Algorithms (pt 2)

Flows: Ford-Fulkerson thm

Recap

Grading + midterm scores:

HWs 0 to 4 are graded

All Perusall is updated

Midterm "guess" is on banner

↳ drop deadline is Sunday (?)

due next Friday

Next: HW6, plus reading as usual

↳ MST, SP-T, MSSP-T

Ch 10: Flows

Motivation:

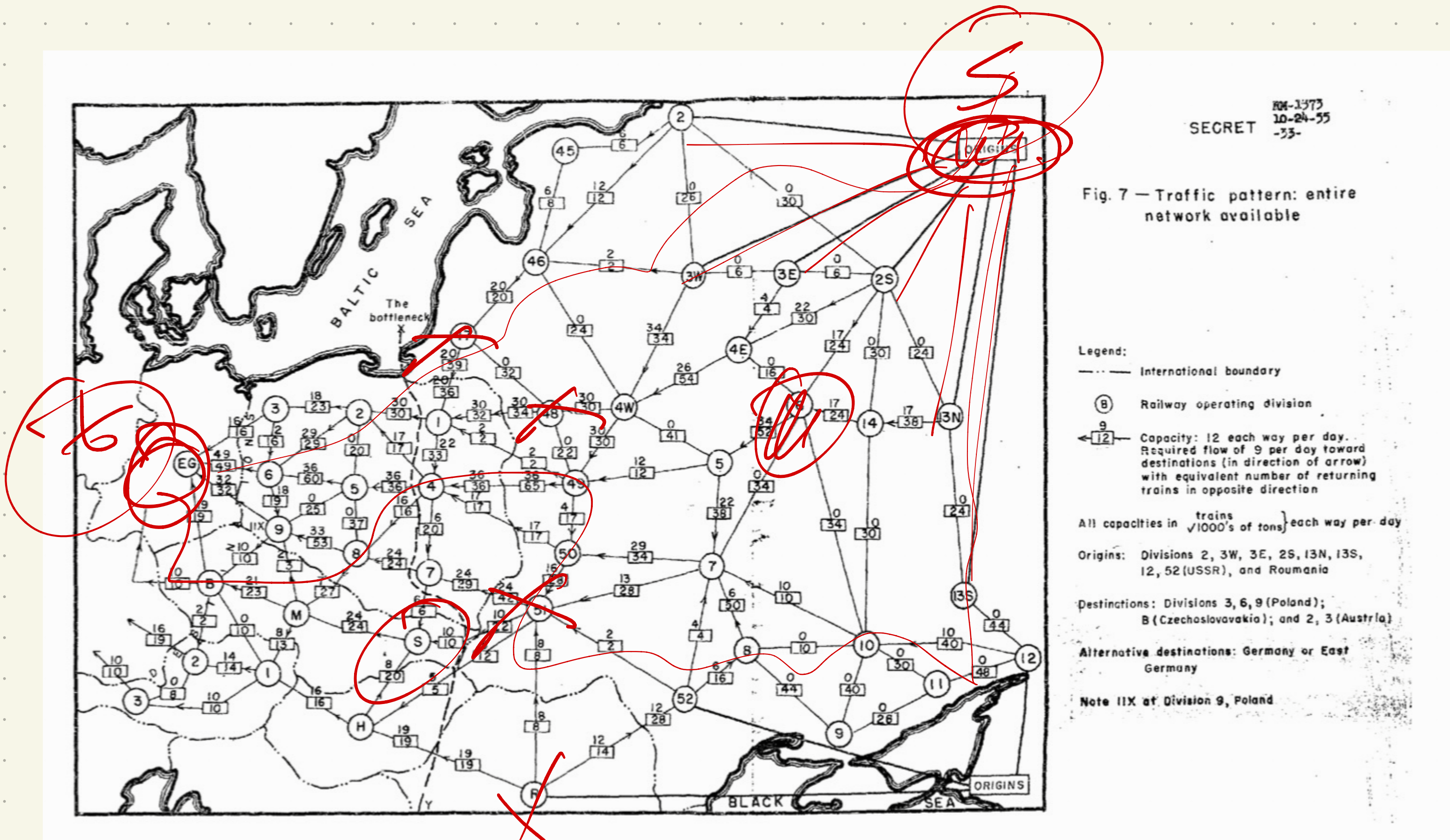
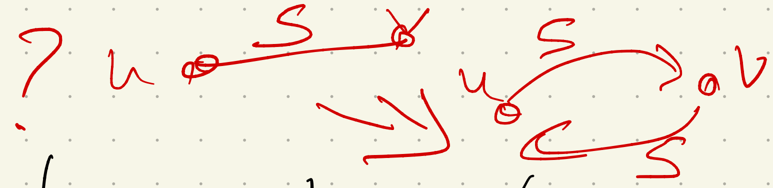


Figure 10.1. Harris and Ross's map of the Warsaw Pact rail network. (See Image Credits at the end of the book.)

How to send from one vertex to another?

How to <u>divide</u> one vertex from another?



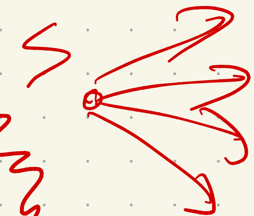
More formally:

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

Each edge is given a capacity c(e). → maximum amount it can carry

Assume:

- No edges enter s.
- No edges leave t.
- Every $c(e) \in \mathbb{Z}$. never are irrational → can't store those in computer



Goal:

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

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

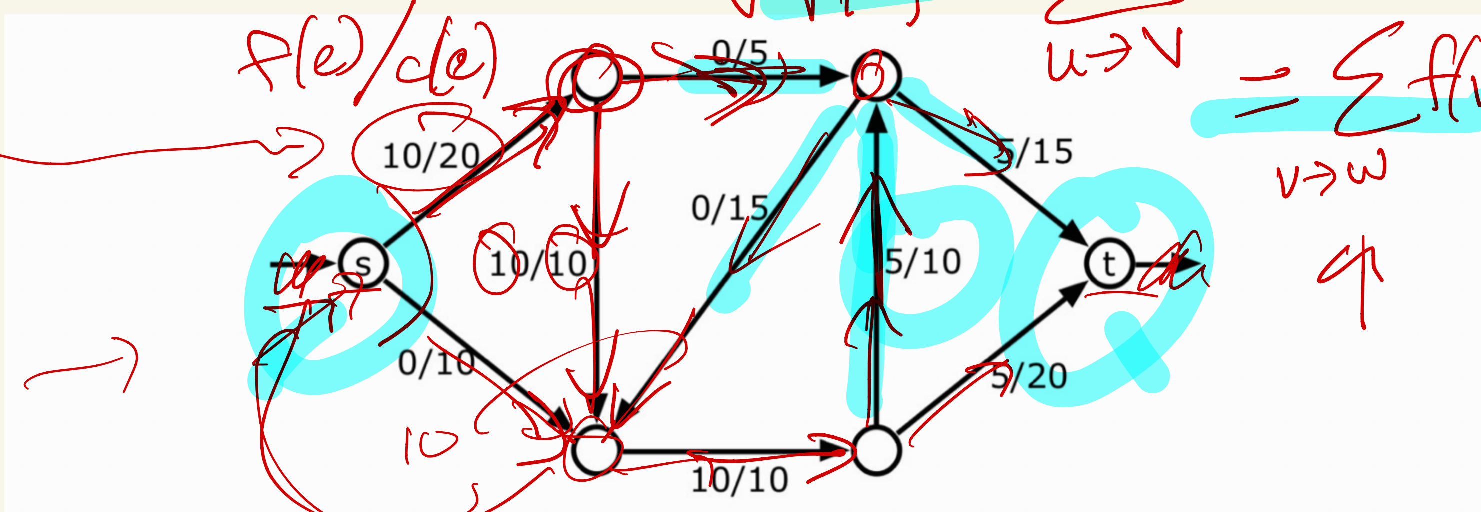
Flows:

A **flow** is a function $f: E \to \mathbb{R}^+$, where $f(e)$ is the amount of flow going over edge e.

Must satisfy 2 things:

- Edge constraints: $0 \le f(e) \le c(e)$ don't overload edge
- Vertex constraints: by design don't want product shipped unless it can get to t

$$\forall v \ne s,t: \sum_{u \to v} f(u \to v) = \sum_{v \to w} f(v \to w)$$



An (s,t)-flow with value 10. Each edge is labeled with its flow/capacity.

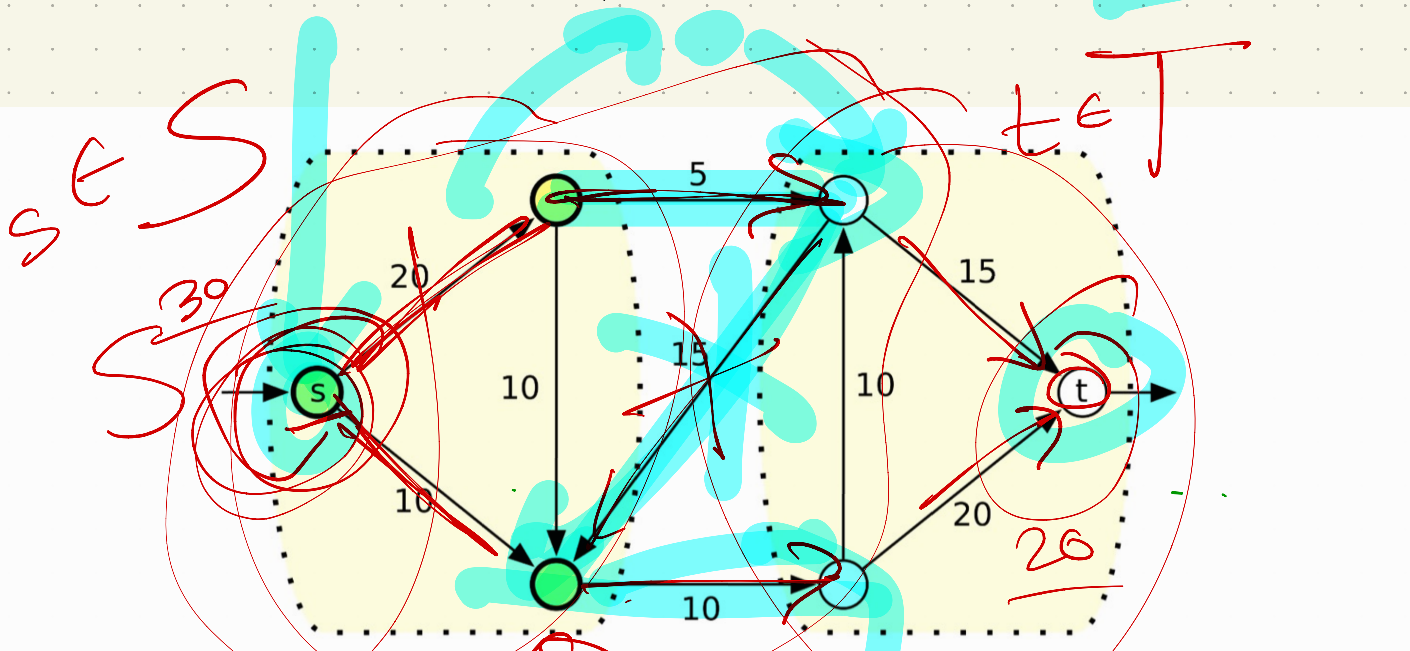
$$\text{Value}(f) = \sum_{e \text{ out of } s} f(e) = \sum_{e \text{ into } t} f(e)$$

Cuts:

An <u>s-t cut</u> is a partition of the vertices into 2 sets, S and T, so that:

- $s \in S$
- $t \in T$
- $S \cap T = \emptyset$, $S \cup T = V$

The <u>capacity</u> of a cut is $\displaystyle\sum_{\substack{\overrightarrow{uv} \in E \\ \text{with } u \in S,\, v \in T}} c(\overrightarrow{uv})$



An (s, t)-cut with capacity 15. Each edge is labeled with its capacity.

Thm: (Ford-Fulkerson '54, Elias-Feinstein-Shannon '56)

The max flow value = min cut value

Wow! these seem so different...

One way is easy:

Any flow (max) ≤ any cut (min cut).

Why?

any flow has to get out of S in order to reach $t \in T$

Next: Show that can get them equal.

How?

Well, take some flow, f.

Either:

1. It f is maximum, in which case find a cut of equal value.

2. It isn't, then find a bigger flow.

Key: Build a residual graph

$c(e)$ capacity, $f(e)$ flow

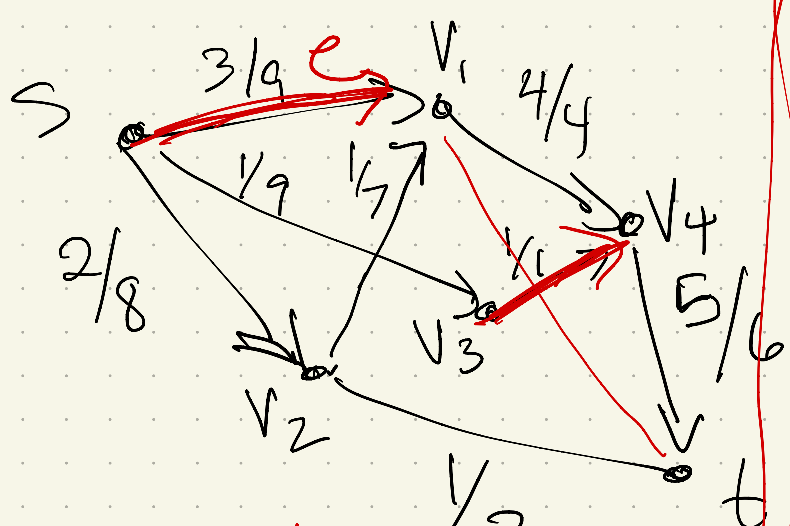
Residual capacity: Given G & F:

$$C_f(\underbrace{u \to v}_{\text{edge}}) := \begin{cases} c(u\to v) - f(u \to v) & \text{if } u \to v \in E \\ f(u \to v) & \text{if } v \to u \in E \\ 0 & \text{otherwise} \end{cases}$$

build 2 res. cap.

$C_f(v_1 \to t) = 0$

Ex: flow on G:



$e = s \to v_1$

$c(e) = 9$

$f(e) = 3$

$C_f(s \to v_1) = 9 - 3 = 6$

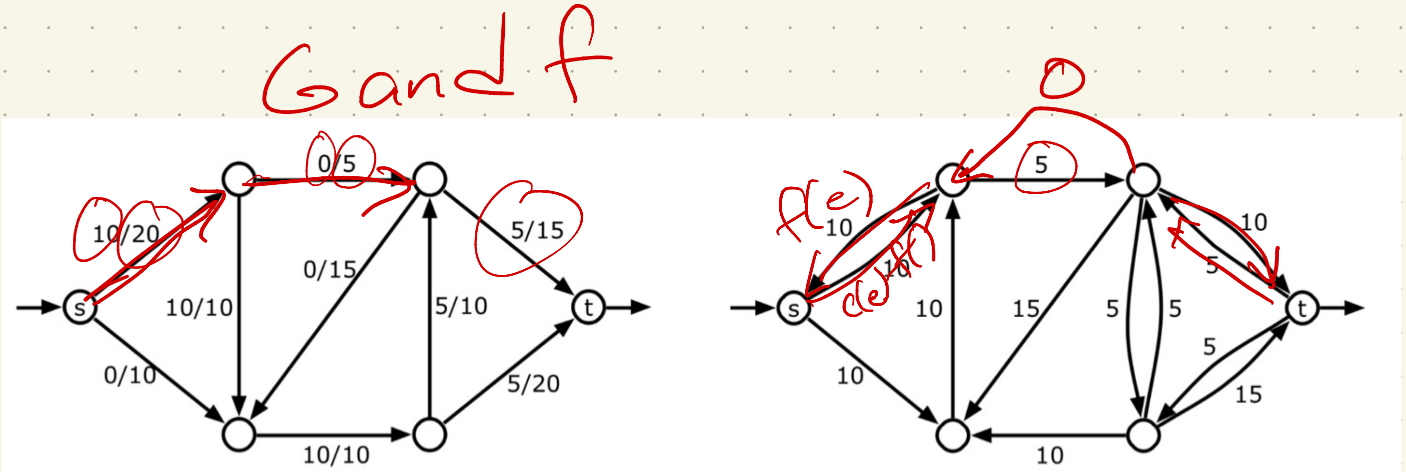
$C_f(v_1 \to s) = f(s \to v_1) = 3$

$C_f(v_3 \to v_4) = 1 - 1 = 0$

$C_f(v_4 \to v_3) = 1$

We usually visualize this as a new graph, G_f:

G and f



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

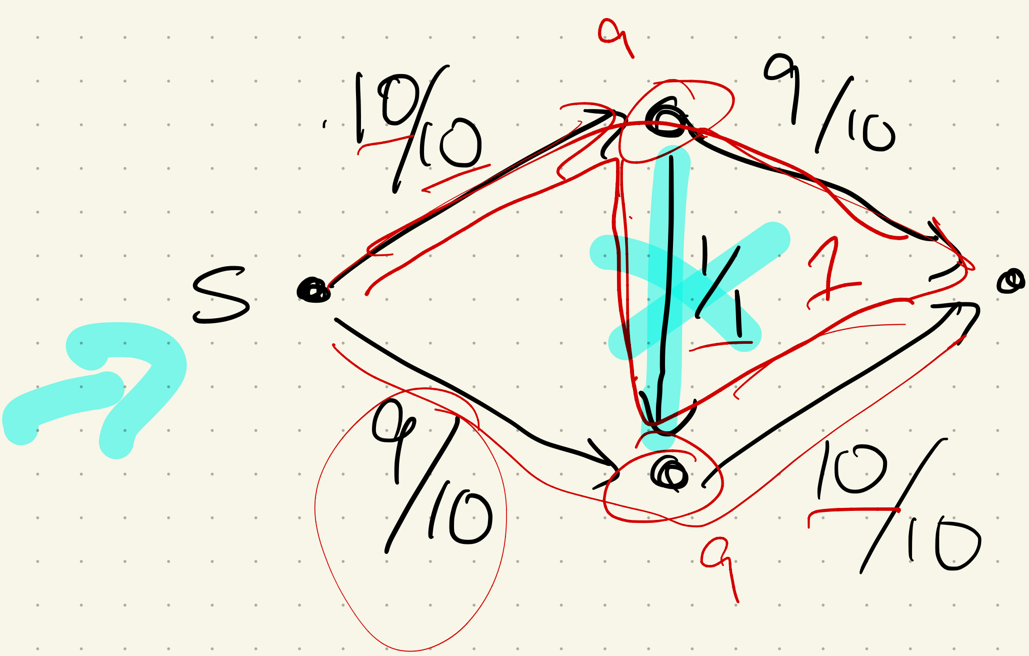
Intution:

A path in G_f if a way to send more flow!

(or get a cut of same value as f, if no path from $s \rightsquigarrow t$ in G_f)

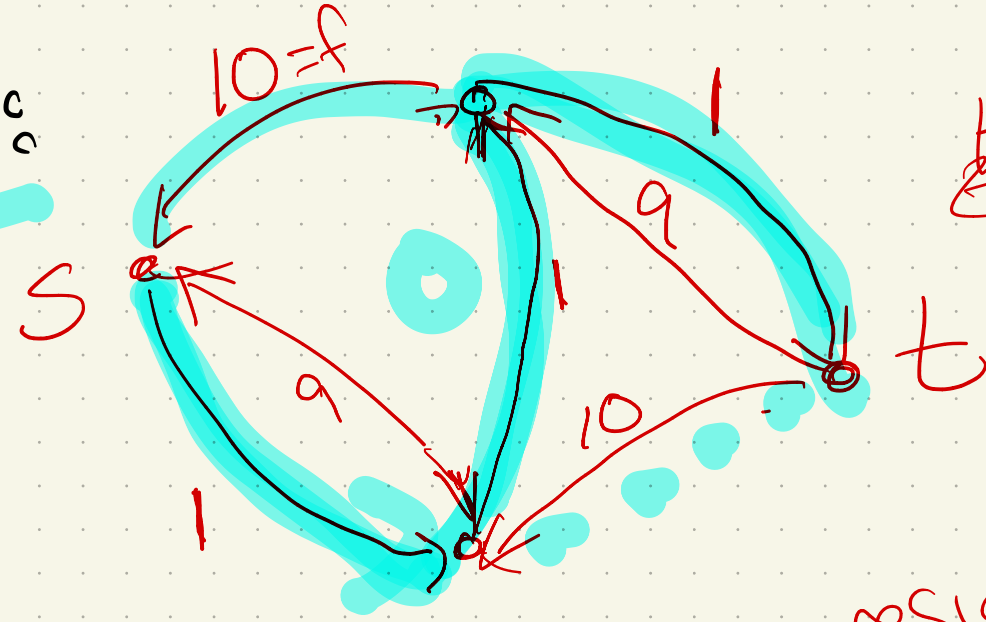
Another example:

greedy "stuck" flow from last time



no s-t path of flow left

G_f:



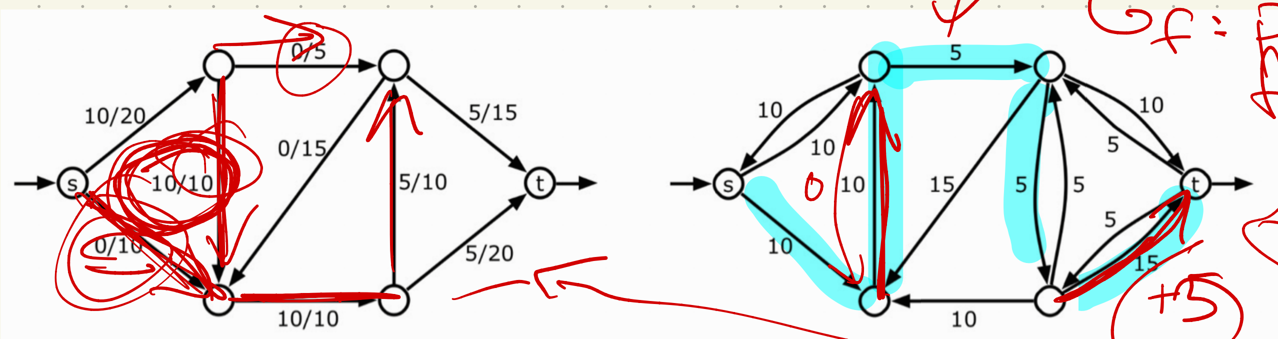
path!! 9a

(only pos resid, cap edges shown)

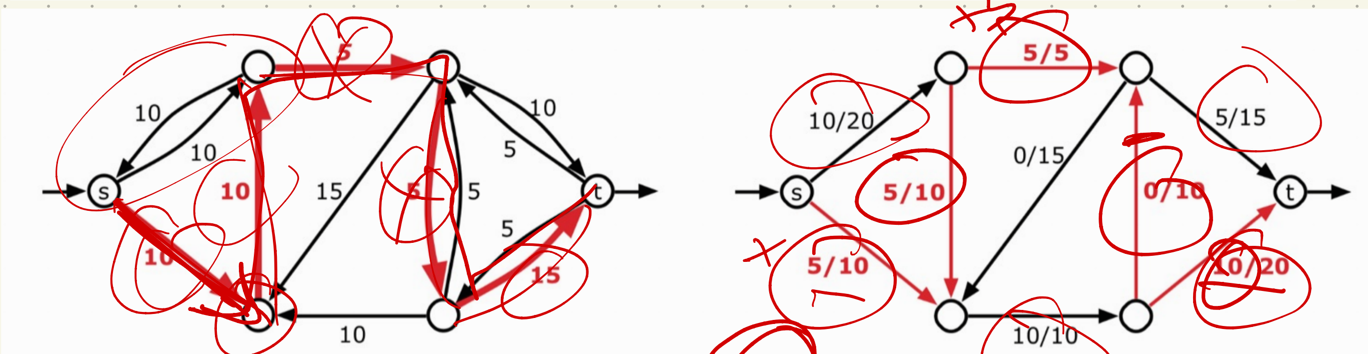
Augmenting a path:

Suppose there is a path in G_f from s to t:

G_f: path from s to t



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



An augmenting path in G_f with value $F = 5$ and the augmented flow f'.

$$f' = \begin{cases} \text{for } e \text{ on path:} \\ \text{if } u \to v \text{ is in } G, \\ \quad \text{reset } f(u \to v) \mathrel{+}= \text{value of path (here} = 5) \\ \text{if } v \to u \text{ is in } G \\ \quad \text{reset } f(u \to v) = f(u \to v) - \text{value of path} \end{cases}$$

new flow: changes flow along that s-t path in G_f

<u>Claim</u>: f' is also a feasible flow!

(& larger)

Why?

- For any $u \to v$ not on augmenting path, value is same, which means $\leq c(e)$

- For $u \to v$ on augmenting path,

$$f'(u \to v) = f(u \to v) + F$$

value of path

$$\geq f(u \to v) \geq 0$$

Still feasible!

$\leq c$

b/c F was chosen to be $\leq$ largest capacity

or $f(v \to u)$, in which case unpushing

$\hookrightarrow$ math to verify

Claim: If f is a maximum flow, the G_f has no augmenting path.

↳ $s \rightsquigarrow t$ path in G_f

Proof: contradiction

Assume f is maximum.

Build G_f & find path.

Use this path to build a bigger flow f'.

contradiction — f wasn't maximum.

So → if maximum flow, G_f has no $s \rightsquigarrow t$ path left.

__So:__ f wasn't a max flow, since f' is larger.

On other hand:

if G_f has no $s \to t$ path, find $|S|$ = set of vertices that s can reach.

cut!

__Claim:__ $(S, V-S)$ is a cut.

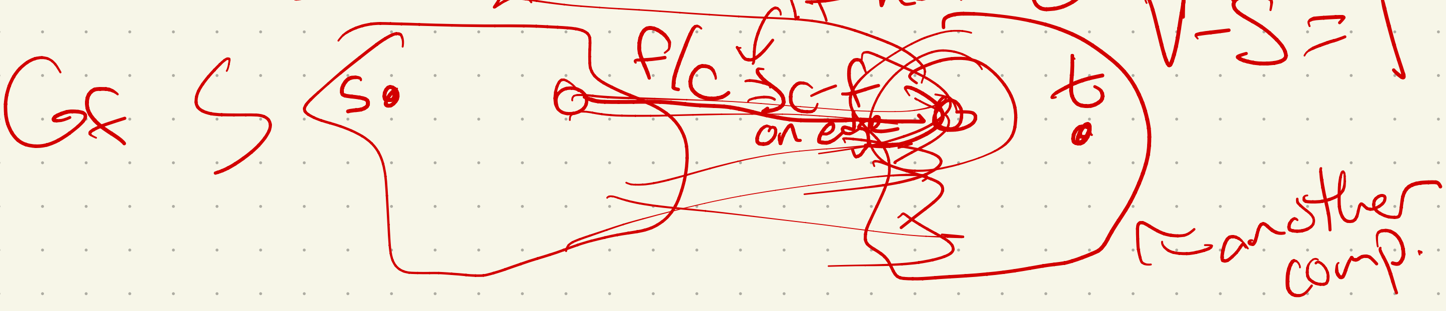
(& f uses every $S \to V-S$ edge to its max capacity)

Why?

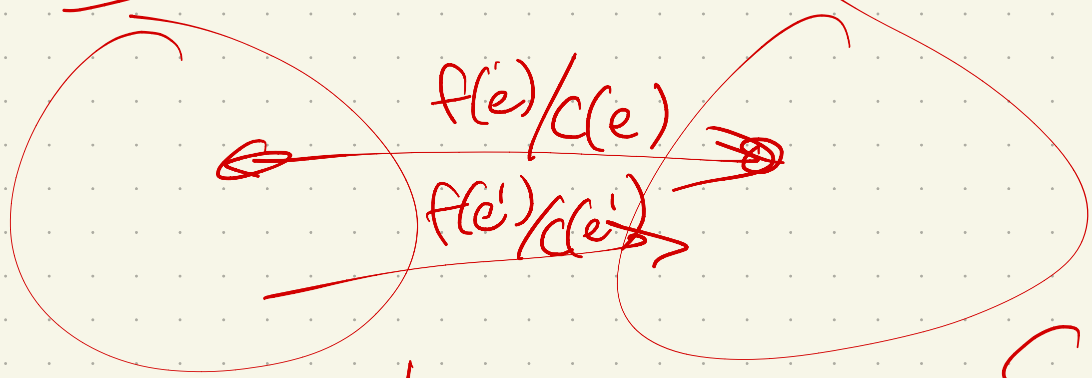
run WFS on G_f starting at $S \Rightarrow$ component of G_f containing S

if not at capacity

$V-S = T$



G S V − S



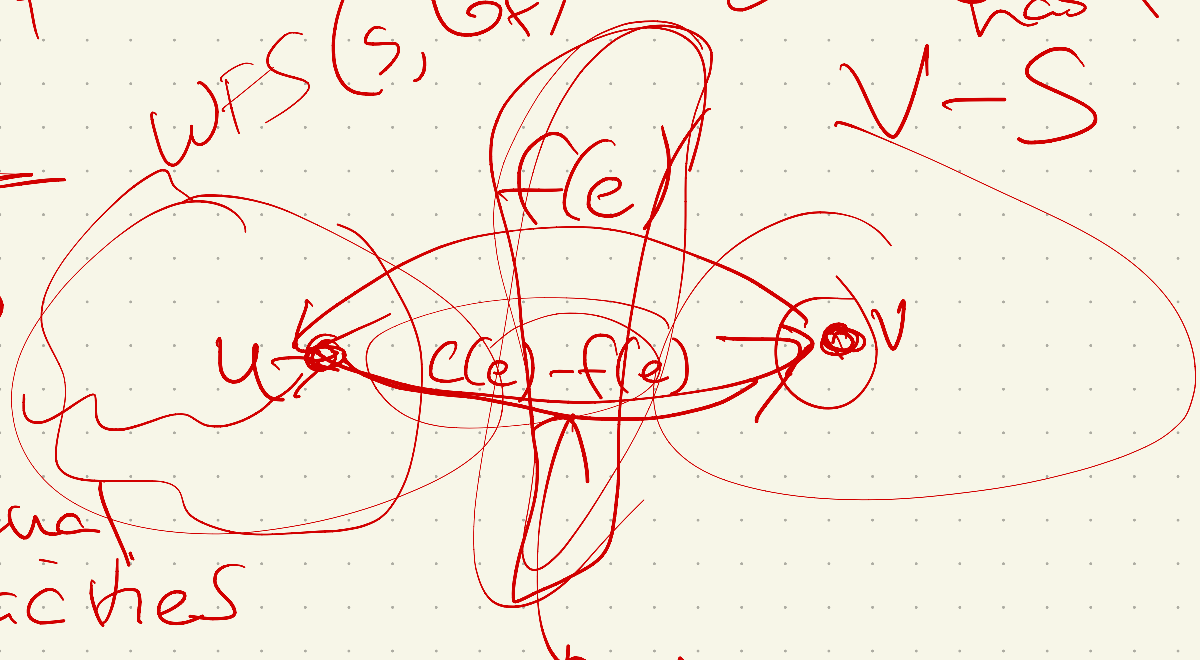
if no $s \rightsquigarrow t$ path

$S \rightsquigarrow T$

every edge has $f(e) = c(e)$

G_f WFS(s, G_f)

S V − S



residual capacities

if this is $\neq 0$

s could reach v

means: $f(e) = c(e)$

since couldn't WFS to v

Immediate Algorithm: (F-F alg)

Start with $f = 0$.

Build G_f

WFS(G_f, s)

While t & s in same component:

  find $s \to t$ path via WFS

  Augement along the path to get f'

  $f \leftarrow f'$

  Build G_f

  WFS(G_f, s)

Runtime:

Why all this integrality stuff?
We are assuming each path pushes at least 1 more unit of flow!

Can it be that bad?

Yes:

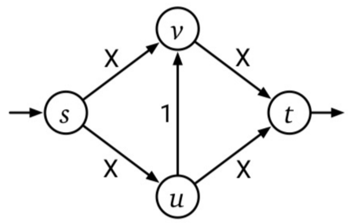
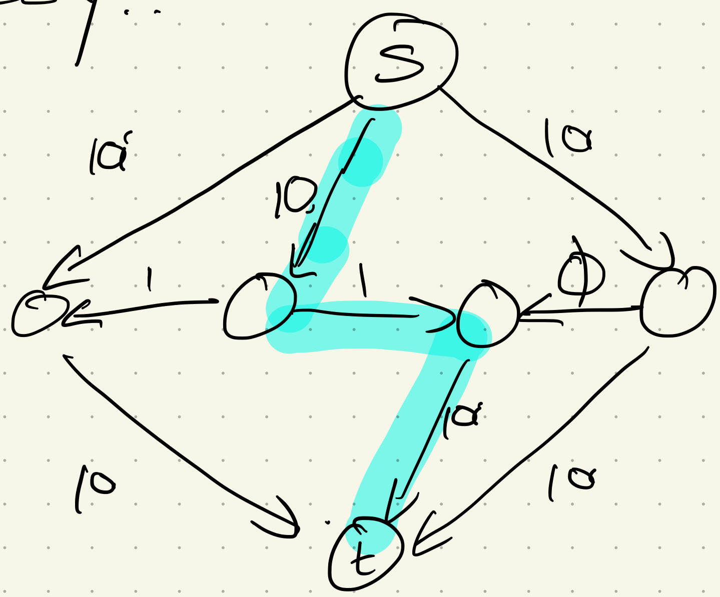


Figure 10.7. Edmonds and Karp's bad example for the Ford-Fulkerson algorithm.

How "big" is f?
(Remember, not part of input!)

What if it's not integers?

Messy!!



The key:

$\phi = \frac{1+\sqrt{5}}{2}$

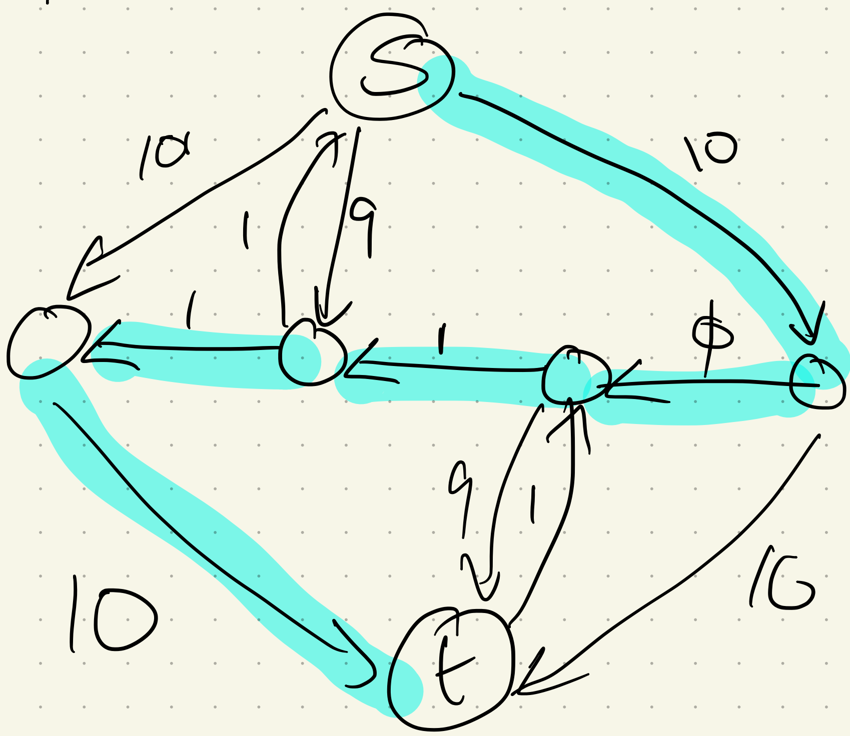
WHY??

Simple:

$1 - \phi = \phi^2$

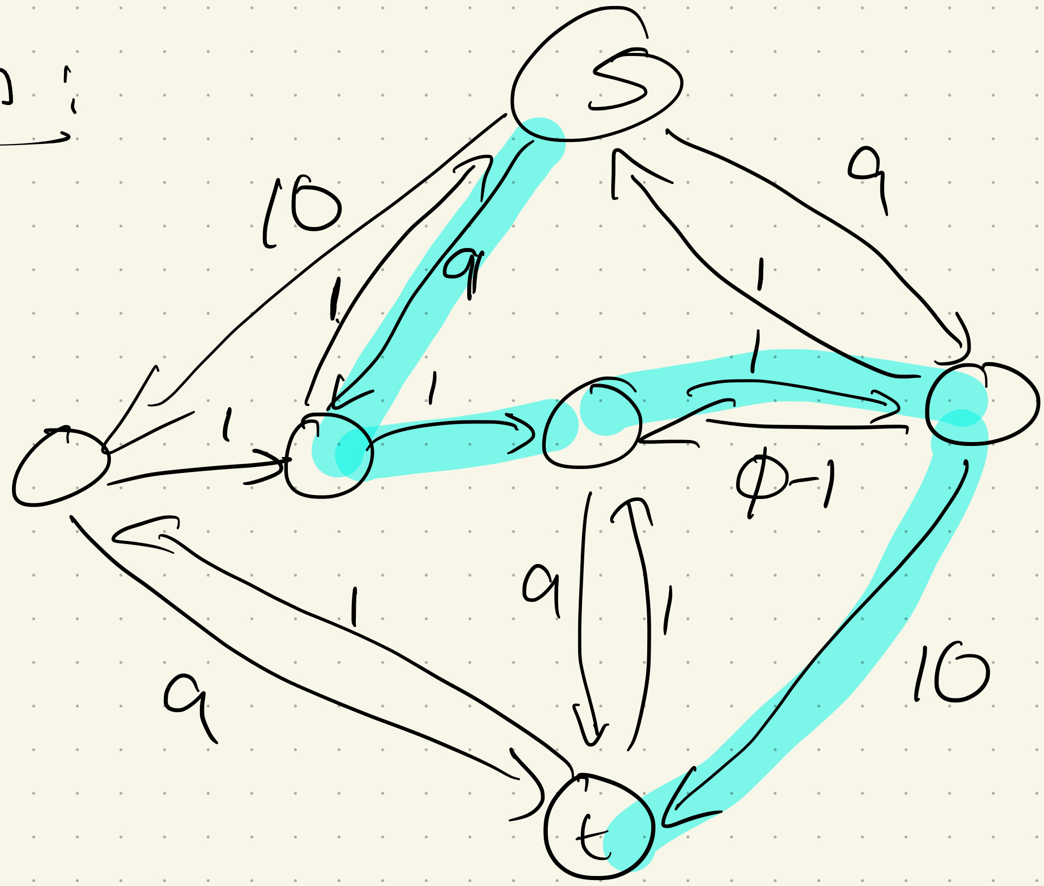
G_f:

Next path:



New G_f:

Then:



Continue to push:

Ends with:

ϕ, 0, and $1-\phi=\phi^2$

Repeat:

- ϕ^2, 0, ϕ^3

then

-

etc ⋮

However: max flow is $=21$