CSCI 3100

Flows, pt 2

Announcements

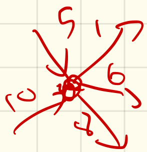
More formally:

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

Each edge is given a capacity $c(e)$.
↳ maximum amount that can be sent along it.

Assume:

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



Goal: $f : E \to \mathbb{Z}$ st
① $0 \le f(e) \le c(e)$
② $\sum_{e \text{ into } v} f(e) = \sum_{e \text{ out of } v} f(e)$

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

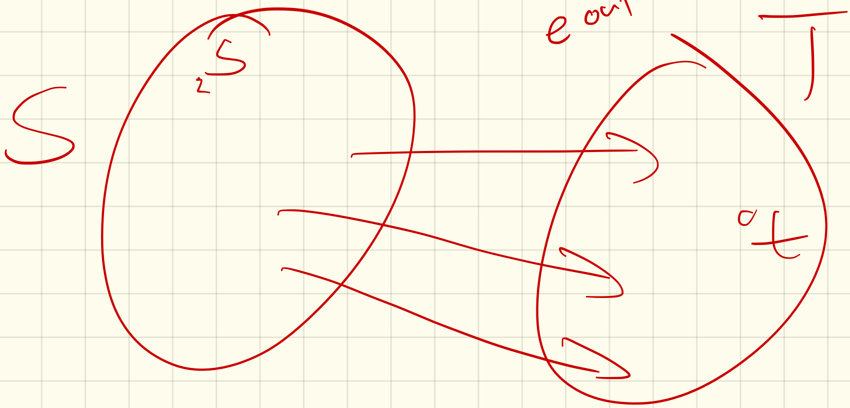
Min cut: find smallest set of edges to delete in order to disconnect s + t

Thm: (Ford - Fulkerson '54, Elias-Feinstein-Shannon '56)
The max flow value
= min cut value

Last time:
any flow ≤ any cut

Why?

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



$$\text{Cost}(S,T) = \sum_{e \text{ out of } S} c(e)$$

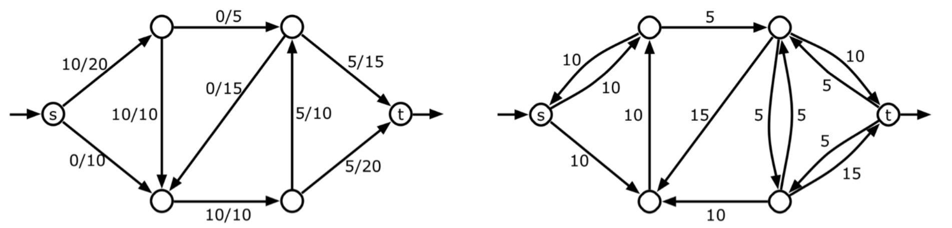
Today:

- An algorithm for max flow
  (continued from last time)

- The pf of correctness will prove F-F thm

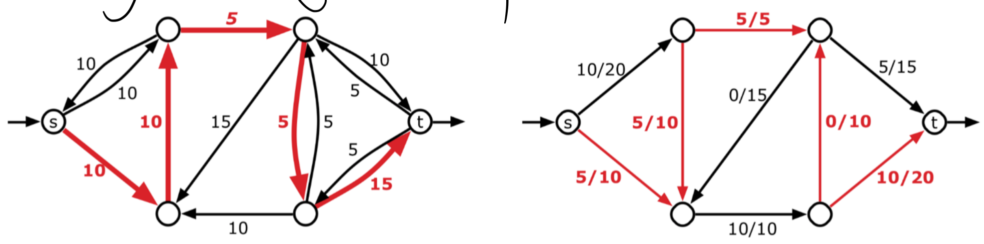
Keys:

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 $\forall e$

Construct G_f $= G$

$O(f^*)$ While there is s-t path in G_f:

Let P be a simple s-t path

$f' \leftarrow$ augment (f, P)

$f \leftarrow f'$

update G_f

call DFS

$O(m+n)$

return f

Last time:

Lemma: At each stage, flow & residual values are integers.

(f' is after augmenting)

Lemma: In each iteration,

$$\text{value}(f') > \text{value}(f).$$

In each iteration, value improves

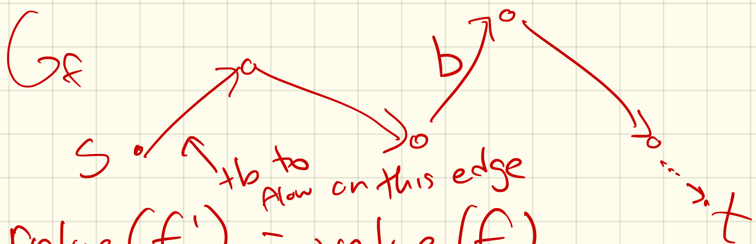
pf:

found a path P in G_f

This P had some bottleneck edge.

By prior lemma, that edge was an integer & 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 & stays an integer)

So: Running time is:

$$O(|f^*| \cdot m)$$

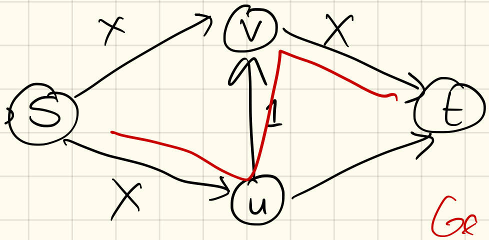
$$O(Cm)$$

C: Σ all capacities

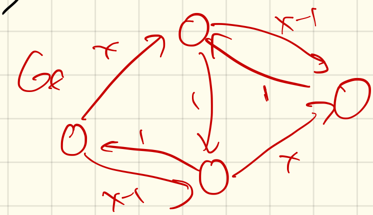
$$O(C'm)$$

C': $\sum_{\text{edges out of } s} c(e)$

Note: This is the best we can do!



Worst path:



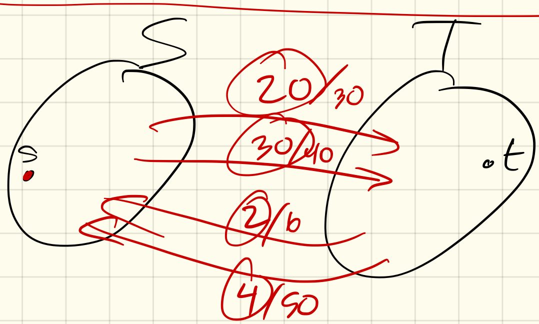
To do better, need to consider how to choose a "good" augmenting path.

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

$$\Rightarrow \text{value}(f) = f^{out}(s) - f^{in}(s)$$

pf (cont)

for all $v \in S$ <u>other</u> than s

$$f^{in}(v) = f^{out}(v)$$

$$\Rightarrow f^{out}(v) - f^{in}(v) = 0$$

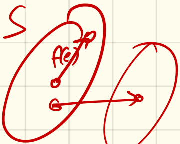
$$\Rightarrow v(f) = \sum_{v \in S} \left(f^{out}(v) - f^{in}(v)\right)$$

Rewrite: Consider edges

if $e = uv$:

<u>$u, v \in S$</u>:

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



<u>$u, v \notin S$</u>: not in sum

$u \in S, v \notin S$

appears as $+f(e)$

$u \notin S, v \in S$

appears as $-f(e)$

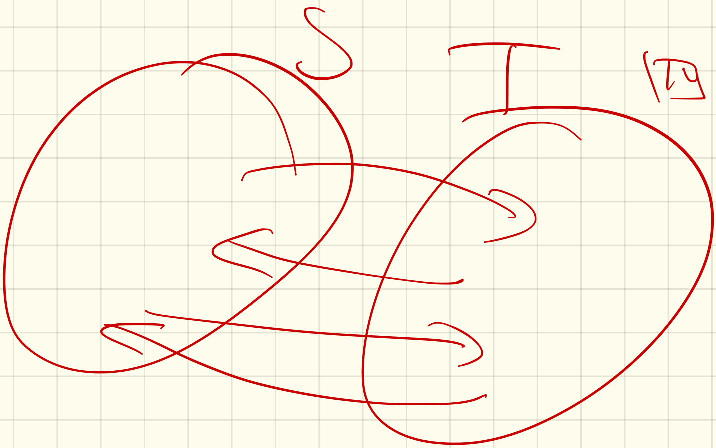
Goal: $v(f) = f^{out}(S) - f^{in}(S)$

have:

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

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

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



<u>Thm:</u> Let f be any s-t flow
+ (S,T) an s-t cut.
$$\text{value}(f) \le \text{cost}(S,T)$$

pf

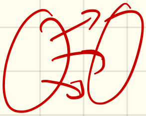
$$\text{value}(f) = f^{out}(S) - f^{in}(S)$$

(last thm)

$$\le f^{out}(S)$$

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

$$\le \sum_{e \text{ out of } S} c(e)$$

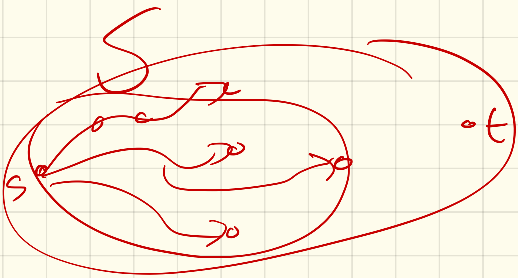


$$= \text{cost}(S,T)$$

Thm: If f is s-t flow with no s-t path in G_f, then $\exists$ s-t cut (S^*, T^*) in G with $\text{cost}(S^*, T^*) = \text{value}(f)$.

$$\boxed{\text{\underline{cor}: max flow = min cut}}$$

Pf: use G_f:



pf (cont):

Faster versions

- Depend upon choosing <u>good</u> augmenting paths!

<u>Ex</u>: Edmonds-Karp:
choose largest bottleneck edge

$\hookrightarrow O(E^2 \log E \log |f^*|)$

<u>Ex</u>: <u>shortest</u> augmenting path

$\hookrightarrow O(VE^2)$

Even more!

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

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

Next week: