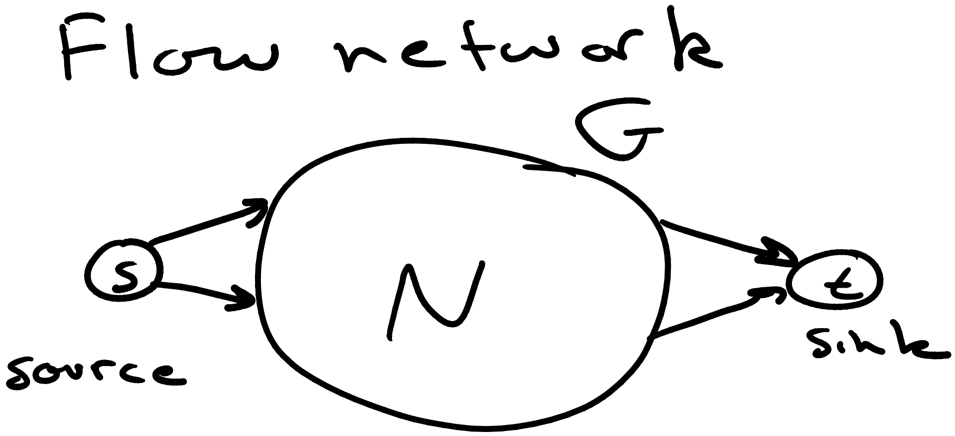
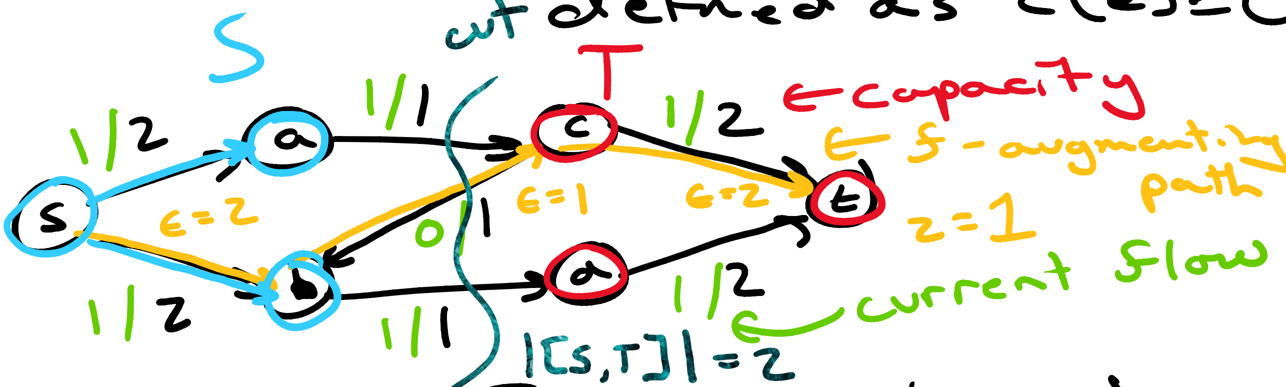
Lecture 15 - Network Flow

Thursday, March 17, 2022 10:13 AM

Flow network



$\forall e \in E(G)$: we have a capacity defined as $c(e) \geq 0$



a <u>flow</u> on G assigns to each edge a flow value $f(e)$

these flow values must be <u>feasible</u>

$\forall e \in E(G)$:

$$0 \leq f(e) \leq c(e)$$

$\forall v \in V(G)$:

$f^-(v)$ = flow into v

aka sum of flows on incoming edges

aka sum of flows on incoming edges

$f^+(v)$ = flow out of v

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

flow of our whole G
val(f) = total flow
defining our current flow

$$\mathrm{val}(f) = f^+(s) - f^-(s) \leftarrow \text{usually } 0$$
$$= f^-(t) - f^+(t)$$

__maximum flow__ = feasible flow where val(f) is maximum

Given a feasible flow f, an f-augmenting path is a source $\rightarrow$ sink path where
$\forall e \in P_f$:
- if P_f follows direction of e then $f(e) < c(e)$

then $f(e) < c(e)$

- if P_f goes against direction of e

then $f(e) > 0$

$\epsilon(e) = c(e) - f(e)$ = tolerance of e
for forward edge

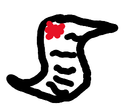
$\epsilon(e) = f(e)$ = tolerance of e
for backward edge

Given P_f, we consider the minimum tolerance $\forall e \in P_f \rightarrow z$

To augment our flow:

$\forall e \in P_f$: $f(e) += z$ for forward edges

$f(e) -= z$ for backward edges

Note : when we augment a flow, we increase val(f) by z

source-sink cut $[S, T]$

S = source set of vertices

T = sink set of vertices

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

the cap of this cut

the size of this cut
aka the capacity of the cut

$$= \sum c(e) \quad \forall e \in [S, T]$$

$S = \{$vertices than can be reached from s by following pseudo-f-augmenting path$\}$

$T = \{$everything else$\}$

Note: the size of a cut gives us a bound on flow

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

Big Question

Does min cut = max flow

{Answer = yes

to prove this, let's consider some equivalences

↳ to prove this, let's show the same equivalences

1. f is a max flow
2. no f-augmenting paths
3. $|[S,T]| = \text{val}(f)$

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

$(1 \Rightarrow 2)$ contrapositive

$\neg 2 \Rightarrow \neg 1$

$\exists$ f-augmenting path $\Rightarrow$ f is not a max flow

→ We've already seen how to increase flow given an f-augmenting path

$(2 \Rightarrow 3)$

no f-augmenting paths $\Rightarrow$ cut equal to flow on the network

S = set of reachable vertices from s following potential f-aug paths

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

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

$$f(e) = 0$$

$$\text{val}(f) = \sum \text{flows from } S \to T \underbrace{- \sum \text{flows from } T \to S}_{=0}$$

$$= \sum \text{flows from } S \to T$$

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

(3 => 1) cut = flow => flow is max

Note: the capacities on edges gives us our cut = flow

Q: can we increase our flow

A: No. we've observed that forward edges are at capacity forward edges are at capacity and backward edges are at zero flow → no f-aug paths

⇒ our flow is maximum ✓

Combined with our prior inequality from above → cut ≥ max flow and cut = max flow

$\boxed{\text{minimum cut} = \text{maximum flow}}$

QED

To get max-flow / min cut:

initialize all flows to zero

while ∃ some f-aug path:

  find z = min tol on path

  update flows by min tol

⇒ we're done

min cut is defined as $[S, T]$ where S is "reachable" vertices

= Ford-Fulkerson algorithm

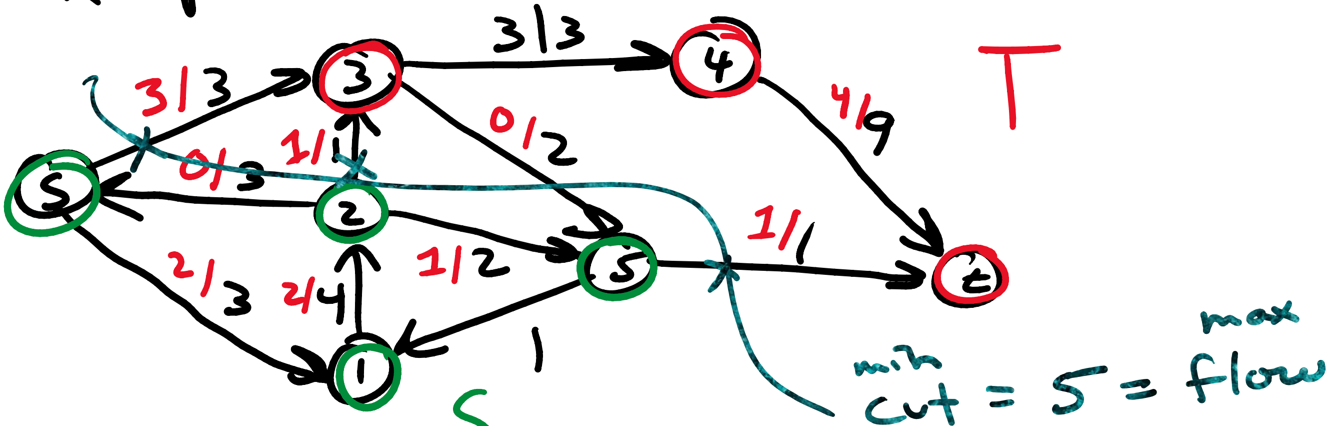
where s is reach[illegible]

→ Ford-Fulkerson algorithm

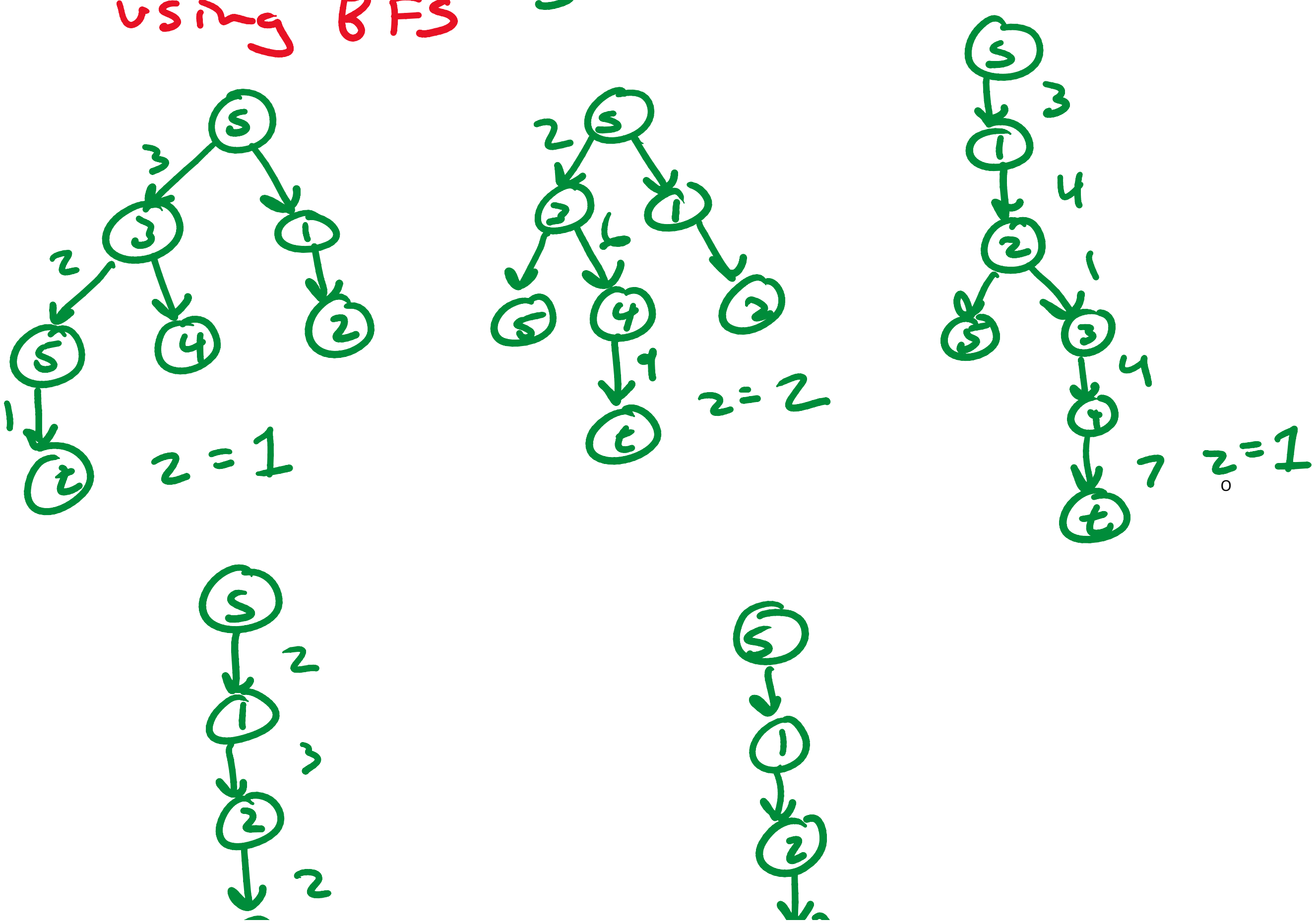
if we use BFS to find f-aug paths

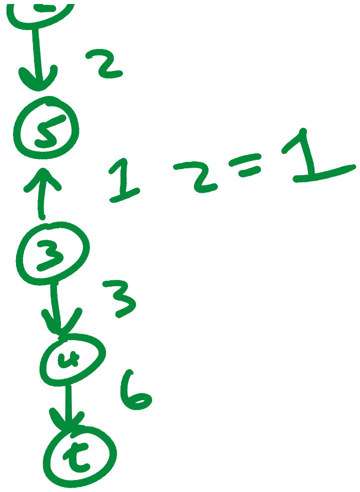
→ Edmonds-Karp algorithm

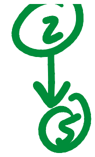
Example



using BFS




2
s
1 2=1
3
3
4
6
t


2
s