Flow network — dir graph G
vertices s, t
capacities $c(e)$

Maximum Flow

Compute $f : E \to \mathbb{R}$
- $f(u \to v) \geq 0$
- $f(u \to v) \leq c(u \to v)$
- $\sum_u f(u \to v) = \sum_v f(v \to w)$

maximize net flow out of s

Minimum Cut

Compute partition $V = S \sqcup T$ $S \cap T = \emptyset$

minimize $\sum_{u \in S} \sum_{v \in T} c(u \to v)$

Maxflow Mincut Theorem — Max flow = Min cut

```
Ford-Fulkerson(G, s, t, c)
    f ← 0
    G_f ← G
    while there is a path from s to t in G_f      ← WFS O(E)
        augment f along path                      → add integer to flow value
        rebuild G_f                               ← O(E)
    return f
```

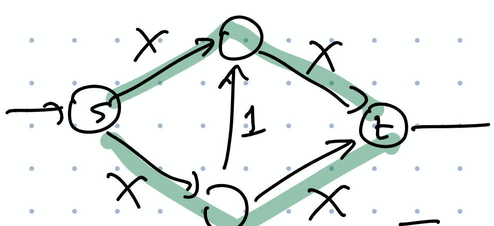
If this algo halts, it returns a max flow

If all caps are integers → integer max flow

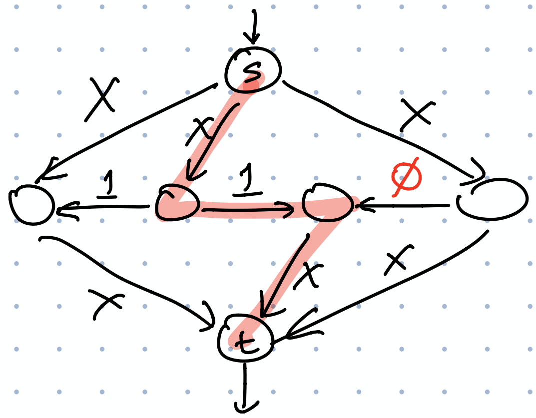
→ $O(E \cdot \underbrace{|f^*|})$

E = #edges value of max flow $\geq$ #iterations

exponential in # bits in input



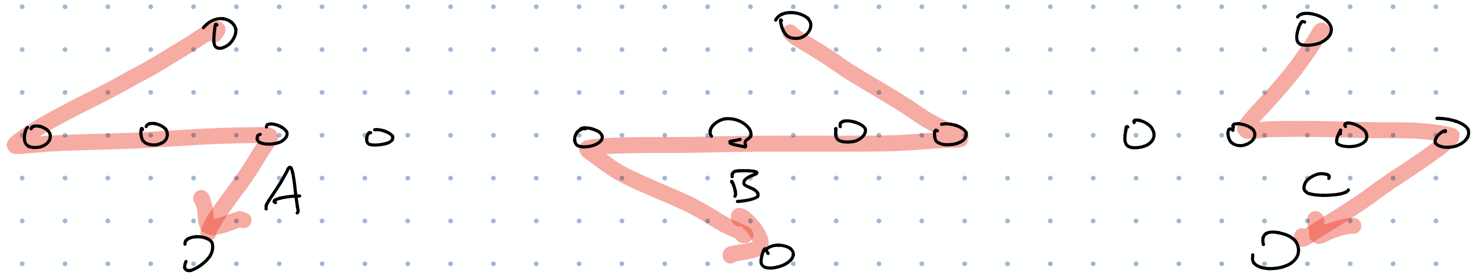
Input: $O(1 + \log X)$ Time $= O(X)$



$\phi = \frac{\sqrt{5}-1}{2} \approx 0.618$

Max flow value $= 2X+1 = 201$

$X = 100$



ϟ BCBABCBABCBA ...

∞ loop!

$1 \quad \phi \quad \phi \quad \phi^2 \quad \phi^2 \quad \phi^3 \quad \phi^3$ ....

Total flow in limit $= 1 + \sum_{i \geq 0} 2\phi^i$ < 7

Why should we care?

— Integers are artificial

— Floating point errors

Flow = assignment of numbers to edges satisfying constraints

= sum of paths and cycles

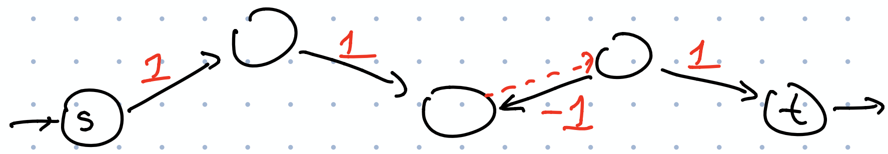
Flows can be combined

f, f' flows $\longrightarrow$ $f+f'$ is also flow

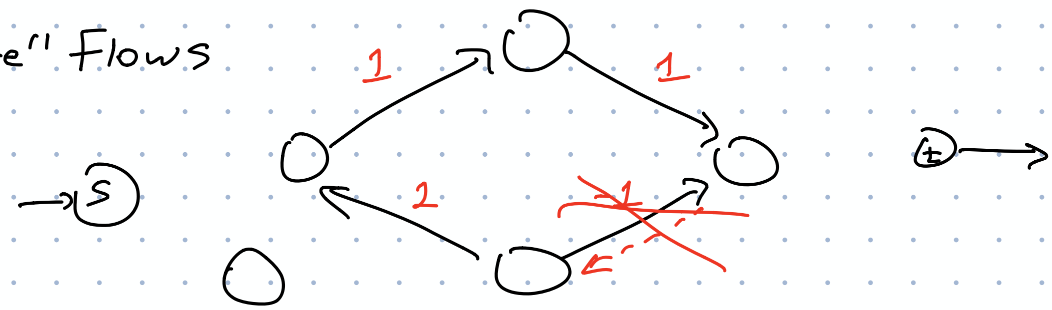
$f-f'$

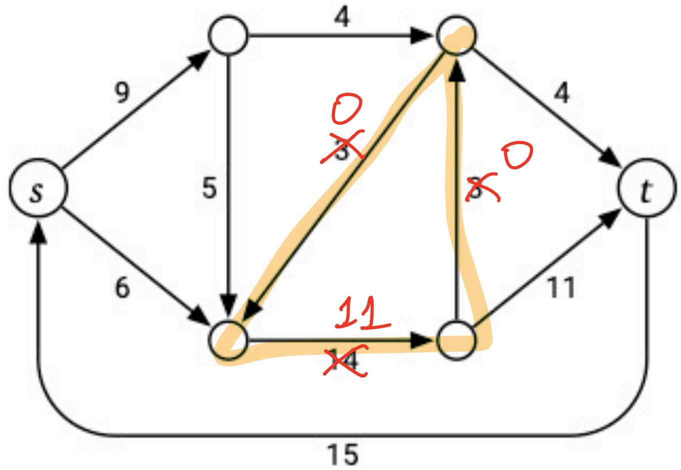
$5f+\pi \cdot f'$

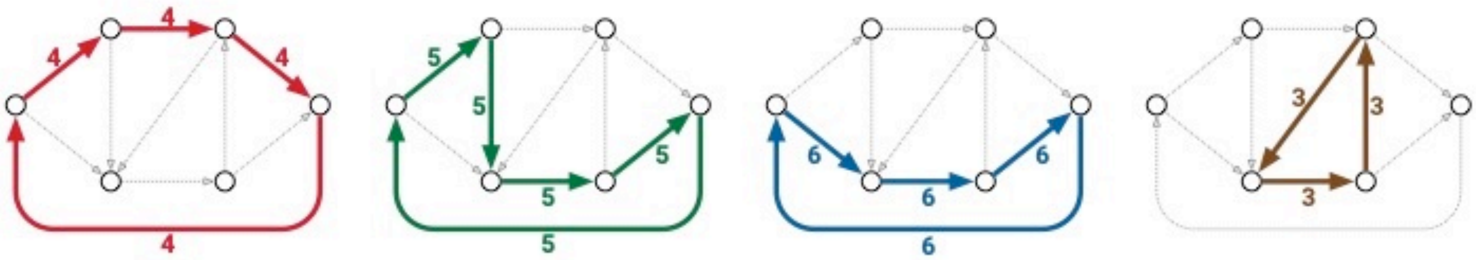
Residual Paths "are" flows



Residual Cycles "are" flows







Flow Decomp: Every non-neg flow is sum of non-neg path and non-neg cycle flows

Every circulation is sum of cycles

Proof: By induction on # edges with $f>0$.

Let f be any circulation.

- $f=0 \Rightarrow f$ is empty sum ✓
- [f is a cycle $\Rightarrow$ trivial]
- $f \neq 0 \Rightarrow$

Start at any vertex, follow edges with $f>0$ until you repeat some vertex

Cycle in f — C

subtract flow from every edge in C

$\min\{f(e) \mid e \in C\} = f_{min}$

By I.H., we can decompose $\boxed{f - f_{min} \cdot C}$ into cycles

---

1. Any acyclic flow $\rightarrow$ Σ paths
2. There is an acyclic max flow
3. Given f, compute flow decomp in $O(VE)$ time

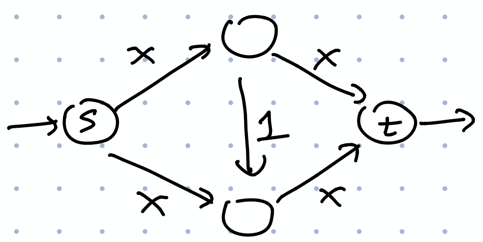
$O(V)$ - find cycle

$O(E)$ iteration

tight in worst case

Any maxflow alg that works one path at a time take $\Omega(VE)$ time.

Edmonds-Karp — Choose a good augmenting path
?



(1) Choose augmenting path with largest capacity "Fattest"

(2) choose augmenting path with fewest edges "shortest"

Fattest path — maximize min capacity

$O(E \log V)$ time per iteration "best-First search" like Dijkstra's/Kruskal's

iterations? Assume $c(e)$ are integers.

Middle — current flow f in G

$\boxed{f' = \text{maximum flow in } G_f}$

Then $f + f'$ is max flow G

Flow decomposition $\Longrightarrow$ f' is the sum of $\leq E$ paths and some cycles

$\Longrightarrow$ fattest path in the decomposition carries $\geq |f'|/E$ flow

$\Longrightarrow$ has capacity $\geq |f'|/E$

$\Longrightarrow$ Fattest path from s to t has capacity $\geq |f'|/E$

$|f|$ increases by $\geq |f'|/E$

$|f'|$ decreases by $\geq |f'|/E$ at each iteration

So after k iterations

$|f'| \leq \left(1 - \frac{1}{E}\right)^k \cdot |f^*|$ For some k, $|f'| < 1$

$\uparrow$ max flow in G

$\Downarrow$ integer

$|f'| = 0$

$k = \boxed{E \cdot \ln|f^*| \text{ iterations}}$

TWMUF: $1+x \le e^x$

$$|f'| \le \left(1-\frac{1}{E}\right)^k \cdot |f^*| = \left(1-\frac{1}{E}\right)^{E\cdot \ln|f^*|} |f^*|$$
$$\le \left(e^{-1/E}\right)^{E\cdot \ln|f^*|} \cdot |f^*|$$
$$= e^{-\ln|f^*|} \cdot |f^*|$$
$$= \frac{1}{|f^*|} \cdot |f^*| = 1$$

$\boxed{\tilde{O}(E^2 \log V \cdot \log|f^*|) \text{ time}}$ polynomial time

---

Shortest aug. path Use BFS instead of WFS

min #edges

Intuition: Over time, vertices get further from s at each iteration in G_f

Let G_i = residual graph after i augmentations

$G_0 = G$

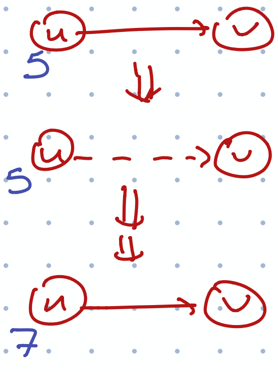
$dist_i(v) = level_i(v) =$ distance from s to v in G_i (unweighted shortest path)

Lemma: $level_i(v) \ge level_{i-1}(v)$ for all v

Proof: induction on $level_i(v)$

Lemma: Each edge $u \to v$ leaves the residual graph $\le O(V)$ times

Intuition:



When $u \to v$ reappears

dist(u) has gone up

dist(u) $\le V$ (or ∞)

$\#\text{iterations} = O(VE) \Rightarrow \boxed{O(VE^2) \text{ time}}$

Orlin's algorithm [2012] $\boxed{O(VE) \text{ time}}$

Use Orlin or Ford-Fulkerson in your HW/exams