Announcements

• No homework this week!
Last Time

• Network Flows
Definitions

A *network* is a directed graph G with designated *source* and *sink* vertices s and t.

A *flow* is a subgraph of G so that for each vertex v other than s and t $d_{in}(v) = d_{out}(v)$.

The *size* of a flow is $d_{out}(s) - d_{in}(s)$. 
Problems

• Given a network G, what is the largest size of a flow in the network?
• How do we find such a flow?
• How can we show that there isn’t a larger flow?
Today

- Maxflow-Mincut
- Applications
Question: Maxflow

What is the largest flow size in this network?

A) 1
B) 2
C) 3
D) 4
E) 5
Creating Flow

• How do you find a flow?
  – Path from s to t

• Find more flow:
  – Add s-t paths

• Even better: these paths can “cancel” existing edges.
Augmenting Paths

**Definition:** Given a network $G$ and flow $F$ an *augmenting path* is an $s$-$t$ path that uses either edges of $G$ unused by $F$ in the forwards direction, or edges used by $F$ in the backwards direction.

**Lemma:** Given an augmenting path, you can add it to $F$ to get a path with 1 more unit of flow.
Question: Augmenting Path

Does this flow have an augmenting path?

A) Yes
B) No
When Can’t You Augment?
Cuts

**Definition:** A *cut* is a partition of the vertices into two sets $S$ and $T$, which contain $s$ and $t$, respectively.

The *size* of a cut is the total number of edges from vertices in $S$ to vertices in $T$. 
Question: Cut Size

What is the size of the cut below?

A) 1
B) 2
C) 3
D) 4
E) 5
Cuts and Flows

**Lemma (V 8.3.1):** For a network $G$ a flow $F$ and a cut $(S,T)$ it is the case that

$$\text{Size}(F) = \#\{\text{edges in } F \text{ from } S \text{ to } T\} - \#\{\text{edges in } F \text{ from } T \text{ to } S\}$$

**Remark:** This says that the number of people leaving the city, is the number crossing into the next state (assuming that's where they are headed).
Proof

Consider the sum over all $v$ in $S$ of $d_{out}(v)-d_{in}(v)$.

On the one hand this is 0 except for $v = s$, where it is $\text{Size}(F)$.

On the other hand, each edge contributes to one in degree and one out degree. This makes its total contribution 0 unless it crosses the cut. This gives 1 for each edge from $S$ to $T$ and -1 for each edge from $T$ to $S$. 
Note

If we take $T = \{t\}$, we find:

$$\text{Size}(F) = d_{\text{in}}(t) - d_{\text{out}}(t).$$

The total flow out of $s$ equals the total flow into $t$. 
Maxflow-Mincut

Theorem (V. 8.3.2): For any network $G$ the size of a maximum flow in $G$ is the same as the size of a minimum cut.
Idea

• A cut is a bottleneck. No flow can be more than any cut.

• Suppose that as much traffic is leaving the city as possible (maxflow). You try to escape in your car and find yourself blocked by traffic. The roads that are full describe a matching cut.
Maxflow $\leq$ Mincut

Let $F$ be a flow and $C$ be a cut.

By Lemma:

$$\text{Size}(F) = \#\{\text{edges in } F \text{ from } S \text{ to } T\} - \#\{\text{edges in } F \text{ from } T \text{ to } S\} \leq \text{Size}(C)$$

Any flow is smaller than any cut, so the maximum flow size is at most than the minimum cut size.
Maxflow ≥ Mincut

• Let F be a maximum flow.
• No augmenting paths.
• Let S be the set of vertices v you can reach from s using unused forward edges or used backwards edges.
• F uses all edges out of S, no edges into S.
• Lemma says: Size(F) = Number of edges out of S = Size(C).
• Maxflow ≥ Mincut.
Capacities

Everything still works if we allow multiple edges. This basically corresponds to some routes having a higher capacity than others (if there are two edges from $u$ to $v$, you can run two units of flow there). You can also generalize further by giving each edge a real numbered “capacity” for how much flow it can handle.
Konig’s Theorem can be thought of as a special case of Maxflow-Mincut.
Menger’s Theorem

• Maxflow-Mincut *is* the edge version of Menger’s Theorem. It says that the minimum number of edges you need to remove to disconnect s from t (the smallest cut size), is the maximum number of edge-disjoint paths (the maximum flow size).

• You can also use Maxflow-Mincut to prove the vertex Menger’s Theorem with extra work.