CSCI 3100

Flows, pt 2
More formally:

Given a directed graph with two designated vertices, $s$ and $t$.

Each edge is given a capacity $c(e)$.

Assume:

- No edges enter $s$.
- No edges leave $t$.
- Every $c(e) \in \mathbb{Z}$ in integers.

Goal:

Max flow: find the most we can ship from $s$ to $t$ without exceeding any capacity.

Min cut: find smallest set of edges to delete in order to disconnect $s$ and $t$. 
Thm: (Ford-Fulkerson '54, Elias-Feinstein-Shannon '56) The max flow value = min cut value

Last time:
any flow \leq\ any cut

why?

\text{Value}(f) = \sum_{e\text{ out of }S} f(e)

\text{Cost}((S,T)) = \sum_{e\text{ out of }S} c(e)
Today:
- An algorithm for max flow
  (continued from last time)
- The proof of correctness will prove F-F theorem
Keys:

Residual network $G_f$:

A flow $f$ in a weighted graph $G$ and the corresponding residual graph $G_f$.

Augmenting a path:

An augmenting path in $G_f$ with value $F = 5$ and the augmented flow $f'$. 
Algorithm: Ford–Fulkerson (1956)

**MAX Flow (G):**

Let $f(e) = 0$ initially

Construct $G_f = G$

While there is s-t path in $G_f$:

Let $P$ be a simple s-t path

$f' \leftarrow \text{augment } (f, P)$

$f \leftarrow f'$

update $G_f$

return $f$

Last time:

Lemma: At each stage, flow & residual values are integers.
Lemma: In each iteration,
value \( f' \) > value \( f \).
In each iteration, value improves.

pf:
found a path \( P \) in \( G_f \).
This \( P \) had some bottleneck edge.
By prior lemma, that edge was an integer.
It was positive.
value \( f' \) increased by this bottleneck amount:

\[ \text{Value}(f') = \text{value}(f) + \text{bottleneck}(P) \]
Cor: The while loop halts after \( O(\text{value}(f^*)) \) iterations, where \( f^* \) is a maximum flow (since gets larger by at least 1 each stay an integer).

So: Running time is:

\[ O(1f^*, m) \]

\[ O(Cm) \]

\[ C = \sum_{\text{all capacities}} \]

\[ O(C'm) \]

\[ \sum_{\text{edges out of} s} \]
Note: This is the best we can do!

Worst path:

To do better, need to consider how to choose a "good" augmenting path.
Thm: The F-F algorithm terminates with a maximum flow.

To prove this, we'll use cuts!

Fact: For any S-T cut, \[ \text{value}(f) = f^{out}(s) - f^{in}(s) \]

pf:
\[
\text{value}(f) = f^{out}(s)
\]

Know \( f^{in}(s) = 0 \)

since \( s \) has no incoming edges

\[ \Rightarrow \text{value}(f) = f^{out}(s) - f^{in}(s) \]
for all $v \in S$ other than $s$

\[ f^n(v) = f^{out}(v) \]

\[ \Rightarrow f^{out}(v) - f^n(v) = 0 \]

\[ \Rightarrow v(f) = \sum_{v \in S} (f^{out}(v) - f^n(v)) \]

Rewrite: Consider edges

if $e = uv$:

$u, v \in S$:

know appears twice in sum above- once pos, once neg.

$u, v \notin S$:

not in sum

$u \in S, v \notin S$:

appears as $+f(e)$

$u \notin S, v \in S$:

appears as $-f(e)$
Goal: $v(f) = f^{out}(S) - f^{in}(S)$

have:

$v(f) = \sum_{v \in S} (f^{out}(v) - f^{in}(v))$

$= \sum_{e \in \text{out of } S} f(e) - \sum_{e \in \text{into } S} f(e)$

$= f^{out}(S) - f^{in}(S)$
Thm: Let $f$ be any $s$-$t$ flow in $G$, and let $T$ be any $s$-$t$ cut.

Value ($f$) $\leq$ Cost ($S,T$)

\[
\text{value} (f) = f^\text{out} (S) - f^\text{in} (S) \leq f^\text{out} (S) = \sum_{e \text{ out of } S} f(e) \leq \sum_{e \text{ out of } S} C(e) = \text{cost}(S,T)
\]
Thm: If \( f \) is a \( s-t \) flow with no \( s-t \) path in \( G_f \), then \( \exists \) an \( s-t \) cut \( (S^*, T^*) \) in \( G \) with \( \text{cost}(S^*, T^*) = \text{value}(f) \).

Cor: \( \text{max flow} = \text{min cut} \)

Proof: Use \( G_f \):
pf (cont.)
Faster versions
- Depend upon choosing good augmenting paths!

Ex: Edmonds - Karp:
choose largest bottleneck edge
\( \Rightarrow \mathcal{O}(E^2 \log E \log K^*) \)

Ex: shortest augmenting path
\( \Rightarrow \mathcal{O}(VE^2) \)
### Table: Maximum-Flow Algorithms

<table>
<thead>
<tr>
<th>Technique</th>
<th>Direct</th>
<th>With dynamic trees</th>
<th>Sources</th>
</tr>
</thead>
<tbody>
<tr>
<td>Blocking flow</td>
<td>$O(V^2E)$</td>
<td>$O(VE \log V)$</td>
<td>[Dinitis; Sleator and Tarjan]</td>
</tr>
<tr>
<td>Network simplex</td>
<td>$O(V^2E)$</td>
<td>$O(VE \log V)$</td>
<td>[Dantzig; Goldfarb and Hao; Goldberg, Grigoriadis, and Tarjan]</td>
</tr>
<tr>
<td>Push-relabel (generic)</td>
<td>$O(V^2E)$</td>
<td>—</td>
<td>[Goldberg and Tarjan]</td>
</tr>
<tr>
<td>Push-relabel (FIFO)</td>
<td>$O(V^3)$</td>
<td>$O(V^2 \log(V^2/E))$</td>
<td>[Goldberg and Tarjan]</td>
</tr>
<tr>
<td>Push-relabel (highest label)</td>
<td>$O(V^2 \sqrt{E})$</td>
<td>—</td>
<td>[Cheriyan and Maheshwari; Tunçel]</td>
</tr>
<tr>
<td>Pseudoflow</td>
<td>$O(V^2E)$</td>
<td>$O(VE \log V)$</td>
<td>[Hochbaum]</td>
</tr>
<tr>
<td>Compact abundance graphs</td>
<td>$O(VE)$</td>
<td>—</td>
<td>[Orlin 2012]</td>
</tr>
</tbody>
</table>

Several purely combinatorial maximum-flow algorithms and their running times.