Network Flows and Cuts

Lecture 13

March 6, 2018

Most slides are courtesy Prof. Chekuri

How many edges to cut?

For the graph depicted on the right. How many edges have to be cut before there is no path from s to t?

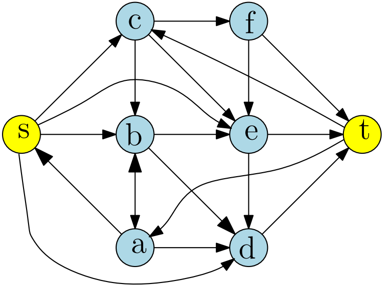
(A) 1

(B) 2

(C) 3

(D) 4

(E) 5



How many edges to cut?

For the graph depicted on the right. How many edges have to be cut before there is no path from ***s*** to ***t***?

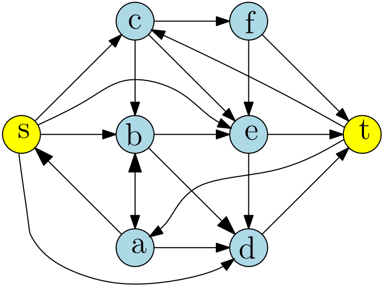
(A) 1

(B) 2

(C) 3

(D) 4

(E) 5

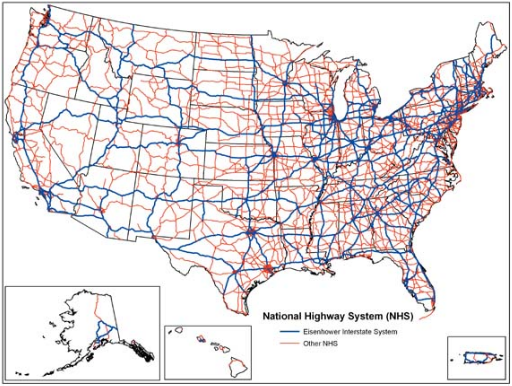


Related Q. At most how many edge disjoint paths from ***s*** to ***t***?

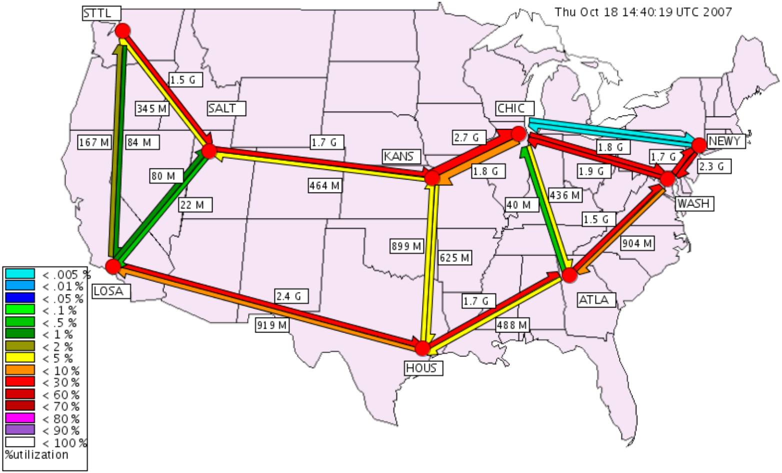
Part I

Network Flows: Introduction and Setup

Transportation/Road Network



Internet Backbone Network



Common Features of Flow Networks

1. **Network** represented by a (directed) *graph* $G = (V, E)$.
2. Each edge e has a **capacity** $c(e) \geq 0$ that limits amount of *traffic* on e.
3. *Source(s)* of traffic/data.
4. *Sink(s)* of traffic/data.
5. Traffic *flows* from sources to sinks.
6. Traffic is *switched/interchanged* at nodes.

Common Features of Flow Networks

1. **Network** represented by a (directed) *graph* $G = (V, E)$.
2. Each edge e has a **capacity** $c(e) \geq 0$ that limits amount of *traffic* on e.
3. *Source(s)* of traffic/data.
4. *Sink(s)* of traffic/data.
5. Traffic *flows* from sources to sinks.
6. Traffic is *switched/interchanged* at nodes.

Flow abstract term to indicate stuff (traffic/data/etc) that **flows** from sources to sinks.

Single Source/Single Sink Flows

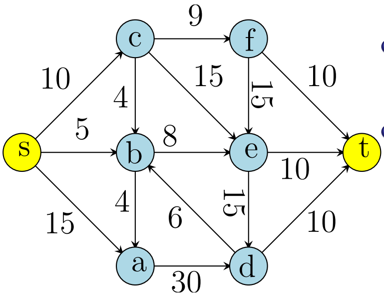
Simple setting:

- Single source s and single sink t.
- Every other node v is an **internal** node.
- Flow originates at s and terminates at t.

Single Source/Single Sink Flows

Simple setting:

- Single source s and single sink t.
- Every other node v is an **internal** node.
- Flow originates at s and terminates at t.

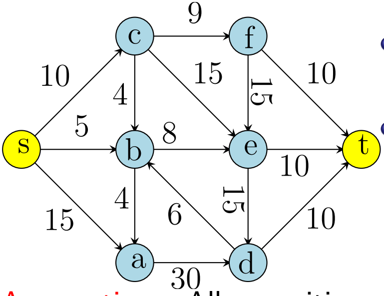


- Each edge e has a capacity $c(e) \geq 0$.
- Sometimes assume: Source $s \in V$ has no incoming edges, and sink $t \in V$ has no outgoing edges.

Single Source/Single Sink Flows

Simple setting:

- Single source s and single sink t.
- Every other node v is an **internal** node.
- Flow originates at s and terminates at t.



- Each edge e has a capacity $c(e) \geq 0$.
- Sometimes assume: Source $s \in V$ has no incoming edges, and sink $t \in V$ has no outgoing edges.

Assumptions: All capacities are integer, and every vertex has at least one edge incident to it.

Definition of Flow

Two ways to define flows:

1. edge based, or
2. path based.

Essentially equivalent but have different uses.

Edge based definition is more compact.

Edge Based Definition of Flow

Definition

Flow in network $G = (V, E)$, is function $f : E \rightarrow \mathbb{R}^{\geq 0}$ s.t.

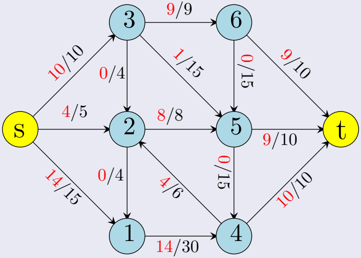


Figure: Flow with value.

Edge Based Definition of Flow

Definition

Flow in network $G = (V, E)$, is function $f : E \to \mathbb{R}^{\geq 0}$ s.t.

1. Capacity Constraint: For each edge e, $f(e) \leq c(e)$.

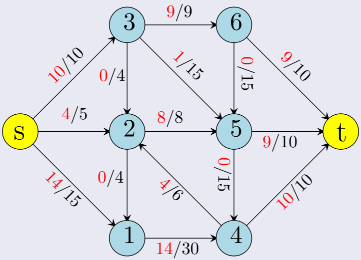


Figure: Flow with value.

Edge Based Definition of Flow

Definition

Flow in network $G = (V, E)$, is function $f : E \to \mathbb{R}^{\geq 0}$ s.t.

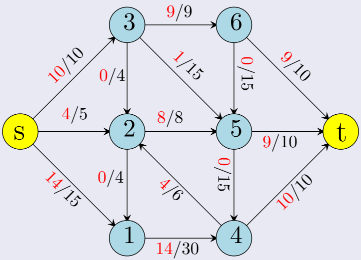


Figure: Flow with value.

1. Capacity Constraint: For each edge e, $f(e) \leq c(e)$.
2. Conservation Constraint: For each vertex $v \neq s, t$.

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

Edge Based Definition of Flow

Definition

Flow in network $G = (V, E)$, is function $f : E \to \mathbb{R}^{\geq 0}$ s.t.

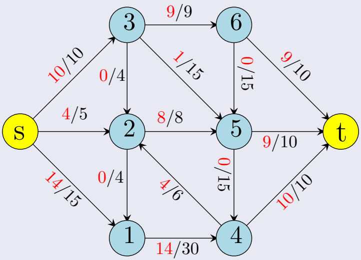


Figure: Flow with value.

1. Capacity Constraint: For each edge e, $f(e) \leq c(e)$.
2. Conservation Constraint: For each vertex $v \neq s, t$.

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

3. Value of flow= (total flow out of source) − (total flow in to source).

More Definitions and Notation

Flow in and out of vertex v

$$f^{\text{in}}(v) = \sum_{e \text{ into } v} f(e) \qquad f^{\text{out}}(v) = \sum_{e \text{ out of } v} f(e)$$

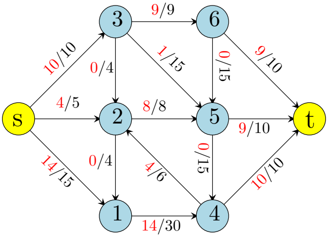


Figure: Flow with value.

More Definitions and Notation

Flow in and out of vertex v

$$f^{\text{in}}(v) = \sum_{e \text{ into } v} f(e) \qquad f^{\text{out}}(v) = \sum_{e \text{ out of } v} f(e)$$

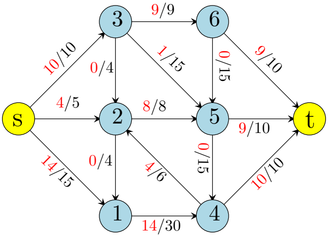


Figure: Flow with value.

More Definitions and Notation

Flow in and out of vertex v

$$f^{\text{in}}(v) = \sum_{e \text{ into } v} f(e) \qquad f^{\text{out}}(v) = \sum_{e \text{ out of } v} f(e)$$

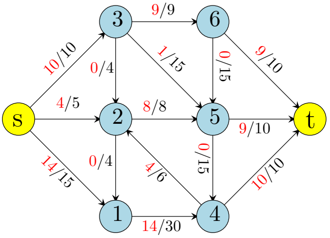


Figure: Flow with value.

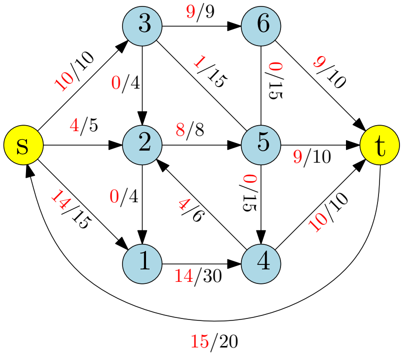
Definition. For network G with source s, the **value** of flow f is defined as

$$v(f) = f^{\text{out}}(s) - f^{\text{in}}(s)$$

Value of flow?

In the flow depicted on the right, the value of the flow is.

(A) 6.
(B) 13.
(C) 18.
(D) 28.
(E) 43.



A Path Based Definition of Flow

Intuition: Flow goes from source s to sink t along a path.

$\mathcal{P}$: set of all *simple* paths from s to t. $|\mathcal{P}|$ can be **exponential** in n.

A Path Based Definition of Flow

Intuition: Flow goes from source s to sink t along a path.

$\mathcal{P}$: set of all *simple* paths from s to t. $|\mathcal{P}|$ can be **exponential** in n.

Definition (Flow by paths.)

A **flow** in network $G = (V, E)$, is function $f : \mathcal{P} \rightarrow \mathbb{R}^{\geq 0}$ s.t.

A Path Based Definition of Flow

Intuition: Flow goes from source s to sink t along a path.

$\mathcal{P}$: set of all *simple* paths from s to t. $|\mathcal{P}|$ can be **exponential** in n.

Definition (Flow by paths.)

A **flow** in network $G = (V, E)$, is function $f : \mathcal{P} \rightarrow \mathbb{R}^{\geq 0}$ s.t.

1. Capacity Constraint: For each edge e, total flow on e is $\leq c(e)$.

A Path Based Definition of Flow

Intuition: Flow goes from source s to sink t along a path.

$\mathcal{P}$: set of all *simple* paths from s to t. $|\mathcal{P}|$ can be **exponential** in n.

Definition (Flow by paths.)

A **flow** in network $G = (V, E)$, is function $f : \mathcal{P} \rightarrow \mathbb{R}^{\geq 0}$ s.t.

1. Capacity Constraint: For each edge e, total flow on e is $\leq c(e)$.

$$\sum_{p \in \mathcal{P} : e \in p} f(p) \leq c(e)$$

A Path Based Definition of Flow

Intuition: Flow goes from source s to sink t along a path.

$\mathcal{P}$: set of all *simple* paths from s to t. $|\mathcal{P}|$ can be **exponential** in n.

Definition (Flow by paths.)

A **flow** in network $G = (V, E)$, is function $f : \mathcal{P} \rightarrow \mathbb{R}^{\geq 0}$ s.t.

1. Capacity Constraint: For each edge e, total flow on e is $\leq c(e)$.

$$\sum_{p \in \mathcal{P} : e \in p} f(p) \leq c(e)$$

2. Conservation Constraint:

A Path Based Definition of Flow

Intuition: Flow goes from source s to sink t along a path.

$\mathcal{P}$: set of all *simple* paths from s to t. $|\mathcal{P}|$ can be **exponential** in n.

Definition (Flow by paths.)

A **flow** in network $G = (V, E)$, is function $f : \mathcal{P} \to \mathbb{R}^{\geq 0}$ s.t.

1. Capacity Constraint: For each edge e, total flow on e is $\leq c(e)$.

$$\sum_{p \in \mathcal{P} : e \in p} f(p) \leq c(e)$$

2. Conservation Constraint: No need! Automatic.

A Path Based Definition of Flow

Intuition: Flow goes from source s to sink t along a path.

$\mathcal{P}$: set of all *simple* paths from s to t. $|\mathcal{P}|$ can be **exponential** in n.

Definition (Flow by paths.)

A **flow** in network $G = (V, E)$, is function $f : \mathcal{P} \rightarrow \mathbb{R}^{\geq 0}$ s.t.

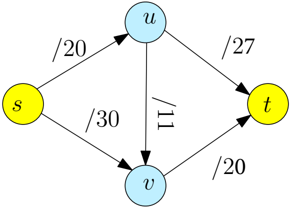
1. Capacity Constraint: For each edge e, total flow on e is $\leq c(e)$.

$$\sum_{p \in \mathcal{P}: e \in p} f(p) \leq c(e)$$

2. Conservation Constraint: No need! Automatic.

Value of flow: $\sum_{p \in \mathcal{P}} f(p)$.

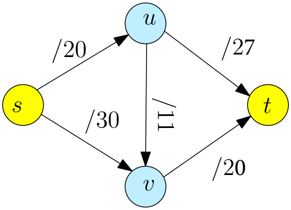
Example



$\mathcal{P} = \{p_1, p_2, p_3\}$
$p_1 : s \rightarrow u \rightarrow t$
$p_2 : s \rightarrow u \rightarrow v \rightarrow t$
$p_3 : s \rightarrow v \rightarrow t$

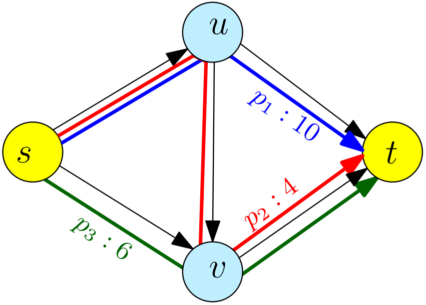
$f(p_1) = 10, f(p_2) = 4, f(p_3) = 6$

Example



$\mathcal{P} = \{p_1, p_2, p_3\}$
$p_1 : s \rightarrow u \rightarrow t$
$p_2 : s \rightarrow u \rightarrow v \rightarrow t$
$p_3 : s \rightarrow v \rightarrow t$

$f(p_1) = 10, f(p_2) = 4, f(p_3) = 6$



Path based flow implies edge based flow

Lemma

Given a path based flow $f : \mathcal{P} \rightarrow \mathbb{R}^{\geq 0}$ *there is an edge based flow* $f' : E \rightarrow \mathbb{R}^{\geq 0}$ *of the same value.*

Proof.

□

Path based flow implies edge based flow

Lemma

Given a path based flow $f : \mathcal{P} \rightarrow \mathbb{R}^{\geq 0}$ *there is an edge based flow* $f' : E \rightarrow \mathbb{R}^{\geq 0}$ *of the same value.*

Proof.

For each edge e define $f'(e) = \sum_{p: e \in p} f(p)$.

□

Path based flow implies edge based flow

Lemma

Given a path based flow $f : \mathcal{P} \rightarrow \mathbb{R}^{\geq 0}$ *there is an edge based flow* $f' : E \rightarrow \mathbb{R}^{\geq 0}$ *of the same value.*

Proof.

For each edge e define $f'(e) = \sum_{p:e \in p} f(p)$.
f' satisfies capacity and conservation constraints. (**Exercise**)

□

Capacity Constraint: For each edge e, total flow on e is $\leq c(e)$.

$$\sum_{p \in \mathcal{P}: e \in p} f(p) \leq c(e)$$

Path based flow implies edge based flow

Lemma

Given a path based flow $f : \mathcal{P} \rightarrow \mathbb{R}^{\geq 0}$ there is an edge based flow $f' : E \rightarrow \mathbb{R}^{\geq 0}$ of the same value.

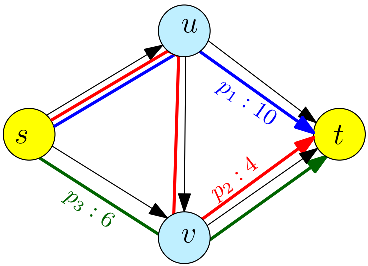
Proof.

For each edge e define $f'(e) = \sum_{p:e \in p} f(p)$.
f' satisfies capacity and conservation constraints. (**Exercise**)
Value of f and f' are equal. (**Exercise**) □

Capacity Constraint: For each edge e, total flow on e is $\leq c(e)$.

$$\sum_{p \in \mathcal{P}: e \in p} f(p) \leq c(e)$$

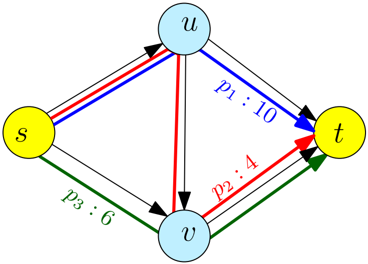
Example



$\mathcal{P} = \{p_1, p_2, p_3\}$
$p_1 : s \to u \to t$
$p_2 : s \to u \to v \to t$
$p_3 : s \to v \to t$

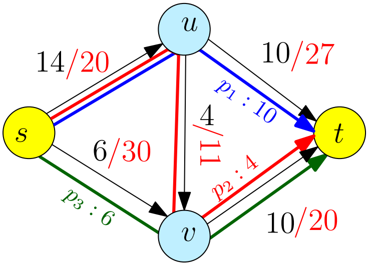
$f(p_1) = 10, f(p_2) = 4, f(p_3) = 6$

Example



$\mathcal{P} = \{p_1, p_2, p_3\}$
$p_1 : s \to u \to t$
$p_2 : s \to u \to v \to t$
$p_3 : s \to v \to t$

$f(p_1) = 10, f(p_2) = 4, f(p_3) = 6$



$f'(s \to u) = 14$
$f'(u \to v) = 4$
$f'(s \to v) = 6$
$f'(u \to t) = 10$
$f'(v \to t) = 10$

Flow Decomposition

Edge based flow to Path based Flow

Lemma

Given an edge based flow $f' : E \rightarrow \mathbb{R}^{\geq 0}$*, there is a path based flow* $f : \mathcal{P} \rightarrow \mathbb{R}^{\geq 0}$ *of same value.*

Flow Decomposition

Edge based flow to Path based Flow

Lemma

Given an edge based flow $f' : E \to \mathbb{R}^{\geq 0}$*, there is a path based flow* $f : \mathcal{P} \to \mathbb{R}^{\geq 0}$ *of same value. Moreover,* f *assigns non-negative flow to at most* m *paths where* $|E| = m$ *and* $|V| = n$*.*

Flow Decomposition

Edge based flow to Path based Flow

Lemma

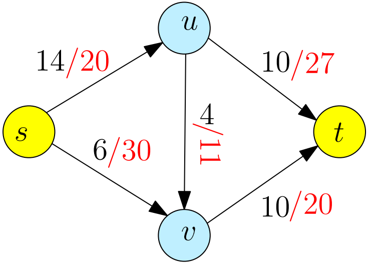
Given an edge based flow $f' : E \rightarrow \mathbb{R}^{\geq 0}$*, there is a path based flow* $f : \mathcal{P} \rightarrow \mathbb{R}^{\geq 0}$ *of same value. Moreover,* f *assigns non-negative flow to at most* m *paths where* $|E| = m$ *and* $|V| = n$*.*

Given f'*, the path based flow can be computed in* $O(mn)$ *time.*

Flow Decomposition

Example

How to decompose the following flow:



Flow Decomposition

Edge based flow to Path based Flow

Algorithm

1. Remove all edges with $f'(e) = 0$.
2. Find a path p from s to t.

Flow Decomposition

Edge based flow to Path based Flow

Algorithm

1. Remove all edges with $f'(e) = 0$.
2. Find a path p from s to t.
3. Assign $f(p)$ to be $\min_{e \in p} f'(e)$.

Flow Decomposition

Edge based flow to Path based Flow

Algorithm

1. Remove all edges with $f'(e) = 0$.
2. Find a path p from s to t.
3. Assign $f(p)$ to be $\min_{e \in p} f'(e)$.
4. Reduce $f'(e)$ for all $e \in p$ by $f(p)$.

Flow Decomposition

Edge based flow to Path based Flow

Algorithm

1. Remove all edges with $f'(e) = 0$.
2. Find a path p from s to t.
3. Assign $f(p)$ to be $\min_{e \in p} f'(e)$.
4. Reduce $f'(e)$ for all $e \in p$ by $f(p)$.
5. Repeat until no path from s to t.

Flow Decomposition

Edge based flow to Path based Flow

Algorithm

1. Remove all edges with $f'(e) = 0$.
2. Find a path p from s to t.
3. Assign $f(p)$ to be $\min_{e \in p} f'(e)$.
4. Reduce $f'(e)$ for all $e \in p$ by $f(p)$.
5. Repeat until no path from s to t.

Proof Idea.

- In each iteration at least one edge has flow reduced to zero.

Flow Decomposition

Edge based flow to Path based Flow

Algorithm

1. Remove all edges with $f'(e) = 0$.
2. Find a path p from s to t.
3. Assign $f(p)$ to be $\min_{e \in p} f'(e)$.
4. Reduce $f'(e)$ for all $e \in p$ by $f(p)$.
5. Repeat until no path from s to t.

Proof Idea.

- In each iteration at least one edge has flow reduced to zero.
- Hence, at most m iterations. Can be implemented in

Flow Decomposition

Edge based flow to Path based Flow

Algorithm

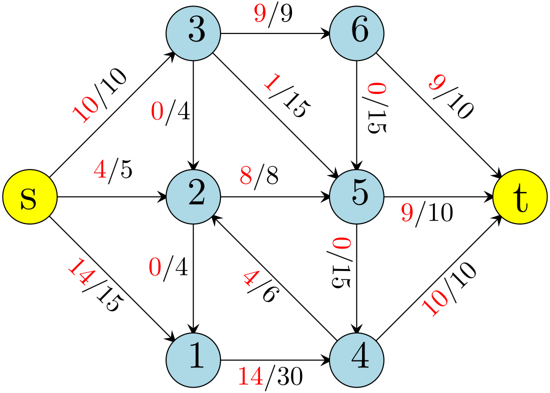
1. Remove all edges with $f'(e) = 0$.
2. Find a path p from s to t.
3. Assign $f(p)$ to be $\min_{e \in p} f'(e)$.
4. Reduce $f'(e)$ for all $e \in p$ by $f(p)$.
5. Repeat until no path from s to t.

Proof Idea.

- In each iteration at least one edge has flow reduced to zero.
- Hence, at most m iterations. Can be implemented in $O(m(m+n))$ time. $O(mn)$ time requires care. □

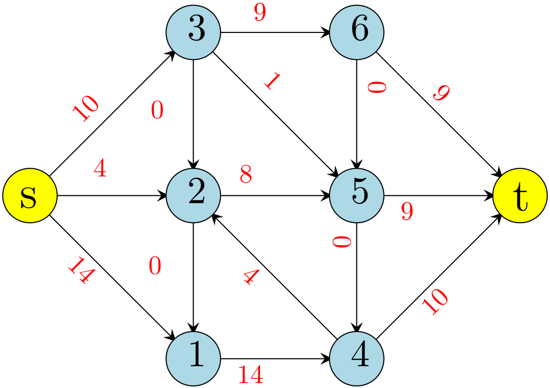
Example

flow/capacity



Example

flow



Summary: Edge vs Path based Flow

Edge based flows:

1. **compact** representation, only m values to be specified, and
2. need to check flow conservation explicitly at each internal node.

Path flows:

1. in some applications, paths more natural,
2. not compact,
3. no need to check flow conservation constraints.

Equivalence shows that we can go back and forth easily.

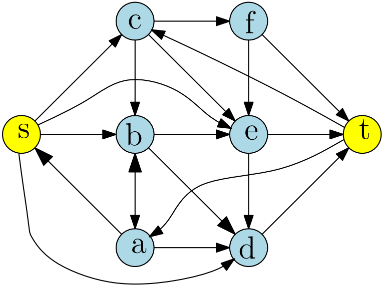
Back to the begining

If $f : \mathcal{P} \rightarrow \mathbb{R}^+$ is a path based flow on this network, then can paths p, p' with $f(p), f(p') = 1$ share edges?

(A) Yes

(B) No

(C) May be



Capacity 1 on all edges.

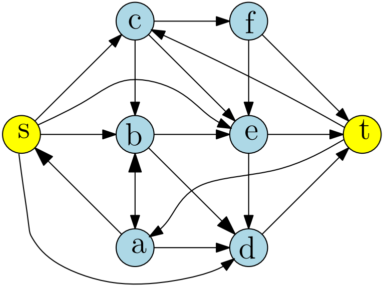
Back to the begining

If $f : \mathcal{P} \to \mathbb{R}^+$ is a path based flow on this network, then can paths p, p' with $f(p), f(p') = 1$ share edges?

(A) Yes

(B) No

(C) May be



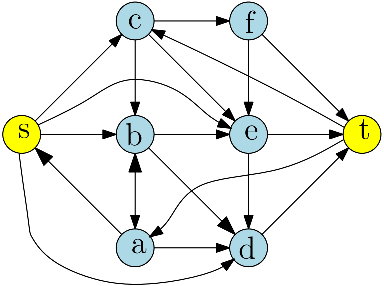
Capacity 1 on all edges.

Paths with flow 1 are edge disjoint.

Back to the begining

If $f : \mathcal{P} \to \mathbb{R}^+$ is a path based flow on this network, then can paths p, p' with $f(p), f(p') = 1$ share edges?

- **(A)** Yes
- **(B)** No
- **(C)** May be



Capacity 1 on all edges.

Paths with flow 1 are edge disjoint.

Value of the flow $\leq$ # edge disjoint paths. (**Exercise**)

The Maximum-Flow Problem

Problem

Input A network G with capacity c and source s and sink t.

Goal Find flow of **maximum** value.

The Maximum-Flow Problem

Problem

Input A network G with capacity c and source s and sink t.

Goal Find flow of **maximum** value.

Question: Given a flow network, what is an *upper bound* on the maximum flow between source and sink?

Part II

Cuts

Cuts

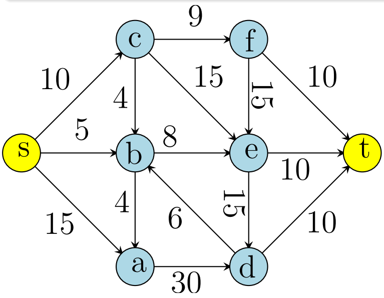
Definition (s-t cut)

Given a flow network an **s-t cut** is a set of edges $E' \subset E$ such that removing E' *disconnects* s from t: in other words there is no directed $s \rightarrow t$ path in $E - E'$.
The **capacity** of a cut E' is $c(E') = \sum_{e \in E'} c(e)$.

Cuts

Definition (s-t cut)

Given a flow network an **s-t cut** is a set of edges $E' \subset E$ such that removing E' *disconnects* s from t: in other words there is no directed $s \to t$ path in $E - E'$.
The **capacity** of a cut E' is $c(E') = \sum_{e \in E'} c(e)$.

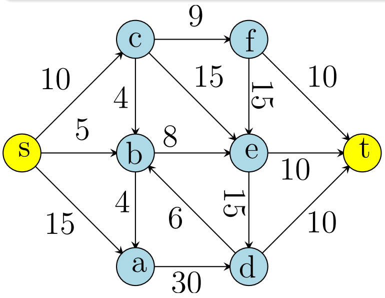


Cuts

Definition (s-t cut)

Given a flow network an **s-t cut** is a set of edges $E' \subset E$ such that removing E' *disconnects* s from t: in other words there is no directed $s \to t$ path in $E - E'$.

The **capacity** of a cut E' is $c(E') = \sum_{e \in E'} c(e)$.

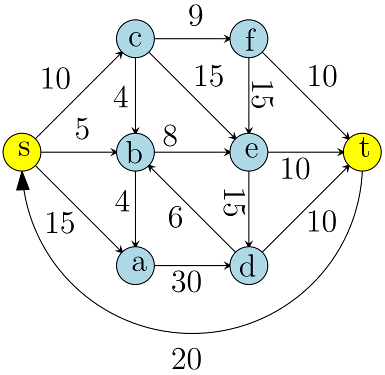


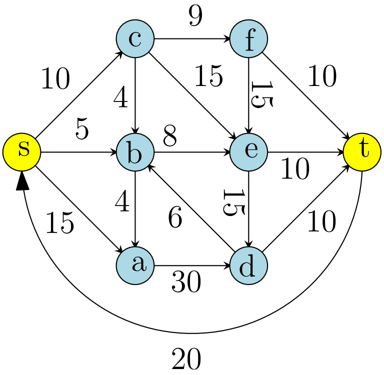
Caution:

1. Cut may leave $t \to s$ paths!
2. There might be many s-t cuts.

s — t cuts

A death by a thousand cuts

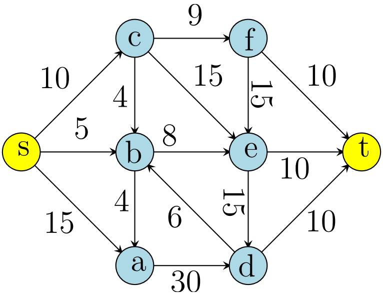




Minimal Cut

Definition (Minimal s-t cut.)

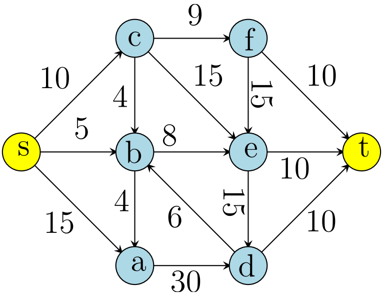
Given a s-t flow network $G = (V, E)$, $E' \subseteq E$ is a **minimal cut** if for all $e \in E'$, if $E' \setminus \{e\}$ is not a cut.



Minimal Cut

Definition (Minimal s-t cut.)

Given a s-t flow network $G = (V, E)$, $E' \subseteq E$ is a **minimal cut** if for all $e \in E'$, if $E' \setminus \{e\}$ is not a cut.



Observation: given a cut E', can check efficiently whether E' is a minimal cut or not. How?

Is this a minimal cut?

Definition (Minimal s-t cut.)

Given a s-t flow network $G = (V, E)$ with n vertices and m edges, $E' \subseteq E$ is a **minimal cut** if for all $e \in E'$, $E' \setminus \{e\}$ is not a cut.

Checking if a set E' forms a minimal s-t cut can be done in

- **(A)** $O(n + m)$.
- **(B)** $O(n \log n + m)$.
- **(C)** $O((n + m) \log n)$.
- **(D)** $O(nm)$.
- **(E)** $O(nm \log n)$.
- **(F)** You flow, me cut.

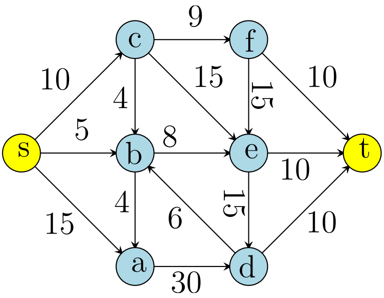
Cuts as Vertex Partitions

Let $A \subset V$ such that

1. $s \in A$, $t \notin A$, and
2. $B = V \setminus A$ (hence $t \in B$).

The **cut** (A, B) is the set of edges

$$c(A, B) = \{(u, v) \in E \mid u \in A, v \in B\}.$$



Cuts as Vertex Partitions

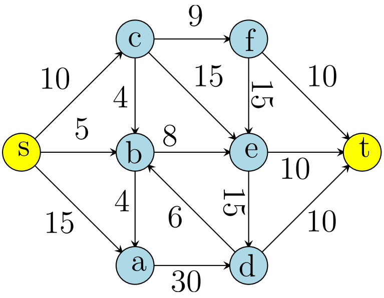
Let $A \subset V$ such that

1. $s \in A$, $t \notin A$, and
2. $B = V \setminus A$ (hence $t \in B$).

The **cut** (A, B) is the set of edges

$$c(A, B) = \{(u, v) \in E \mid u \in A, v \in B\}.$$

Cut $c(A, B)$ is set of edges leaving A.



Cuts as Vertex Partitions

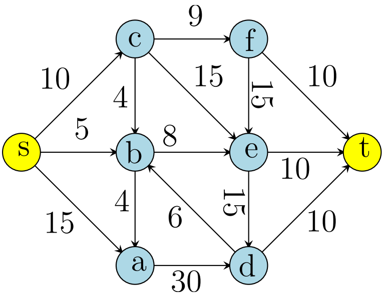
Let $A \subset V$ such that

1. $s \in A$, $t \notin A$, and
2. $B = V \setminus A$ (hence $t \in B$).

The **cut** (A, B) is the set of edges

$$c(A, B) = \{(u, v) \in E \mid u \in A, v \in B\}.$$

Cut $c(A, B)$ is set of edges leaving A.



Claim

$c(A, B)$ is an s-t cut.

Cuts as Vertex Partitions

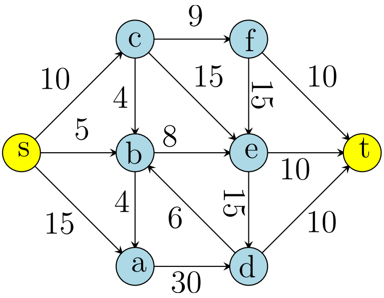
Let $A \subset V$ such that

1. $s \in A$, $t \notin A$, and
2. $B = V \setminus A$ (hence $t \in B$).

The **cut** (A, B) is the set of edges

$$c(A, B) = \{(u, v) \in E \mid u \in A, v \in B\}.$$

Cut $c(A, B)$ is set of edges leaving A.



Claim

$c(A, B)$ is an s-t cut.

Proof.

Let P be any $s \to t$ path in G. Since t is not in A, P has to leave A via some edge (u, v) in $c(A, B)$. □

Cuts as Vertex Partitions

Lemma

Suppose E' *is an* s-t *cut. Then there is a cut* $c(A, B)$ *such that* $c(A, B) \subseteq E'$.

Cuts as Vertex Partitions

Lemma

Suppose E' is an s-t cut. Then there is a cut $c(A, B)$ such that $c(A, B) \subseteq E'$.

Proof.

E' is an s-t cut implies no path from s to t in $(V, E - E')$.

1. Let A be set of all nodes reachable by s in $(V, E - E')$.
2. Since E' is a cut, $t \not\in A$.
3. Claim: $c(A, B) \subseteq E'$.

Cuts as Vertex Partitions

Lemma

Suppose E' *is an* s-t *cut. Then there is a cut* $c(A, B)$ *such that* $c(A, B) \subseteq E'$.

Proof.

E' is an s-t cut implies no path from s to t in $(V, E - E')$.

1. Let A be set of all nodes reachable by s in $(V, E - E')$.
2. Since E' is a cut, $t \not\in A$.
3. Claim: $c(A, B) \subseteq E'$. If not, then some edge $(u, v) \in c(A, B)$ is not in E'.

Cuts as Vertex Partitions

Lemma

Suppose E' *is an* s-t *cut. Then there is a cut* $c(A, B)$ *such that* $c(A, B) \subseteq E'$.

Proof.

E' is an s-t cut implies no path from s to t in $(V, E - E')$.

1. Let A be set of all nodes reachable by s in $(V, E - E')$.
2. Since E' is a cut, $t \notin A$.
3. Claim: $c(A, B) \subseteq E'$. If not, then some edge $(u, v) \in c(A, B)$ is not in E'. This implies, (i) $v \notin A$, (ii) v will be reachable by s and should be in A.

Cuts as Vertex Partitions

Lemma

Suppose E' *is an* s-t *cut. Then there is a cut* $c(A, B)$ *such that* $c(A, B) \subseteq E'$.

Proof.

E' is an s-t cut implies no path from s to t in $(V, E - E')$.

1. Let A be set of all nodes reachable by s in $(V, E - E')$.
2. Since E' is a cut, $t \notin A$.
3. Claim: $c(A, B) \subseteq E'$. If not, then some edge $(u, v) \in c(A, B)$ is not in E'. This implies, (i) $v \notin A$, (ii) v will be reachable by s and should be in A. A contradiction. □

Cuts as Vertex Partitions

Lemma

Suppose E' *is an* s-t *cut. Then there is a cut* $c(A, B)$ *such that* $c(A, B) \subseteq E'$.

Proof.

E' is an s-t cut implies no path from s to t in $(V, E - E')$.

1. Let A be set of all nodes reachable by s in $(V, E - E')$.
2. Since E' is a cut, $t \not\in A$.
3. Claim: $c(A, B) \subseteq E'$. If not, then some edge $(u, v) \in c(A, B)$ is not in E'. This implies, (i) $v \not\in A$, (ii) v will be reachable by s and should be in A. A contradiction. □

Corollary

Every minimal s-t *cut* E' *is a cut of the form* $c(A, B)$.

Alternate notation for cuts

Other common notation for cuts:

Undirected graphs: $G = (V, E)$ and $A \subset V$. $\delta_G(A)$ or $\delta(A)$ is set of edges with one end point in A and the other end point in $V \setminus A$.

Directed graphs: $G = (V, E)$ and $A \subset V$.
Edges going out of A

$$\delta_G^+(A) = \{(u, v) \in E \mid u \in A, v \in V \setminus A\}$$

Edges coming into A

$$\delta_G^-(A) = \{(u, v) \in E \mid u \in V \setminus A, v \in A\}$$

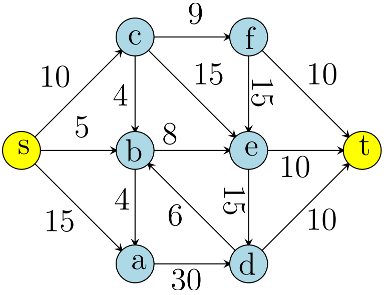
Minimum Cut

Definition

Given a flow network an ***s*-*t*** **minimum** cut is a cut ***E′*** of smallest capacity amongst all ***s*-*t*** cuts.

The minimum cut in the network flow depicted is:

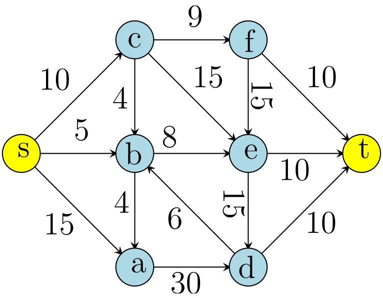
- **(A)** 10
- **(B)** 18
- **(C)** 28
- **(D)** 30
- **(E)** 48.
- **(F)** No minimum cut, no cry.



Minimum Cut

Definition

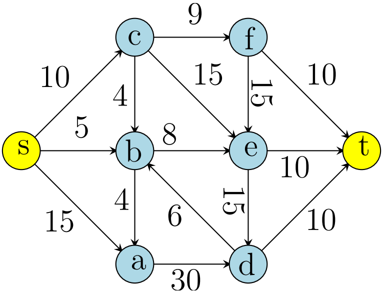
Given a flow network an ***s*-*t*** **minimum** cut is a cut ***E′*** of smallest capacity amongst all ***s*-*t*** cuts.



Minimum Cut

Definition

Given a flow network an s-t **minimum** cut is a cut E' of smallest capacity amongst all s-t cuts.



Observation: exponential number of s-t cuts and no "easy" algorithm to find a minimum cut.

The Minimum-Cut Problem

Problem

Input A flow network G

Goal Find the capacity of a *minimum* s-t cut

Flows and Cuts

Lemma

For any s-t cut E', **maximum** *s-t flow $\leq$ capacity of E'.*

Proof.

Formal proof easier with path based definition of flow.
Suppose $f : \mathcal{P} \rightarrow \mathbb{R}^{\geq 0}$ is a max-flow.

Flows and Cuts

Lemma

For any s-t cut E', **maximum** *s-t flow $\leq$ capacity of E'.*

Proof.

Formal proof easier with path based definition of flow.
Suppose $f : \mathcal{P} \rightarrow \mathbb{R}^{\geq 0}$ is a max-flow.

Every path $p \in \mathcal{P}$ contains an edge $e \in E'$. Why?

Flows and Cuts

Lemma

For any s-t cut E', **maximum** *s-t flow $\leq$ capacity of E'.*

Proof.

Formal proof easier with path based definition of flow.
Suppose $f : \mathcal{P} \rightarrow \mathbb{R}^{\geq 0}$ is a max-flow.

Every path $p \in \mathcal{P}$ contains an edge $e \in E'$. Why?
Assign each path $p \in \mathcal{P}$ to exactly one edge $e \in E'$.
Let $\mathcal{P}_e$ be paths assigned to $e \in E'$.

Flows and Cuts

Lemma

For any s-t *cut* E', **maximum** s-t *flow* $\leq$ *capacity of* E'.

Proof.

Formal proof easier with path based definition of flow.
Suppose $f : \mathcal{P} \rightarrow \mathbb{R}^{\geq 0}$ is a max-flow.

Every path $p \in \mathcal{P}$ contains an edge $e \in E'$. Why?
Assign each path $p \in \mathcal{P}$ to exactly one edge $e \in E'$.
Let $\mathcal{P}_e$ be paths assigned to $e \in E'$. Then

$$v(f) = \sum_{p \in \mathcal{P}} f(p) = \sum_{e \in E'} \sum_{p \in \mathcal{P}_e} f(p) \leq \sum_{e \in E'} c(e).$$

□

Flows and Cuts

Lemma

For any s-t *cut* E', **maximum** s-t *flow* $\leq$ *capacity of* E'.

Corollary

Maximum s-t *flow* $\leq$ *minimum* s-t *cut.*

Flows and Cuts

Lemma

*$\exists$ an **s-t** cut **E'**, such that **maximum** flow $=$ capacity of **E'**.*

Proof.

Flows and Cuts

Lemma

$\exists$ an s-t cut E', such that **maximum** *flow $=$ capacity of E'.*

Proof.

Intuition: Let f be a maximum *edge* flow. Construct graph G' with edge capacities to $c'(e) = c(e) - f(e)$. Remove edges with $c'(e) = 0$.

Flows and Cuts

Lemma

*$\exists$ an s-t cut E', such that **maximum** flow $=$ capacity of E'.*

Proof.

Intuition: Let f be a maximum *edge* flow. Construct graph G' with edge capacities to $c'(e) = c(e) - f(e)$. Remove edges with $c'(e) = 0$.

Is there an $s \to t$ path in G'?

Flows and Cuts

Lemma

$\exists$ an s-t cut E', such that **maximum** *flow $=$ capacity of E'.*

Proof.

Intuition: Let f be a maximum *edge* flow. Construct graph G' with edge capacities to $c'(e) = c(e) - f(e)$. Remove edges with $c'(e) = 0$.

Is there an $s \to t$ path in G'?

Exercise. □

Max-Flow Min-Cut Theorem

Theorem

In any flow network the maximum s-t flow is equal to the minimum s-t cut.

Max-Flow Min-Cut Theorem

Theorem

In any flow network the maximum s-t *flow is equal to the minimum* s-t *cut.*

Can compute minimum-cut from maximum flow and vice-versa!

Proof coming shortly.

Max-Flow Min-Cut Theorem

Theorem

In any flow network the maximum s-t flow is equal to the minimum s-t cut.

Can compute minimum-cut from maximum flow and vice-versa!

Proof coming shortly.

Many applications:

1. optimization
2. graph theory
3. combinatorics

The Maximum-Flow Problem

Problem

Input A network G with capacity c and source s and sink t.

Goal Find flow of **maximum** value from s to t.

The Maximum-Flow Problem

Problem

Input A network G with capacity c and source s and sink t.

Goal Find flow of **maximum** value from s to t.

Exercise: Given G, s, t as above, show that one can remove all edges into s and all edges out of t without affecting the flow value between s and t.