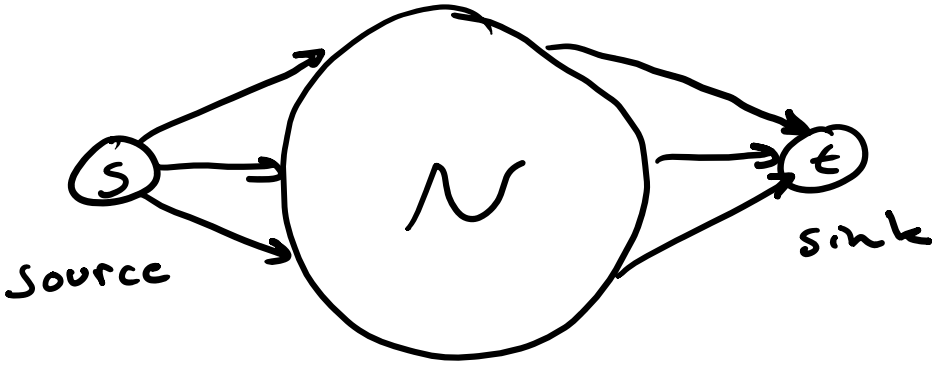
The end of an era:

- Slota's box of 100 green tea bags in his office is no more
- Purchased circa 2017-18
- Expired September 2019 (still tasted fine this week)

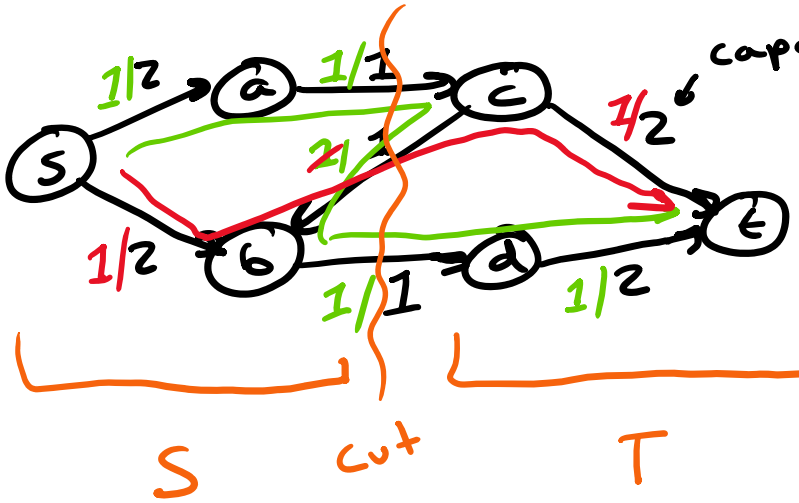
Flow Networks



how much "flow" on each edge

$\forall e \in E(G)$: we have a capacity

defined as $c(e) \geq 0$ (integer)



$[S, T]$ = our cut

$|[S, T]| = 2$

cut OR flow

a flow on G assigns to each edge e some flow value $f(e)$

→ This flow must be feasible

Feasible: $\forall e \in E(G): 0 \leq f(e) \leq c(e)$

$\forall v \in V(G)$: $f^-(v)$ = sum of flows into v

$f^+(v)$ = sum of flows out of v

conservation of flow → $f^-(v) = f^+(v)$

$f^+(s) = f^-(t)$

($f^+(s)$: flows from source; $f^-(t)$: flows into sink)

Give feasible flow f:

we define f-augmenting path P_f

- P_f goes from s to t
- $\forall e \in E(P_f)$:

  if P_f follows a forward edge, we need $f(e) < c(e)$

  if P_f follows a backward edge

if P_f follow a backward edge,
we need $f(e) > 0$

$\epsilon(e) = c(e) - f(e)$ as our tolerance
for forward edges

$\epsilon(e) = f(e)$ as our tolerance
for backward edges

Given P_f, we consider the
minimum tolerance over
all $e \in E(P_f)$

$\longrightarrow$ define as $z = \min_{\forall e \in E(P_f)} \epsilon(e)$

To augment our flow:

$\forall e \in E(P_f)$: $f(e) += z$ for forward e
$f(e) -= z$ for backward e

Source-sink cut

$[S, T]$

S = source set of vertices

S = source set of vertices

T = sink set of vertices

Note: the size of $|[S,T]|$ is just the sum of capacities of the cut edges

Q: How do we know which vertices are in S, T

A: $S = \{$ vertices that can be reached via pseudo-S-aug. paths$\}$

↗ maximal path from s but can't reach t and follows f-aug. path "rules"

$T = \overline{S} = \{V(G) - S\}$

Note: the size of any cut gives us a bound an maximum flow

$$|[S,T]| \geq val(f)$$

↑ total flow on the network

Big Question

total flow on the network

Does minimum cut = maximum flow

A: yes

↳ Let's prove this via a few relations / equivalences

1. f is a maximum flow
2. no f-aug. paths on our network
3. $|[S,T]| = val(f)$

We'll show $1 \Rightarrow 2 \Rightarrow 3 \Rightarrow 1$

$(1 \Rightarrow 2)$ contrapositive

(1 => 2) contrapositive

¬2 => ¬1

there exists an f-aug. path

=> f is not maximum

→ we've already shown how to increase flow using an f-aug path

(2 => 3)

no f-aug. paths => $|[S,T]| = val(f)$

Note: $s \in S$, $t \notin S$

all edges from $S \to T$ have

$c(e) = f(e)$

all edges from $T \to S$ have

$f(e) = 0$

$val(f) = \sum$ flows from $S \to T$

$- \sum$ flows from $T \to S$

$= 0$

$val(f) = \sum$ flows from $S \to T$

$val(f) = \sum$ flows from $S \to T$

$= \sum_{e \in [S,T]} c(e) = |[S,T]|$

$(3 \Rightarrow 1)$ (cut = flow) $\Rightarrow$ flow is maximum

Note: the capacities on edges gives us cut = flow

Q: Can we increase our flow?

A: No. Forward edges are at full capacity and backward edges are at zero flow

$\Rightarrow$ we cannot increase our flow

Combined with with our earlier inequality

$\rightarrow$ { cut $\geq$ max flow
and cut = max flow

minimum cut = maximum flow

□

To get max-flow/min-cut
(min s,t-cut)

Initialize all $f(e) = 0$

while $\exists$ some f-aug. path P_f:

Find z = min tolerance on P_f

update $f(e)$ for all $e \in E(P_f)$

→ we're done

To get our min cut:

define our cut as $[S, T]$

S = vertices reachable from s on pseudo-f-aug. paths

T = all other vertices

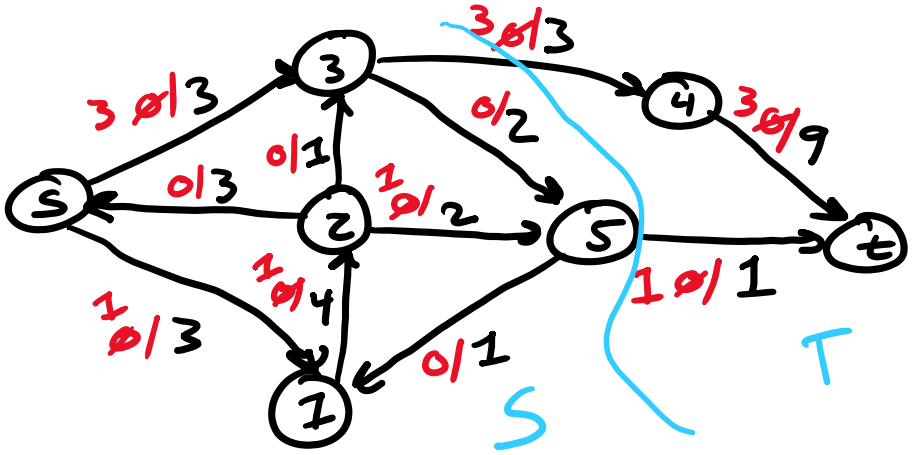
Basic Algorithm:

Ford-Fulkerson Algo

If we use BFS to find P_f:

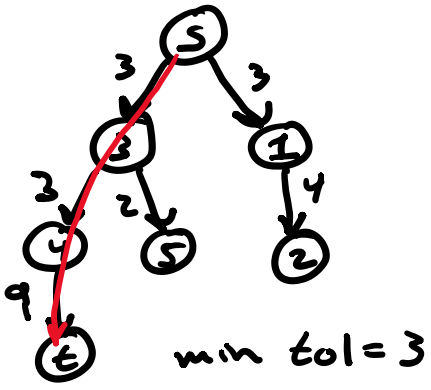
Edmonds-Karp Algo

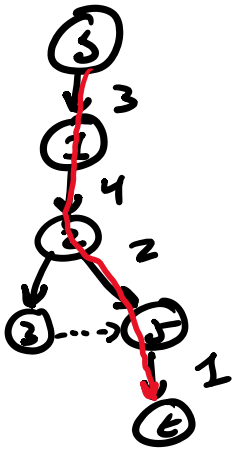
Example

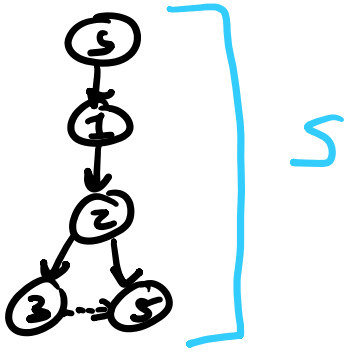


$S = \{s, 1, 2, 3, 5\}$

$T = \{4, t\}$







$val(S) = 4$

$|[S, T]| = 4$

max flow = 4 = min cut

WP7

Assume our G is undirected

→ to make G directed

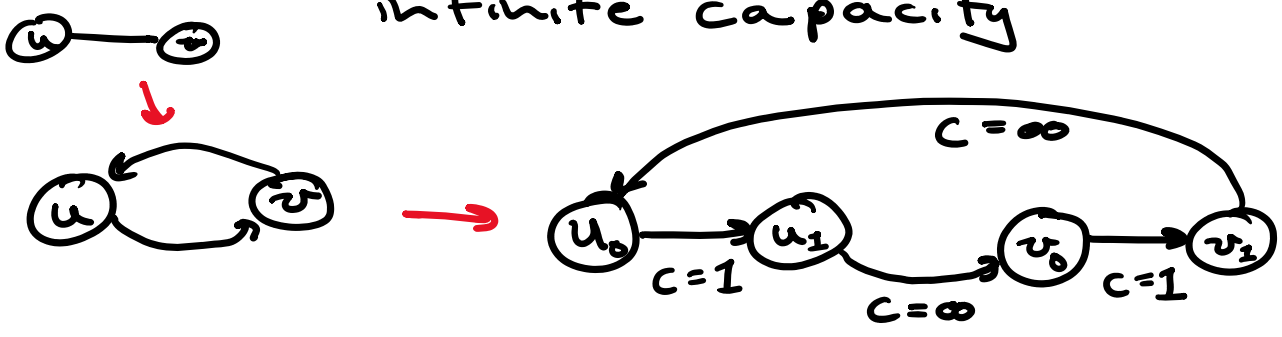
replace each $e = (u, v) \in E(G)$

↳ with $e' = (u \to v)$

$e'' = (v \to u)$

For the transformation:

* replace each vertex v with a single directed edge with unit capacity
* this edge can be specified via new vertices $v_0 \to v_1$, where v_0 has all of v's in-edges, and v_1 has all of v's out edges
* all other edges have infinite capacity



Note: we can define any arbitrary vertices in same G as our source/sink pair

Note: To prove $K(x,y) = \lambda(x,y)$, think of the equivalence

(f-aug paths $\leftrightarrow$ idps)

between our original G and transformed G'