** is a partition of V into $(S, \bar{S})$ such that $s \in S$, $t \notin S$
- The *capacity* of an (s, t)-cut $(S, \bar{S})$ is

$$cap(S, \bar{S}) = \sum_{(u,v) \in E : u \in S, v \in \bar{S}} c(u, v) = \sum_{u \in S} \sum_{v \in \bar{S}} c(u, v)$$

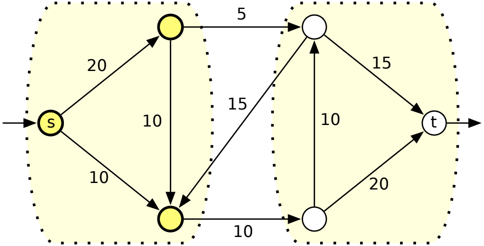


Cuts

Definition

- An ***(s, t)*-cut** is a partition of V into $(S, \bar{S})$ such that $s \in S$, $t \notin S$
- The *capacity* of an (s, t)-cut $(S, \bar{S})$ is

$$cap(S, \bar{S}) = \sum_{(u,v) \in E : u \in S, v \in \bar{S}} c(u, v) = \sum_{u \in S} \sum_{v \in \bar{S}} c(u, v)$$



Problem we'll talk about: find (s, t)-cut of minimum capacity (min cut)

Warmup Theorem

Theorem

Let f be a feasible (s, t)-flow, and let $(S, \bar{S})$ be an (s, t)-cut. Then $|f| \leq cap(S, \bar{S})$.

Warmup Theorem

Theorem

Let f be a feasible (s,t)-flow, and let $(S,\bar{S})$ be an (s,t)-cut. Then $|f| \le \mathbf{cap}(S,\bar{S})$.

$$|f| = \sum_{v\in V} f(s,v) - \sum_{v\in V} f(v,s) \qquad \text{(definition)}$$

Warmup Theorem

Theorem

Let $\boldsymbol{f}$ be a feasible $(\boldsymbol{s},\boldsymbol{t})$-flow, and let $(\boldsymbol{S},\bar{\boldsymbol{S}})$ be an $(\boldsymbol{s},\boldsymbol{t})$-cut. Then $|\boldsymbol{f}| \leq \boldsymbol{cap}(\boldsymbol{S},\bar{\boldsymbol{S}})$.

$$
\begin{aligned}
|f| &= \sum_{v\in V} f(s,v) - \sum_{v\in V} f(v,s) && \text{(definition)} \\
&= \sum_{u\in S}\left(\sum_{v\in V} f(u,v) - \sum_{v\in V} f(v,u)\right) && \text{(flow conservation constraints)}
\end{aligned}
$$

Warmup Theorem

Theorem

Let f be a feasible (s,t)-flow, and let $(S,\bar{S})$ be an (s,t)-cut. Then $|f| \leq cap(S,\bar{S})$.

$$
\begin{aligned}
|f| &= \sum_{v \in V} f(s,v) - \sum_{v \in V} f(v,s) && \text{(definition)} \\
&= \sum_{u \in S} \left(\sum_{v \in V} f(u,v) - \sum_{v \in V} f(v,u) \right) && \text{(flow conservation constraints)} \\
&= \sum_{u \in S} \left(\sum_{v \in \bar{S}} f(u,v) - \sum_{v \in \bar{S}} f(v,u) \right) && \text{(remove terms which cancel)}
\end{aligned}
$$

Warmup Theorem

Theorem

Let f be a feasible (s,t)-flow, and let $(S,\bar{S})$ be an (s,t)-cut. Then $|f| \le cap(S,\bar{S})$.

$$
\begin{aligned}
|f| &= \sum_{v\in V} f(s,v) - \sum_{v\in V} f(v,s) && \text{(definition)}\\
&= \sum_{u\in S}\left(\sum_{v\in V} f(u,v) - \sum_{v\in V} f(v,u)\right) && \text{(flow conservation constraints)}\\
&= \sum_{u\in S}\left(\sum_{v\in \bar{S}} f(u,v) - \sum_{v\in \bar{S}} f(v,u)\right) && \text{(remove terms which cancel)}\\
&\le \sum_{u\in S}\sum_{v\in \bar{S}} f(u,v) && \text{(flow is nonnegative)}
\end{aligned}
$$

Warmup Theorem

Theorem

Let f be a feasible (s,t)-flow, and let $(S,\bar{S})$ be an (s,t)-cut. Then $|f| \le cap(S,\bar{S})$.

$$
\begin{aligned}
|f| &= \sum_{v \in V} f(s,v) - \sum_{v \in V} f(v,s) && \text{(definition)} \\
&= \sum_{u \in S} \left(\sum_{v \in V} f(u,v) - \sum_{v \in V} f(v,u) \right) && \text{(flow conservation constraints)} \\
&= \sum_{u \in S} \left(\sum_{v \in \bar{S}} f(u,v) - \sum_{v \in \bar{S}} f(v,u) \right) && \text{(remove terms which cancel)} \\
&\le \sum_{u \in S} \sum_{v \in \bar{S}} f(u,v) && \text{(flow is nonnegative)} \\
&\le \sum_{u \in S} \sum_{v \in \bar{S}} c(u,v) = cap(S,\bar{S}) && \text{(flow is feasible)}
\end{aligned}
$$

Max-Flow Min-Cut

Corollary

If $\boldsymbol{f}$ avoids every $\bar{\boldsymbol{S}} \to \boldsymbol{S}$ edge and saturates every $\boldsymbol{S} \to \bar{\boldsymbol{S}}$ edge, then $\boldsymbol{f}$ is a maximum flow and $(\boldsymbol{S}, \bar{\boldsymbol{S}})$ is a minimum cut.

Max-Flow Min-Cut

Corollary

If $\boldsymbol{f}$ avoids every $\bar{\boldsymbol{S}} \to \boldsymbol{S}$ edge and saturates every $\boldsymbol{S} \to \bar{\boldsymbol{S}}$ edge, then $\boldsymbol{f}$ is a maximum flow and $(\boldsymbol{S}, \bar{\boldsymbol{S}})$ is a minimum cut.

Theorem (Max-Flow Min-Cut Theorem)

In any flow network, value of max $(\boldsymbol{s}, \boldsymbol{t})$-flow $=$ capacity of min $(\boldsymbol{s}, \boldsymbol{t})$-cut.

Max-Flow Min-Cut

Corollary

If f avoids every $\bar{S} \to S$ edge and saturates every $S \to \bar{S}$ edge, then f is a maximum flow and $(S, \bar{S})$ is a minimum cut.

Theorem (Max-Flow Min-Cut Theorem)

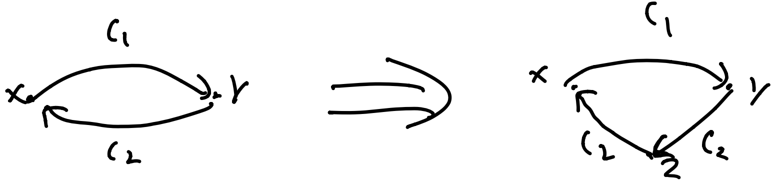
In any flow network, value of max (s, t)-flow $=$ capacity of min (s, t)-cut.

Spend rest of today proving this.

- Many different valid proofs.
- We'll see a classical proof which will naturally lead to algorithms for these problems.

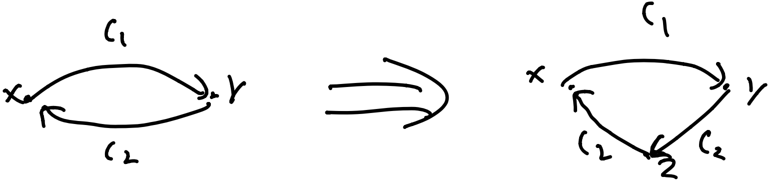
One Direction

Cycles of length **2** will turn out to be annoying. Get rid of them.



One Direction

Cycles of length **2** will turn out to be annoying. Get rid of them.



- Doesn't change max-flow or min-cut
- Increases #edges by constant factor, # nodes to original # edges.

Residual

Let f be feasible (s,t)-flow. Define *residual capacities*:

$$c_f(u,v) = \begin{cases} c(u,v) - f(u,v) & \text{if } (u,v) \in E \\ 0 & \text{otherwise} \end{cases}$$

Residual

Let f be feasible (s,t)-flow. Define *residual capacities*:

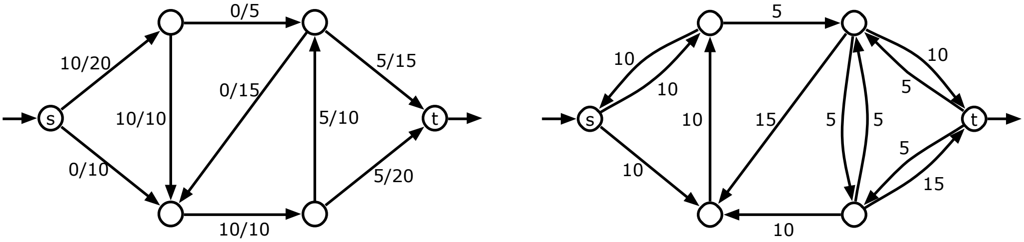
$$c_f(u,v) = \begin{cases} c(u,v) - f(u,v) & \text{if } (u,v) \in E \\ f(v,u) & \text{if } (v,u) \in E \\ 0 & \text{otherwise} \end{cases}$$

Residual

Let f be feasible (s,t)-flow. Define *residual capacities*:

$$c_f(u,v) = \begin{cases} c(u,v) - f(u,v) & \text{if } (u,v) \in E \\ f(v,u) & \text{if } (v,u) \in E \\ 0 & \text{otherwise} \end{cases}$$

Residual Graph: $G_f = (V, E_f)$ where $(u,v) \in E_f$ if $c_f(u,v) > 0$.



A flow f in a weighted graph G and the corresponding residual graph G_f.

Start of Proof

Let f be a max (s, t)-flow with residual graph G_f.
Want to Show: There is a cut $(S, \bar{S})$ with $cap(S, \bar{S}) = |f|$.

Start of Proof

Let f be a max (s,t)-flow with residual graph G_f.
Want to Show: There is a cut $(S,\bar{S})$ with $cap(S,\bar{S}) = |f|$.

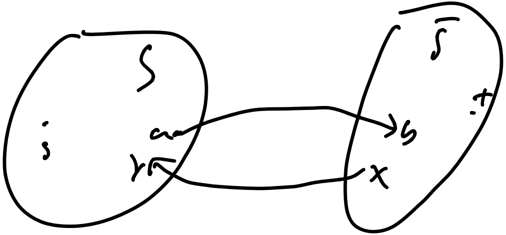
Case 1: There is no $s \to t$ path in G_f

Start of Proof

Let f be a max (s, t)-flow with residual graph G_f.
Want to Show: There is a cut $(S, \bar{S})$ with $cap(S, \bar{S}) = |f|$.

Case 1: There is no $s \to t$ path in G_f

Let $S = \{$vertices reachable from s in $G_f\}$

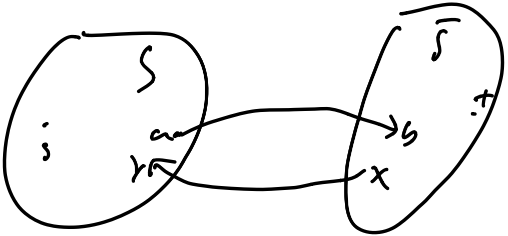


Start of Proof

Let f be a max (s,t)-flow with residual graph G_f.
Want to Show: There is a cut $(S,\bar{S})$ with $cap(S,\bar{S}) = |f|$.

Case 1: There is no $s \to t$ path in G_f

Let $S = \{$vertices reachable from s in $G_f\}$



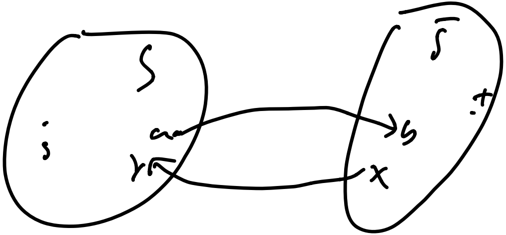
- $(S,\bar{S})$ an (s,t)-cut. ✓

Start of Proof

Let f be a max (s,t)-flow with residual graph G_f.
Want to Show: There is a cut $(S, \bar{S})$ with $cap(S, \bar{S}) = |f|$.

Case 1: There is no $s \to t$ path in G_f

Let $S = \{\text{vertices reachable from } s \text{ in } G_f\}$



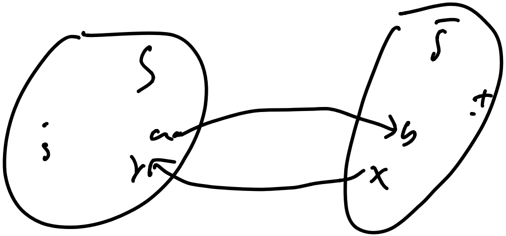
- $(S, \bar{S})$ an (s,t)-cut. ✓
- $c_f(a, b) =$

Start of Proof

> Let f be a max (s, t)-flow with residual graph G_f.
> **Want to Show:** There is a cut $(S, \bar{S})$ with $cap(S, \bar{S}) = |f|$.

Case 1: There is no $s \to t$ path in G_f

Let $S = \{$vertices reachable from s in $G_f\}$



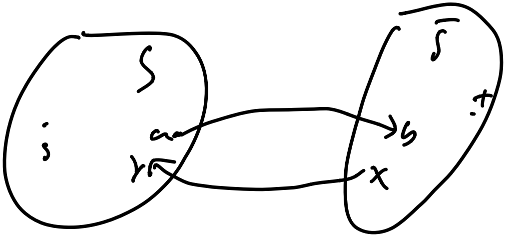
- $(S, \bar{S})$ an (s, t)-cut. ✓
- $c_f(a, b) = 0$

Start of Proof

Let f be a max (s, t)-flow with residual graph G_f.
Want to Show: There is a cut $(S, \bar{S})$ with $cap(S, \bar{S}) = |f|$.

Case 1: There is no $s \to t$ path in G_f

Let $S = \{\text{vertices reachable from } s \text{ in } G_f\}$



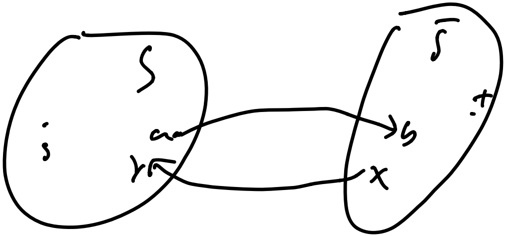
- $(S, \bar{S})$ an (s, t)-cut. ✓
- $c_f(a, b) = 0$
  $\implies c(a, b) - f(a, b) = 0$
  $\implies c(a, b) = f(a, b)$

Start of Proof

Let f be a max (s,t)-flow with residual graph G_f.
Want to Show: There is a cut $(S,\bar{S})$ with $cap(S,\bar{S}) = |f|$.

Case 1: There is no $s \to t$ path in G_f

Let $S = \{\text{vertices reachable from } s \text{ in } G_f\}$



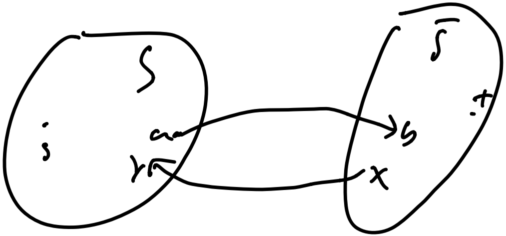
- $(S,\bar{S})$ an (s,t)-cut. ✓
- $c_f(a,b) = 0$
  $\implies c(a,b) - f(a,b) = 0$
  $\implies c(a,b) = f(a,b)$
- $c_f(y,x) =$

Start of Proof

Let f be a max (s,t)-flow with residual graph G_f.
Want to Show: There is a cut $(S,\bar{S})$ with $cap(S,\bar{S}) = |f|$.

Case 1: There is no $s \to t$ path in G_f

Let S = {vertices reachable from s in G_f}



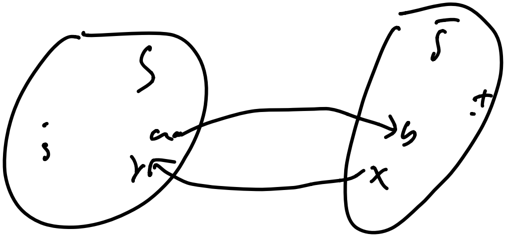
- $(S,\bar{S})$ an (s,t)-cut. ✓
- $c_f(a,b) = 0$
  $\implies c(a,b) - f(a,b) = 0$
  $\implies c(a,b) = f(a,b)$
- $c_f(y,x) = 0$

Start of Proof

Let f be a max (s,t)-flow with residual graph G_f.
Want to Show: There is a cut $(S,\bar{S})$ with $cap(S,\bar{S}) = |f|$.

Case 1: There is no $s \to t$ path in G_f

Let $S = \{$vertices reachable from s in $G_f\}$



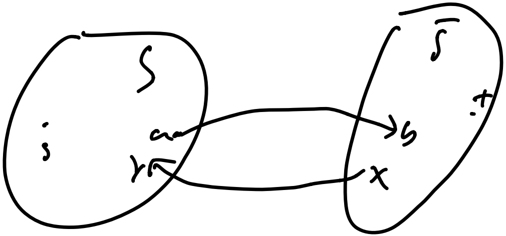
- $(S,\bar{S})$ an (s,t)-cut. ✓
- $c_f(a,b) = 0$
  $\implies c(a,b) - f(a,b) = 0$
  $\implies c(a,b) = f(a,b)$
- $c_f(y,x) = 0$
  $\implies f(x,y) = 0$

Start of Proof

Let f be a max (s,t)-flow with residual graph G_f.
Want to Show: There is a cut $(S,\bar{S})$ with $cap(S,\bar{S}) = |f|$.

Case 1: There is no $s \to t$ path in G_f

Let $S = \{\text{vertices reachable from } s \text{ in } G_f\}$



- $(S,\bar{S})$ an (s,t)-cut. ✓
- $c_f(a,b) = 0$
  $\implies c(a,b) - f(a,b) = 0$
  $\implies c(a,b) = f(a,b)$
- $c_f(y,x) = 0$
  $\implies f(x,y) = 0$

f saturates $S \to \bar{S}$ edges, avoids $\bar{S} \to S$ edges $\implies cap(S,\bar{S}) = |f|$ by corollary

Case 2

Suppose $\exists$ an $s \to t$ path P in G_f.

- Called an *augmenting path*

Idea: show that we can "push" more flow along P, so f not a max flow. Contradiction, can't be in this case.

Case 2

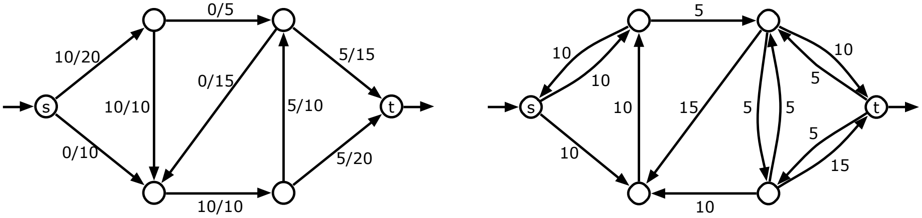
Suppose $\exists$ an $s \to t$ path P in G_f.

- Called an *augmenting path*

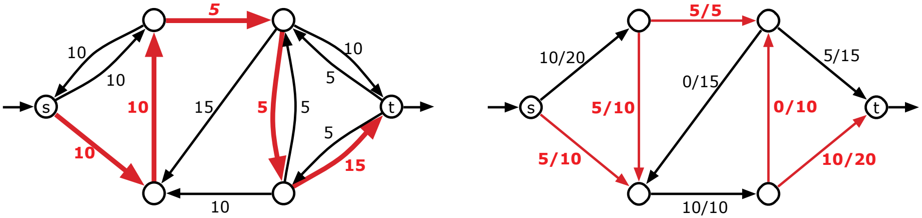
Idea: show that we can "push" more flow along P, so f not a max flow. Contradiction, can't be in this case.

- Foreshadowing: augmenting path allows us to send more flow. Algorithm to increase flow!

Intuition



A flow f in a weighted graph G and the corresponding residual graph G_f.



An augmenting path in G_f with value $F = 5$ and the augmented flow f'.

Formalities

Let P be (simple) augmenting path in G_f. Let $F = \min_{e \in P} c_f(e)$.

Formalities

Let P be (simple) augmenting path in G_f. Let $F = \min_{e \in P} c_f(e)$.

Define new flow f': for all $(u, v) \in E$, let

$$f'(u,v) = \begin{cases} f(u,v) + F & \text{if } (u,v) \text{ in } P \\ f(u,v) - F & \text{if } (v,u) \text{ in } P \\ f(u,v) & \text{otherwise} \end{cases}$$

Formalities

Let P be (simple) augmenting path in G_f. Let $F = \min_{e \in P} c_f(e)$.

Define new flow f': for all $(u, v) \in E$, let

$$f'(u,v) = \begin{cases} f(u,v) + F & \text{if } (u,v) \text{ in } P \\ f(u,v) - F & \text{if } (v,u) \text{ in } P \\ f(u,v) & \text{otherwise} \end{cases}$$

Claim: f' is a feasible (s, t)-flow with $|f'| > |f|$.

Formalities

Let P be (simple) augmenting path in G_f. Let $F = \min_{e \in P} c_f(e)$.

Define new flow f': for all $(u, v) \in E$, let

$$f'(u,v) = \begin{cases} f(u,v) + F & \text{if } (u,v) \text{ in } P \\ f(u,v) - F & \text{if } (v,u) \text{ in } P \\ f(u,v) & \text{otherwise} \end{cases}$$

Claim: f' is a feasible (s,t)-flow with $|f'| > |f|$.

Plan: prove (sketch) each subclaim individually

- $|f'| > |f|$
- f' an (s,t)-flow (flow conservation)
- f' feasible (obeys capacities)

$|f'| > |f|$

Consider first edge of P (out of s), say (s, v_1)

- If $(s, v_1) \in E$, then $f'(s, v_1) = f(s, v_1) + F$
- If $(v_1, s) \in E$ then $f'(v_1, s) = f(v_1, s) - F$

$s \xrightarrow{+F} v_1$ or $s \xleftarrow{-F} v_1$

$|f'| > |f|$

Consider first edge of P (out of s), say (s, v_1)

- If $(s, v_1) \in E$, then $f'(s, v_1) = f(s, v_1) + F$
- If $(v_1, s) \in E$ then $f'(v_1, s) = f(v_1, s) - F$

$s \xrightarrow{+F} v_1$ or $s \xleftarrow{-F} v_1$

$$|f'| = \sum_u f'(s,u) - \sum_u f'(u,s) = |f| + F > |f|$$

f' obeys flow conservation

Consider some $u \in V \setminus \{s, t\}$.

f' obeys flow conservation

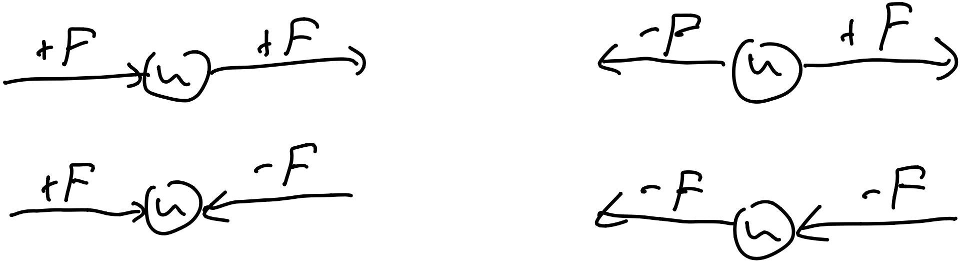
Consider some $u \in V \setminus \{s, t\}$.

- If $u \notin P$, no change in flow at $u \implies$ still balanced.

f' obeys flow conservation

Consider some $u \in V \setminus \{s, t\}$.

- If $u \notin P$, no change in flow at $u \implies$ still balanced.
- If $u \in P$, four possibilities:



f' obeys capacity constraints

Let $(u, v) \in E$

f' obeys capacity constraints

Let $(u, v) \in E$

- If $(u, v), (v, u) \notin P$: $f'(u, v) = f(u, v) \leq c(u, v)$

f' obeys capacity constraints

Let $(u, v) \in E$

- If $(u, v), (v, u) \notin P$: $f'(u, v) = f(u, v) \leq c(u, v)$
- If $(u, v) \in P$:

f' obeys capacity constraints

Let $(u, v) \in E$

- If $(u, v), (v, u) \notin P$: $f'(u, v) = f(u, v) \leq c(u, v)$
- If $(u, v) \in P$:

$$\begin{aligned} f'(u, v) &= f(u, v) + F \\ &\leq f(u, v) + c_f(u, v) \\ &= f(u, v) + c(u, v) - f(u, v) \\ &= c(u, v) \end{aligned}$$

f' obeys capacity constraints

Let $(u, v) \in E$

- If $(u, v), (v, u) \notin P$: $f'(u, v) = f(u, v) \leq c(u, v)$
- If $(u, v) \in P$:

$$\begin{aligned} f'(u, v) &= f(u, v) + F \\ &\leq f(u, v) + c_f(u, v) \\ &= f(u, v) + c(u, v) - f(u, v) \\ &= c(u, v) \end{aligned}$$

- If $(v, u) \in P$:

f' obeys capacity constraints

Let $(u, v) \in E$

- If $(u, v), (v, u) \notin P$: $f'(u, v) = f(u, v) \leq c(u, v)$

- If $(u, v) \in P$:

$$\begin{aligned} f'(u, v) &= f(u, v) + F \\ &\leq f(u, v) + c_f(u, v) \\ &= f(u, v) + c(u, v) - f(u, v) \\ &= c(u, v) \end{aligned}$$

- If $(v, u) \in P$:

$$\begin{aligned} f'(u, v) &= f(u, v) - F \\ &\geq f(u, v) - c_f(v, u) \\ &= f(u, v) - f(u, v) \\ &= 0 \end{aligned}$$

Ford-Fulkerson Algorithm and Integrality

FF Algorithm

Obvious algorithm from previous proof: keep pushing flow!

$\boldsymbol{f} = \vec{\mathbf{0}}$
while($\exists \boldsymbol{s} \to \boldsymbol{t}$ path $\boldsymbol{P}$ in $\boldsymbol{G_f}$) {
    $\boldsymbol{F} = \mathbf{min}_{\boldsymbol{e} \in \boldsymbol{P}}\, \boldsymbol{c_f(e)}$
    Push $\boldsymbol{F}$ flow along $\boldsymbol{P}$ to get new flow $\boldsymbol{f'}$
    $\boldsymbol{f} = \boldsymbol{f'}$
}
return $\boldsymbol{f}$

FF Algorithm

Obvious algorithm from previous proof: keep pushing flow!

$\boldsymbol{f} = \vec{\mathbf{0}}$
while($\exists \boldsymbol{s} \to \boldsymbol{t}$ path $\boldsymbol{P}$ in $\boldsymbol{G_f}$) {
    $\boldsymbol{F} = \min_{e \in P} c_f(e)$
    Push $\boldsymbol{F}$ flow along $\boldsymbol{P}$ to get new flow $\boldsymbol{f'}$
    $\boldsymbol{f} = \boldsymbol{f'}$
}
return $\boldsymbol{f}$ or $\{\boldsymbol{v} \in \boldsymbol{V} : \boldsymbol{v}$ reachable from $\boldsymbol{s}$ in $\boldsymbol{G_f}\}$

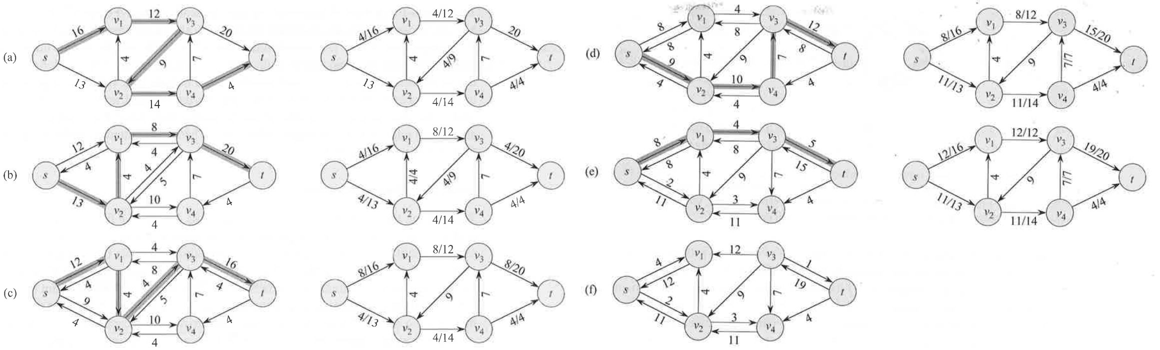
FF Algorithm

Obvious algorithm from previous proof: keep pushing flow!

$f = \vec{0}$
while($\exists s \to t$ path P in G_f) {
    $F = \min_{e \in P} c_f(e)$
    Push F flow along P to get new flow f'
    $f = f'$
}
return f or $\{v \in V : v$ reachable from s in $G_f\}$

Correctness: directly from previous proof

Example



Integrality

Corollary

If all capacities are integers, then there is a max flow such that the flow through every edge is an integer

Integrality

Corollary

If all capacities are integers, then there is a max flow such that the flow through every edge is an integer

Proof.

Induction on iterations of the Ford-Fulkerson algorithm: initially true, stays true $\implies$ true at end. $\square$

Running Time

Theorem

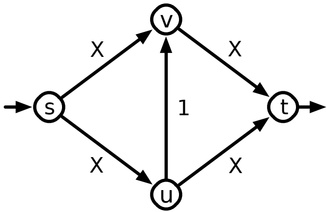
If all capacities are integers and the max flow value is F, Ford-Fulkerson takes time at most $O(F(m+n))$

Running Time

Theorem

If all capacities are integers and the max flow value is F, Ford-Fulkerson takes time at most $O(F(m+n))$

Finding path takes $O(m+n)$ time, increase flow by at least 1



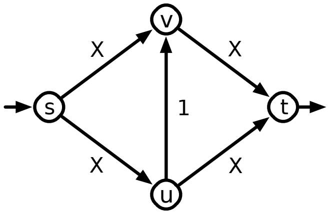
A bad example for the Ford-Fulkerson algorithm.

Running Time

Theorem

If all capacities are integers and the max flow value is $\boldsymbol{F}$, Ford-Fulkerson takes time at most $\boldsymbol{O(F(m+n))}$

Finding path takes $\boldsymbol{O(m+n)}$ time, increase flow by at least $\mathbf{1}$



A bad example for the Ford-Fulkerson algorithm.

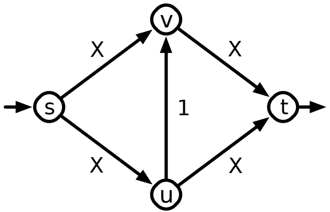
Running time $\geq \#$ iterations.

Running Time

Theorem

If all capacities are integers and the max flow value is F, Ford-Fulkerson takes time at most $O(F(m+n))$

Finding path takes $O(m+n)$ time, increase flow by at least 1



A bad example for the Ford-Fulkerson algorithm.

Running time $\geq$ # iterations.
This example:

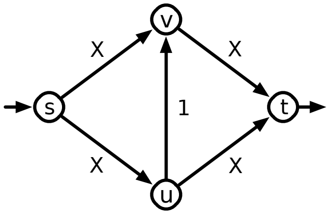
- Running time: $\Omega(x)$

Running Time

Theorem

If all capacities are integers and the max flow value is F, Ford-Fulkerson takes time at most $O(F(m+n))$

Finding path takes $O(m+n)$ time, increase flow by at least $\mathbf{1}$



A bad example for the Ford-Fulkerson algorithm.

Running time ≥ # iterations.

This example:

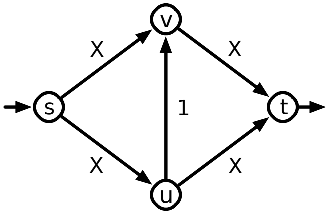
- Running time: $\Omega(x)$
- Input size $O(\log x) + O(1)$

Running Time

Theorem

If all capacities are integers and the max flow value is F, Ford-Fulkerson takes time at most $O(F(m+n))$

Finding path takes $O(m+n)$ time, increase flow by at least $\mathbf{1}$



A bad example for the Ford-Fulkerson algorithm.

Running time $\geq$ # iterations.
This example:

- Running time: $\Omega(x)$
- Input size $O(\log x) + O(1)$

$\Longrightarrow$ Exponential time!