Flows and Cuts

Graph Theory (Fall 2011)
Rutgers University
Swastik Kopparty

1 Flows

We now study flows on graphs.

Definition 1 (Directed Graph). *A directed graph G is a tuple (V, E) where $E \subseteq V^2$. Here V is the set of vertices and E is the set of directed edges. If $(u, v) \in E$, we say that there is an edge in the graph G* from *u* to *v.*

Further, if $S, T \subseteq V$, we denote by $E(S, T)$ the set of edges from some vertex in S to some vertex in T:

$$E(S,T) = \{(u,v) \in E \mid u \in S, v \in T\}.$$

Let $G = (V, E)$ be a directed graph, and let $s, t \in V$. A **flow** in G from s to t is an allocation of "current" to each edge of G, such that the net current entering any vertex $v \neq s, t$ equals 0. Formally, it is given by a function $f : E \to \mathbb{R}$ such that $f(e) \geq 0$ for each $e \in E$, and such that for each $v \neq s, t$:

$$\mathsf{NetFlow}(v) = 0,$$

where

$$\mathsf{NetFlow}(v) = \sum_{e \in E : e = (v, \cdot)} f(e) - \sum_{e \in E : e = (\cdot, v)} f(e)$$

(Notice that the expression for $\mathsf{NetFlow}(v)$ equals the total flow on edges out of v minus the total flow on edges into v).

We have the following simple lemma.

Lemma 2. *Let $S \subseteq V$. Let $\overline{S}$ denote $V \setminus S$. Then*

$$\sum_{v \in S} \mathsf{NetFlow}(v) = \sum_{e \in E(S, \overline{S})} f(e) - \sum_{e \in E(\overline{S}, S)} f(e).$$

Proof. Immediate from the definitions. □

Applying this to $S = V$, we get that $\mathsf{NetFlow}(s) + \mathsf{NetFlow}(t) = 0$, and so $\mathsf{NetFlow}(s) = -\mathsf{NetFlow}(t)$. We call this quantity the **value of the flow** f from s to t.

$$\mathsf{Value}(f) = \mathsf{NetFlow}(s).$$

2 Flows under capacity constraints

Let $G = (V, E)$ be a directed graph. Let $c : E \to \mathbb{R}$ be a function with $c(e) \geq 0$ for each $e \in E$. We will call $c(e)$ the **capacity** of the edge E.

Let $s, t \in V$. A flow from s to t satisfying the capacities c is a flow $f : E \to \mathbb{R}$ such that $f(e) \leq c(e)$ for each edge $e \in E$.

3 Cuts

A cut in a graph G is simply a partition of the vertex set into two nonempty sets. If s, t are two vertices of G, an (s, t)-cut is a partition of the vertex set into two nonempty sets such that s is in one set and t is in the other.

Every cut in the graph G gives a simple upper bound on the maximum possible value of a flow satisfying given capacity constraints.

Lemma 3. *For every flow f satisfying the capacities c, and for every $S \subseteq V$, such that $s \in S$ and $t \in \overline{S}$,*

$$\mathsf{Value}(f) \leq \mathsf{Capacity}(S, \overline{S}).$$

Proof. Using Lemma 2, we have:

$$\begin{aligned}
\mathsf{NetFlow}(s) &= \sum_{v \in S} \mathsf{NetFlow}(v) \\
&= \sum_{e \in E(S,\overline{S})} f(e) - \sum_{e \in E(\overline{S},S)} f(e) \\
&\leq \sum_{e \in E(S,\overline{S})} c(e) - 0 \\
&= \mathsf{Capacity}(S, \overline{S}).
\end{aligned}$$

Here we used the fact that $0 \leq f(e) \leq c(e)$ to derive the "$\leq$" step. □

4 Max-Flow / Min-Cut

In particular, the previous lemma implies that:

$$\max_f \mathsf{Value}(f) \leq \min_S \mathsf{Capacity}(S, \overline{S}),$$

where f varies over flows satisfying c, and S varies over (s, t)-cuts.

The max-flow-min-cut theorem says that these quantities are in fact equal.

Theorem 4 (Max-Flow/Min-Cut)**.** *Let G be a directed graph, and let c be a capacity function on the edges of G. Then:*

$$\max_f \mathsf{Value}(f) = \min_S \mathsf{Capacity}(S, \overline{S}),$$

where f varies over flows satisfying c, and S varies over (s, t)-cuts.

The proof of this theorem will also lead to a simple, quick algorithm to find the maximum flow, as well as a cut with capacity = value (thus showing that the flow is indeed max).

Proof. Let f be a flow satisfying c that maximizes $\mathsf{Value}(f)$. We will show how to use f to find a cut $(S, \overline{S})$ whose capacity is $\mathsf{Value}(f)$. Note that we crucially need to use the fact that f is a **maximum** flow, this argument cannot work with any old flow.

The cut we are looking for is a "bottleneck" for f, across which no extra flow can be sent. Thus it makes sense to look for the set S of vertices that can "receive" more flow. This is what we achieve in the following procedure:

- Initialize $S = \{s\}$.
- Repeatedly do the following until S grows no further:
  - If there exists $u \in S$ and $v \in V \setminus S$ are such that $(u, v) \in E$ and either $f_{(u,v)} < c_{(u,v)}$ or $f_{(v,u)} > 0$, then include v in S.
- Output the cut $(S, \overline{S})$.

We first claim that $(S, \overline{S})$ is an (s, t)-cut. If not, then $t \in S$. Consider the vertices $v_0 = s, v_1, \ldots, v_k = t$ which led to t being included in S (i.e., for each i, either $f_{(v_i, v_{i+1})} < c_{(v_i, v_{i+1})}$ or $f_{(v_{i+1}, v_i)} > 0$). Then we can modify f to get a flow with even higher value as follows: for a suitably small ϵ, either increase $f_{(v_i, v_{i+1})}$ by ϵ or decrease $f_{(v_{i+1}, v_i)}$ by ϵ (by choice of the v_i, at least one of these modifications will be possible without violating the capacity or nonnegativity of the flow). This increases $\mathsf{Value}(f)$ by ϵ, contradicting the maximality of f. Thus $(S, \overline{S})$ is an (s, t)-cut.

Now we verify that $\mathsf{Capacity}(S, \overline{S})$ equals $\mathsf{Value}(f)$.

We have:

$$\begin{aligned}
\mathsf{Capacity}(S, \overline{S}) &= \sum_{e \in E(S, \overline{S})} c(e) \\
&= \sum_{e \in E(S, \overline{S})} f(e) \quad \text{by definition of } S \\
&= \sum_{e \in E(S, \overline{S})} f(e) - \sum_{e \in E(\overline{S}, S)} f(e) \quad \text{by definition of } S \text{ again, } f(e) = 0 \text{ for each } e \in E(\overline{S}, S) \\
&= \mathsf{Value}(f).
\end{aligned}$$

This completes the proof. □

This leads to the following efficient algorithm to find a maximum flow.

- Start with $f = 0$.
- Keep doing the following until the max flow is found:
  - Construct the set S following the algorithm in the proof of Theorem 4.

- If $(S, \overline{S})$ is an (s,t)-cut, then we are done; f is the desired max flow, and $(S, \overline{S})$ is a cut which proves that f is maximum.
- Otherwise, as in the proof of Theorem 4, we can modify f and increase $\mathsf{Value}(f)$.

By inspecting the proof of Theorem 4 (and in particular paying attention to the amount by which $\mathsf{Value}(f)$ can be increased), it is easy to check that if all the capacities $c(e)$ are integers, then this algorithm will find the maximum flow with at most $\sum_e c(e)$ modifications of f.

Another important corollary of this algorithm is the integrality of the max flow:

Corollary 5. *If $c(e)$ is an integer for each edge $e \in E$, then there is a maximum flow f where $f(e)$ is an integer for each edge $e \in E$.*