Chapter 16

Network Flows

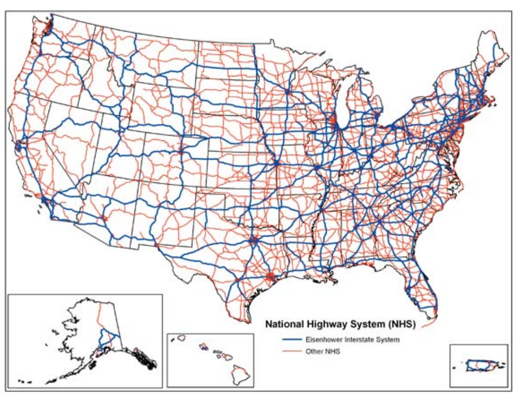
CS 473: Fundamental Algorithms, Spring 2011
March 17, 2011

16.0.0.1 Everything flows

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

16.1 Network Flows: Introduction and Setup

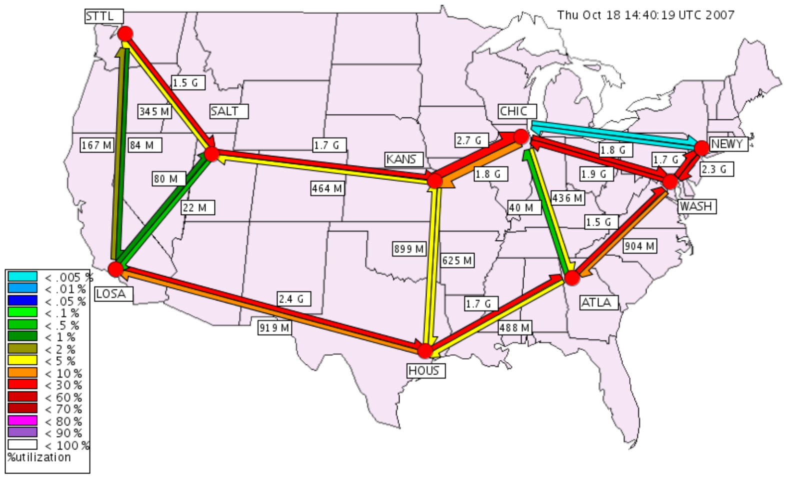
16.1.0.2 Transportation/Road Network

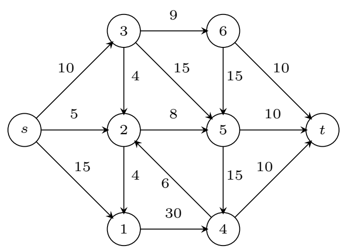


16.1.0.3 Internet Backbone Network

16.1.0.4 Common Features of Flow Networks

- *Network* represented by a (directed) *graph* $G = (V, E)$
- Each edge e has a *capacity* $c(e) \geq 0$ that limits amount of *traffic* on e





- *Source(s)* of traffic/data
- *Sink(s)* of traffic/data
- Traffic *flows* from sources to sinks
- Traffic is *switched/interchanged* at nodes

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

16.1.0.5 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 $c(e) \geq 0$
- Some times it is convenient to assume that source $s \in V$ has no incoming edges and sink $t \in V$ has no outgoing edges

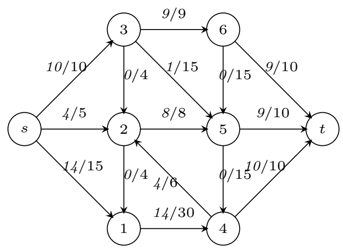


Figure 16.1: Flow with value

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

16.1.0.6 Definition of Flow

Two ways to define flows:

- edge based
- path based

They are essentially equivalent but have different uses.

Edge based definition is more compact.

16.1.0.7 Edge Based Definition of Flow

Definition 16.1.1 *A* **flow** *in a network* $G = (V, E)$*, is a function* $f : E \to \mathbb{R}^{\geq 0}$ *such that*

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

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

- *Value of flow: (total flow out of source)* $-$ *(total flow in to source)*

16.1.0.8 Flow...

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

16.1.0.9 More Definitions and Notation

Notation

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

Definition 16.1.2 *For a network $G = (V, E)$ with source s, the* **value** *of flow f is defined as $v(f) = f^{\text{out}}(s) - f^{\text{in}}(s)$*

16.1.0.10 A Path Based Definition of Flow

Intuition: flow goes from source s to sink t along a path.
$\mathcal{P}$: set of all paths from s to t. $|\mathcal{P}|$ can be *exponential* in n.

Definition 16.1.3 *A flow in a network $G = (V, E)$, is a function $f : \mathcal{P} \to \mathbb{R}^{\geq 0}$ such that*

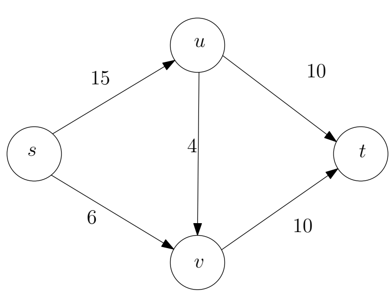
- *Capacity Constraint: For each edge e, total flow on e is $\leq c(e)$.*

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

- *Conservation Constraint: No need! Automatic.*

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

16.1.0.11 Example



$\mathcal{P} = \{p_1, p_2, p_3\}$

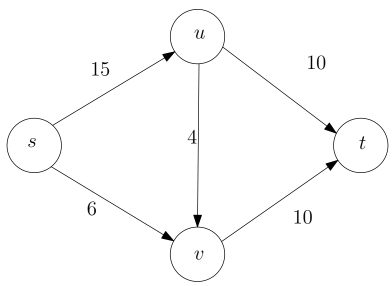
$p_1 : s \to u \to t$
$p_2 : s \to u \to v \to t$
$p_3 : s \to v \to t$
$f(p_1) = 10, f(p_2) = 4, f(p_3) = 6$

16.1.0.12 Path based flow implies Edge based flow

Lemma 16.1.4 *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 ∎

16.1.0.13 Example



$\mathcal{P} = \{p_1, p_2, p_3\}$

$p_1 : s \to u \to t$
$p_2 : s \to u \to v \to t$
$p_3 : s \to v \to t$
$f(p_1) = 10, f(p_2) = 4, f(p_3) = 6$

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

16.1.1 Flow Decomposition

16.1.1.1 Edge based flow to Path based Flow

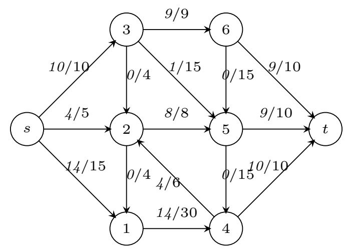
Lemma 16.1.5 *Given an edge based flow $f' : E \to \mathbb{R}^{\geq 0}$, there is a path based flow $f : \mathcal{P} \to \mathbb{R}^{\geq 0}$ of same value. Moreover, f assigns non-negative flow to at most m paths where $|E| = m$ and $|V| = n$. Given f', the path based flow can be computed in $O(mn)$ time.*

16.1.2 Flow Decomposition

16.1.2.1 Edge based flow to Path based Flow

Proof:[Proof Idea]

- remove all edges with $f'(e) = 0$



- find a path p from s to t
- assign $f(p)$ to be $\min_{e \in p} f'(e)$
- reduce $f'(e)$ for all $e \in p$ by $f(p)$
- repeat until no path from s to t
- in each iteration at least on 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.

■

16.1.2.2 Example

16.1.2.3 Edge vs Path based Definitions of Flow

Edge based flows:

- *compact* representation, only m values to be specified
- need to check flow conservation explicitly at each internal node

Path flows:

- in some applications, paths more natural
- not compact
- no need to check flow conservation constraints

Equivalence shows that we can go back and forth easily.

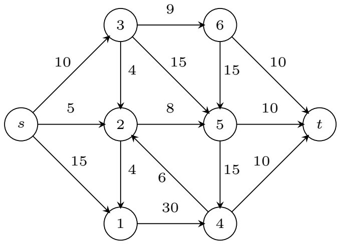
16.1.2.4 The Maximum-Flow Problem

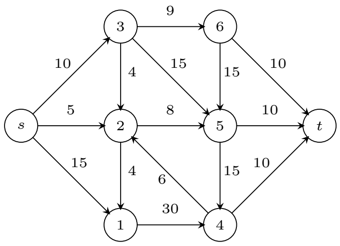
Problem

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?





16.1.2.5 Cuts

Definition 16.1.6 *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* $\sum_{e \in E'} c(e)$*.*

Caution: cut may leave $t \to s$ paths!

16.1.2.6 Minimal Cut

Definition 16.1.7 *Given a flow network an* s-t*,* E' *is a* **minimal cut** *if for all* $e \in E'$*,* $E' - \{e\}$ *is not a cut.*

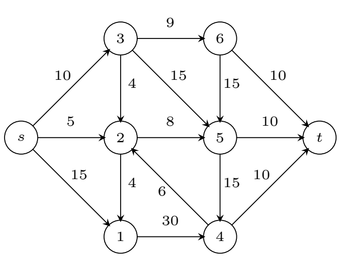
Observation: given a cut E', can check efficiently whether E' is a minimal cut or not. How?

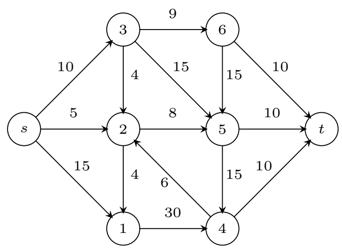
16.1.2.7 Cuts as Vertex Partitions

Let $A \subset V$ such that

- $s \in A$, $t \notin A$
- $B = V - A$ and hence $t \in B$

Define *cut* $(A, B) = \{(u, v) \in E \mid u \in A, v \in B\}$: edges leaving A





Claim 16.1.8 *(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 (A, B). ∎

16.1.2.8 Cuts as Vertex Partitions

Lemma 16.1.9 *Suppose E' is an s-t cut. Then there is a cut (A, B) such that $(A, B) \subseteq E'$.*

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

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

∎

Corollary 16.1.10 *Every* minimal *s-t cut E' is a cut of the form (A, B).*

16.1.2.9 Minimum Cut

Definition 16.1.11 *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.

16.1.2.10 The Minimum-Cut Problem

Problem

Input A flow network G

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