CS 473: Fundamental Algorithms, Spring 2013

Network Flows

Lecture 16
March 15, 2013

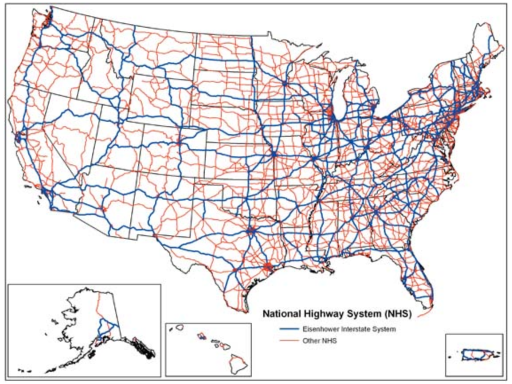
Everything flows

Panta rei – everything flows (literally).
Heraclitus (535–475 BC)

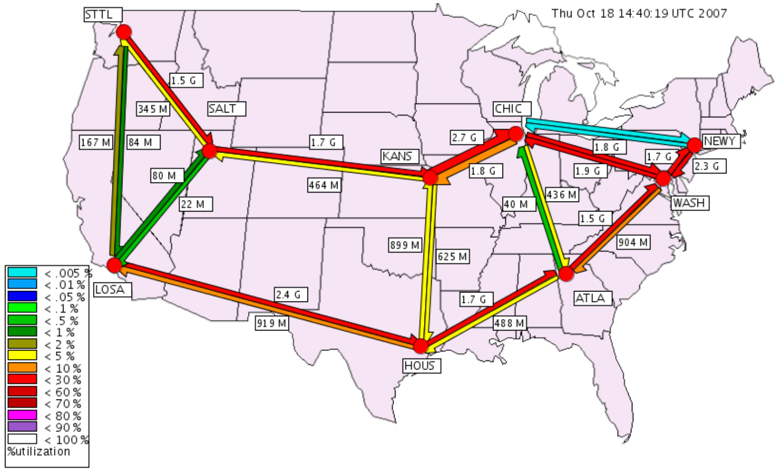
Part I

Network Flows: Introduction and Setup

Transportation/Road Network



Internet Backbone Network



Common Features of Flow Networks

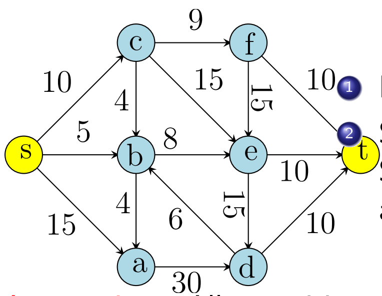
1. **Network** represented by a (directed) *graph* $\mathbf{G = (V, E)}$.
2. Each edge $\mathbf{e}$ has a **capacity** $\mathbf{c(e) \geq 0}$ that limits amount of *traffic* on $\mathbf{e}$.
3. *Source(s)* of traffic/data.
4. *Sink(s)* of traffic/data.
5. Traffic *flows* from sources to sinks.
6. Traffic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

1. Single source $\mathbf{s}$ and single sink $\mathbf{t}$.
2. Every other node $\mathbf{v}$ is an **internal** node.
3. Flow originates at $\mathbf{s}$ and terminates at $\mathbf{t}$.



1. Each edge $\mathbf{e}$ has a capacity $\mathbf{c(e) \geq 0}$.
2. Sometimes assume:
   Source $\mathbf{s \in V}$ has no incoming edges, and sink $\mathbf{t \in V}$ 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 $\mathbf{f : E \rightarrow \mathbb{R}^{\geq 0}}$ s.t.

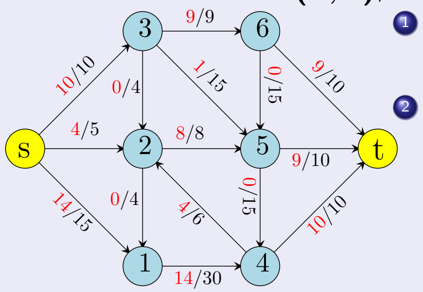


Figure: Flow with value.

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

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

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

Flow...

Conservation of flow law is also known as **Kirchhoff's law**.

More Definitions and Notation

Notation

1. The inflow into a vertex $\mathbf{v}$ is $\mathbf{f^{in}(v)} = \sum_{\mathbf{e} \text{ into } \mathbf{v}} \mathbf{f(e)}$ and the outflow is $\mathbf{f^{out}(v)} = \sum_{\mathbf{e} \text{ out of } \mathbf{v}} \mathbf{f(e)}$
2. For a set of vertices $\mathbf{A}$, $\mathbf{f^{in}(A)} = \sum_{\mathbf{e} \text{ into } \mathbf{A}} \mathbf{f(e)}$. Outflow $\mathbf{f^{out}(A)}$ is defined analogously

Definition

For a network $\mathbf{G = (V, E)}$ with source $\mathbf{s}$, the **value** of flow $\mathbf{f}$ is defined as $\mathbf{v(f) = f^{out}(s) - f^{in}(s)}$.

A Path Based Definition of Flow

Intuition: Flow goes from source $\mathbf{s}$ to sink $\mathbf{t}$ along a path.

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

Definition (Flow by paths.)

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

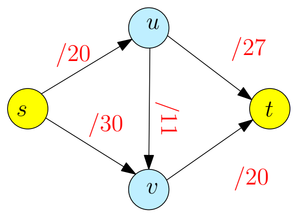
1. Capacity Constraint: For each edge $\mathbf{e}$, total flow on $\mathbf{e}$ is $\leq \mathbf{c(e)}$.

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

2. Conservation Constraint: No need! Automatic.

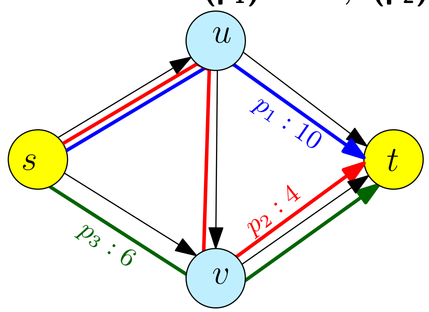
Value of flow: $\sum_{\mathbf{p} \in \mathcal{P}} \mathbf{f(p)}$.

Example



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

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



Path based flow implies edge based flow

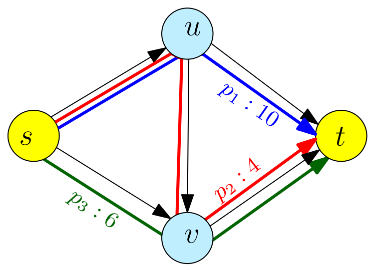
Lemma

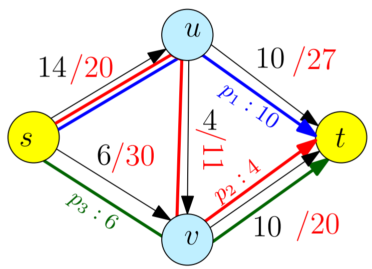
Given a path based flow $f : \mathcal{P} \to \mathbb{R}^{\geq 0}$ *there is an edge based flow* $f' : E \to \mathbb{R}^{\geq 0}$ *of the same value.*

Proof.

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

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 $f' : E \to \mathbb{R}^{\geq 0}$*, 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* $|V| = n$*. Given* f'*, the path based flow can be computed in* $O(mn)$ *time.*

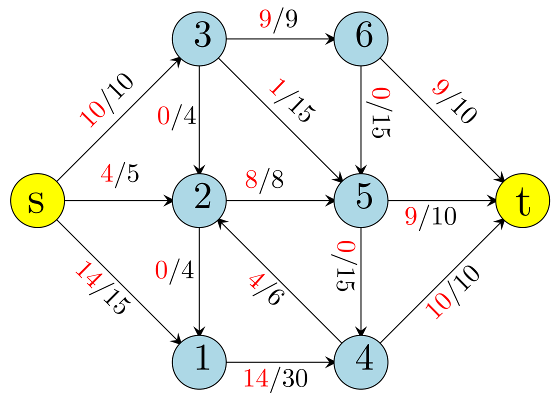
Flow Decomposition

Edge based flow to Path based Flow

Proof Idea.

1. Remove all edges with $\mathbf{f'(e) = 0}$.
2. Find a path $\mathbf{p}$ from $\mathbf{s}$ to $\mathbf{t}$.
3. Assign $\mathbf{f(p)}$ to be $\mathbf{min}_{\mathbf{e \in p}} \mathbf{f'(e)}$.
4. Reduce $\mathbf{f'(e)}$ for all $\mathbf{e} \in \mathbf{p}$ by $\mathbf{f(p)}$.
5. Repeat until no path from $\mathbf{s}$ to $\mathbf{t}$.
6. In each iteration at least on edge has flow reduced to zero.
7. Hence, at most $\mathbf{m}$ iterations. Can be implemented in $\mathbf{O(m(m+n))}$ time. $\mathbf{O(mn)}$ time requires care. □

Example



Edge vs Path based Definitions of Flow

Edge based flows:

1. *compact* representation, only $\mathbf{m}$ 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.

The Maximum-Flow Problem

Problem

Input A network $\mathbf{G}$ with capacity $\mathbf{c}$ and source $\mathbf{s}$ and sink $\mathbf{t}$.

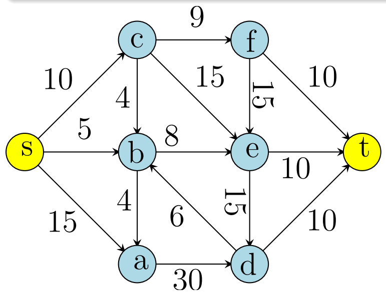
Goal Find flow of *maximum* value.

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

Cuts

Definition (s-t cut)

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

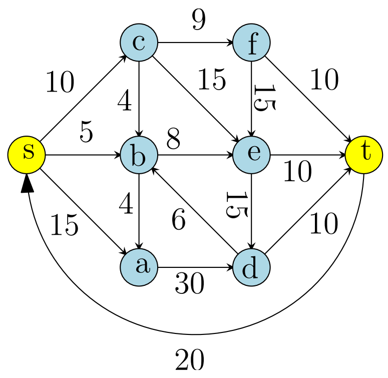


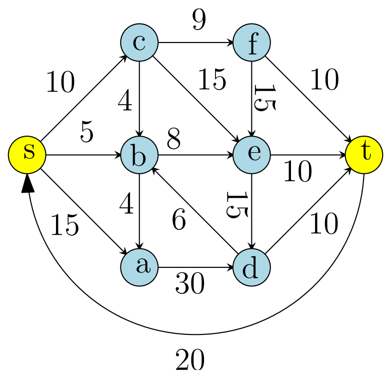
Caution:

1. Cut may leave $\mathbf{t} \rightarrow \mathbf{s}$ paths!
2. There might be many **s-t** cuts.

s — t cuts

A death by a thousand cuts

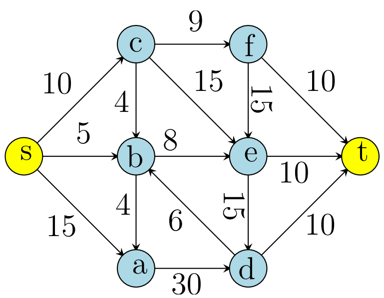




Minimal Cut

Definition (Minimal s-t cut.)

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



Observation: given a cut $\mathbf{E}'$, can check efficiently whether $\mathbf{E}'$ is a minimal cut or not. How?

Cuts as Vertex Partitions

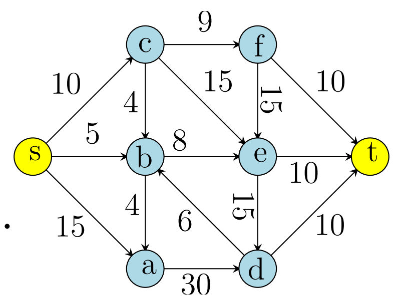
Let $\mathbf{A} \subset \mathbf{V}$ such that

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

The **cut** $(\mathbf{A}, \mathbf{B})$ is the set of edges

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

Cut $(\mathbf{A}, \mathbf{B})$ is set of edges leaving $\mathbf{A}$.



Claim

$(\mathbf{A}, \mathbf{B})$ *is an* **s-t** *cut.*

Proof.

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

Cuts as Vertex Partitions

Lemma

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

Proof.

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

1. Let $\mathbf{A}$ be set of all nodes reachable by $\mathbf{s}$ in $\mathbf{(V, E - E')}$.
2. Since $\mathbf{E'}$ is a cut, $\mathbf{t} \notin \mathbf{A}$.
3. $\mathbf{(A, B) \subseteq E'}$. Why?If some edge $\mathbf{(u, v)} \in \mathbf{(A, B)}$ is not in $\mathbf{E'}$ then $\mathbf{v}$ will be reachable by $\mathbf{s}$ and should be in $\mathbf{A}$, hence a contradiction. □

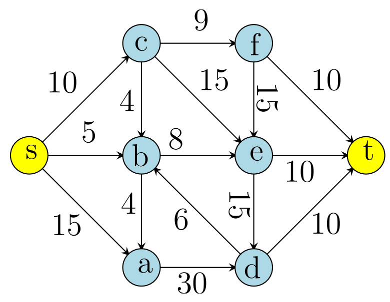
Corollary

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

Minimum Cut

Definition

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



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

The Minimum-Cut Problem

Problem

Input A flow network $\mathbf{G}$

Goal Find the capacity of a *minimum* $\mathbf{s}$-$\mathbf{t}$ cut

Flows and Cuts

Lemma

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

Proof.

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

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

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

□

Flows and Cuts

Lemma

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

Corollary

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

Max-Flow Min-Cut Theorem

Theorem

In any flow network the maximum $\mathbf{s}$-$\mathbf{t}$ *flow is equal to the minimum* $\mathbf{s}$-$\mathbf{t}$ *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 $\mathbf{G}$ with capacity $\mathbf{c}$ and source $\mathbf{s}$ and sink $\mathbf{t}$.
Goal Find flow of *maximum* value from $\mathbf{s}$ to $\mathbf{t}$.

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