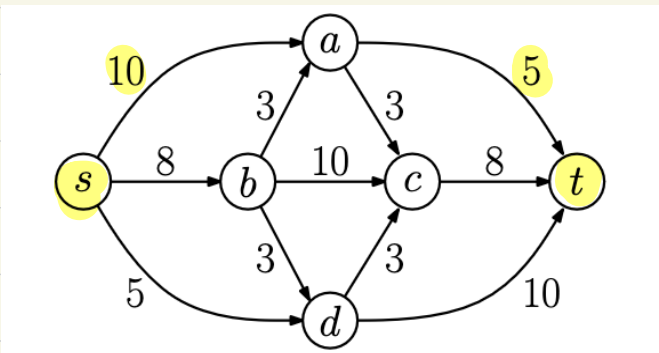
CMSC 451 Oct. 21

Network Flow

Flow network : a digraph $G = (V, E)$

- for each $(u, v) \in E$,
  capacity $c(u, v) \geq 0$
- two special vertices
  source (s) & sink / terminal (t) s.t.
  no edges enter s or leave t



Maximum Flow Problem:

think of edge (u, v) as a pipe
that carry $c(u,v)$ units of water per second.
how much water we can push from s to t?

Formally defining flow.

A flow is a function $f : E \to R^{\geq 0}$ s.t.

real numbers ≥ 0

$f(u,v)$: amount of water that flow through (u,v)

1. (Capacity constraint):

$\forall (u,v) \in E$, $f(u,v) \leq c(u,v)$

2. (Flow conservation / Balance)

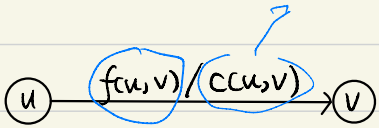
$\forall v \in V \setminus \{s,t\}$,

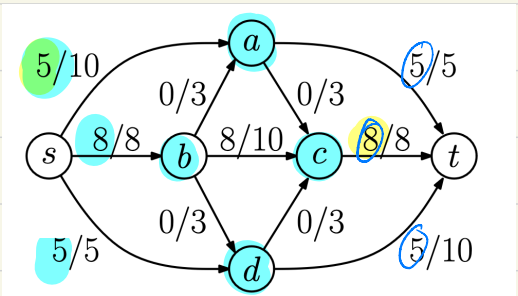
$$\sum_{(u,v) \in E} f(u,v) = \sum_{(v,w) \in E} f(v,w)$$

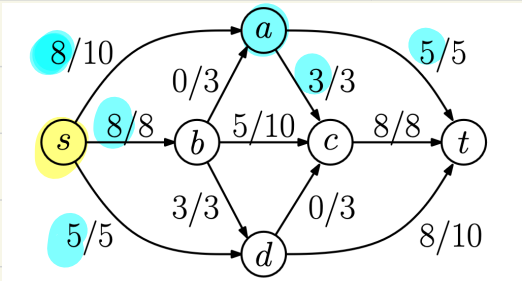
total flow into v. $f^{(in)}(v)$

total flow out of v. $f^{(out)}(v)$

Examples:







$|f| = 5+8+5 = 18$

$|f| = 8+8+5 = 21$

maximum flow

The value of a flow f:

$$|f| = f^{out}(s) = \sum_{w \in V} f(s,w)$$

$$= f^{in}(t)$$

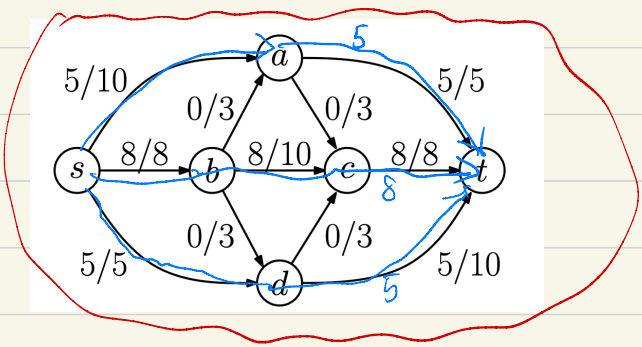
same value

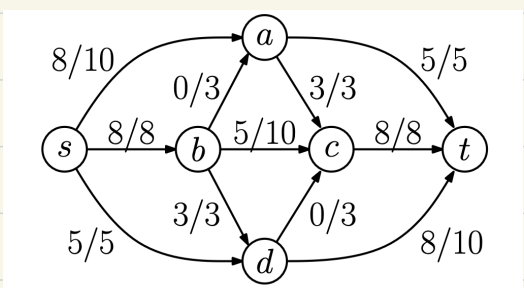
Maximum Flow: find the flow with maximum value

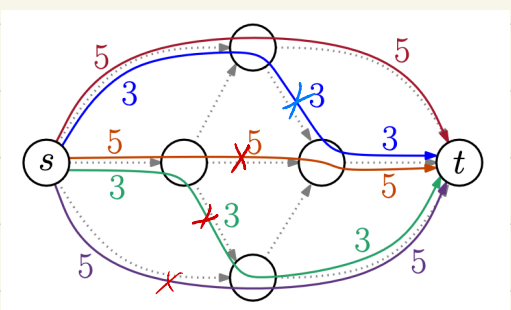
Path-based view:

An equivalent way to view a flow is as a collection of paths from s to t.

- each path is assigned a weight.
- sum of the weights does not exceed edge capacity





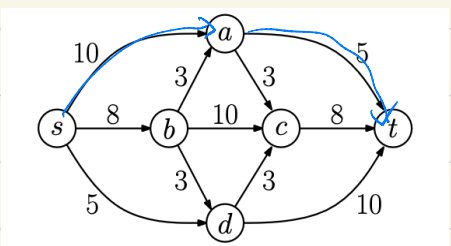


- value of flow = $\sum$ weight of all paths

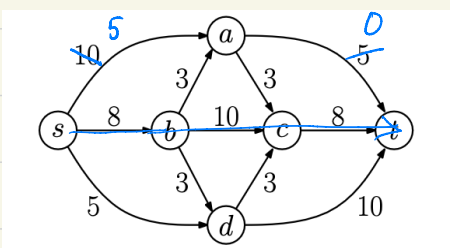
Claim: A flow network has an edge-based flow of value x iff it has a path-based flow of value x.

Greedy Algorithm:

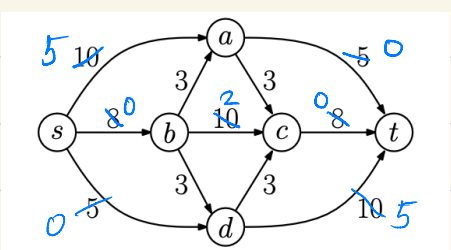
- find any path from s to t that has positive capacities.
- push as much flow as you can along the path.
- reduce the capacities on the path.



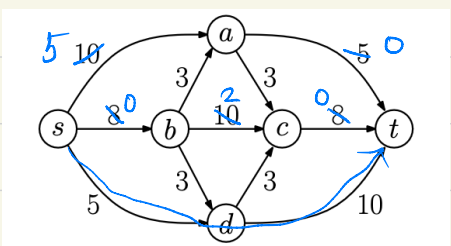
push 5 →



↓ push 8



← push 5

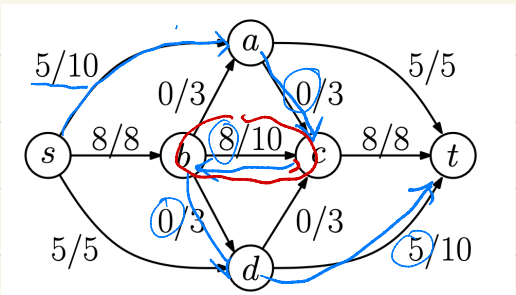


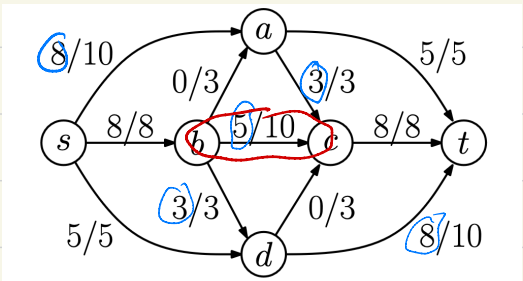
↓ no positive path $s \to t$ anymore!

found a flow w/ value $5+8+5=18$

Greedy is not optimal!

Why?





original greedy:

only consider push though edges w/ unoccupied capacity.

⇒ what we can also do,

push some flow in the reverse direction by reducing the amount of the flow in the forward direction.

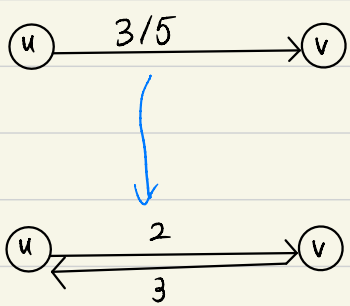
Residual Network:

Given a flow network G and flow f,
define the residual network G_f to be:

- same vertices V, s & t.
- for each edge $(u,v) \in E$, w/ $c(u,v)$ & $f(u,v)$

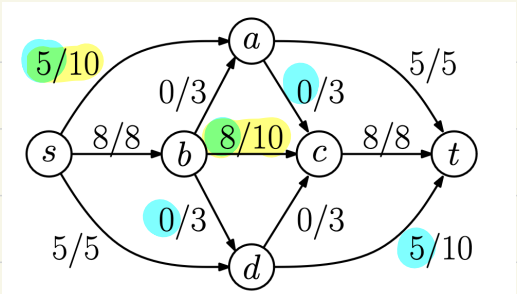
  - add forward edge (u,v) w/ capacity
    $c_f(u,v) = c(u,v) - f(u,v)$

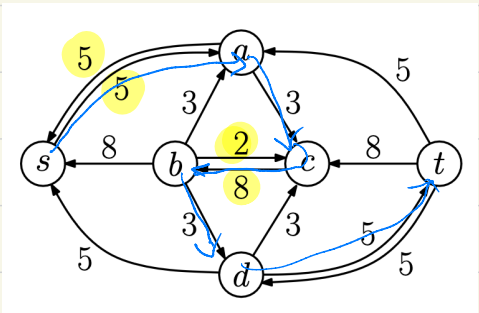
  - add backward edge (v,u) w/ capacity
    $c_f(v,u) = f(u,v)$

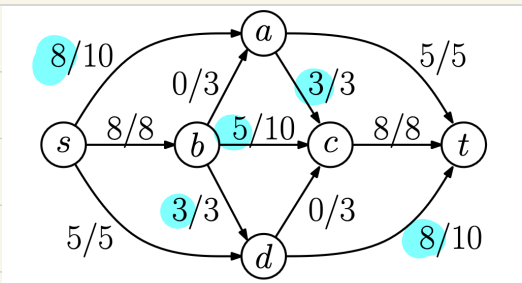


1. you can → push at most 2 more $u \to v$

↳ push at most 3 more $v \to u$


a
5/10
0/3
0/3
5/5
s
8/8
b
8/10
c
8/8
t
0/3
0/3
5/5
d
5/10


a
5
5
5
3
3
s
8
b
2
c
8
t
8
3
3
5
5
d
5


a
8/10
0/3
3/3
5/5
s
8/8
b
5/10
c
8/8
t
3/3
0/3
5/5
d
8/10

Property of Residual Networks

G_f has a (positive) flow $\Rightarrow$ G has a flow better than f.

> Claim. Given network G & flow f, let f' to be a flow on G_f, then $f+f'$ is a flow of G

Proof sketch:

$\forall (u,v) \in E$, want to prove that $f+f'$ satisfies capacity on it.

if (u,v) is a forward edge in G_f:

f valid on G $\Rightarrow$ $0 \le f(u,v) \le c(u,v)$

f' valid on G_f $\Rightarrow$ $0 \le f'(u,v) \le c_f(u,v)$

$= c(u,v) - f(u,v)$

$\Rightarrow 0 \le f(u,v) + f'(u,v) \le c(u,v)$

if (u,v) is a backward edge

$f+f'$ $\Rightarrow$ reducing $f(v,u)$ by $f'(u,v)$

$f'(u,v) \le c_f(u,v) = f(v,u)$

G_f has a flow $\Leftarrow$ G has a flow better than f.

↳ proof is harder! will prove in the next lecture

Ford-Fulkerson Algorithm (1956)

Idea: - Start with $f = 0$.

- repeatedly find a path on the residual graph G_f.
- $f' \leftarrow$ max flow on the path.
- update residual graph based on new flow $f + f'$.

```
ford-fulkerson(G, s, t)
    for (u,v) ∈ E
        f[u,v] ← 0
    while (true)
        Gf ← residual graph.
        if no s->t path : return f.
        p ← any path on Gf s->t.
        c ← min capacity on p in Gf
        f(u,v) ← f(u,v) + c, ∀ (u,v) ∈ p
```

O(n+m) to update the G_f given the updates of f

finding a path O(n+m) time by DFS

O(n+m) to update f-value

Running time : $O(n+m)$ per iteration.

How many iteration?

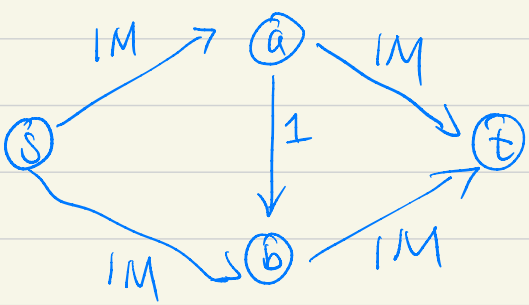
If all capacities are integers

then each round, the value increases at least by 1

$\Rightarrow$ # rounds $\leq$ max possible flow value

$\leq$ sum of capacities.

Total time $O((n+m) \cdot C)$



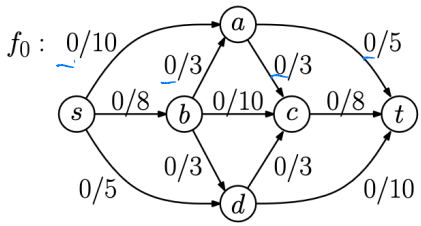
round 1 : $s \to a \to b \to t$ value 1

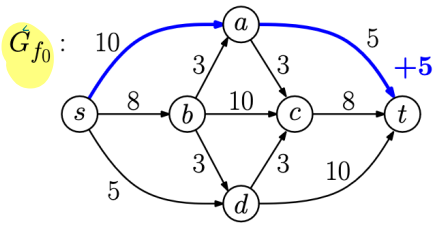
round 2 : $s \to b \to a \to t$ value 1

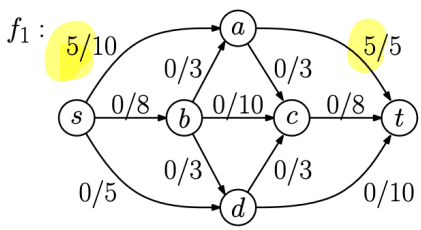
3 -- $a \to b$ -- 1

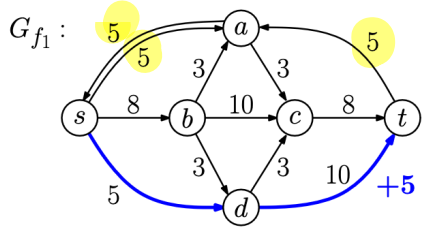
⋮

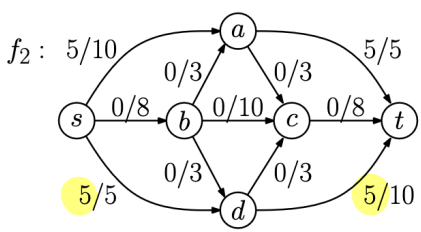
2M

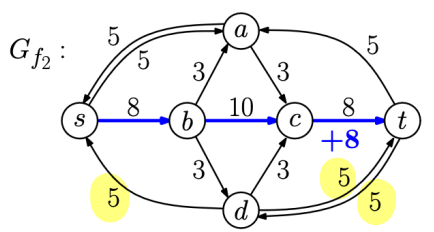


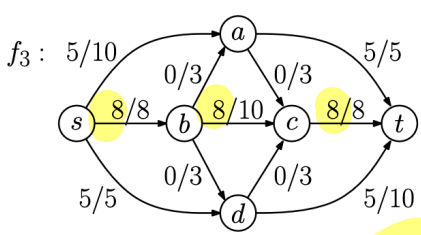


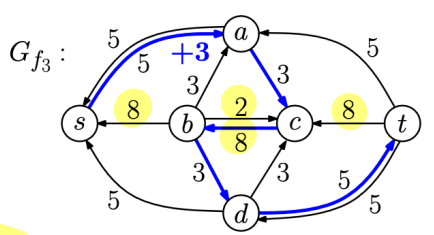




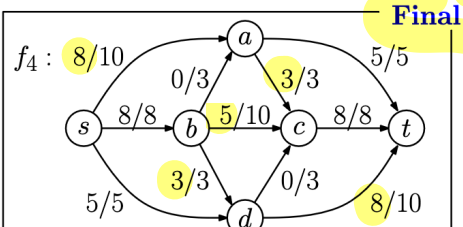








Final flow



No s-t path

