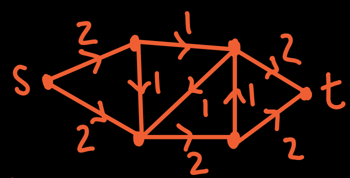
MA/AMA 514

Today §4.3, §4.4

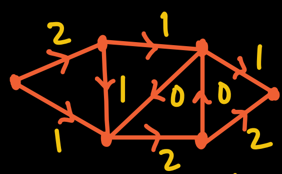
From last time: Given a directed graph $D=(V,A)$, vertices $s,t\in V$, capacity function $c: A\to\mathbb{R}_{\geq 0}$,

Max Flow : $\max\{\text{value}(f) : f \text{ is an } s\text{-}t \text{ flow under } c\}$

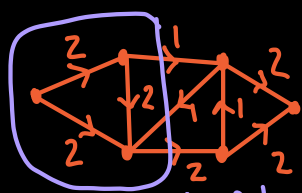
$\overset{=}{\text{Thm}}$ Min Cut : $\min\{c(\delta^{out}(U)) : U\subseteq V \text{ an } s\text{-}t \text{ cut}\}$

Ex: 

$c: A\to\mathbb{R}_{\geq 0}$



max s-t flow under c

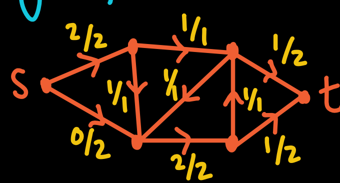


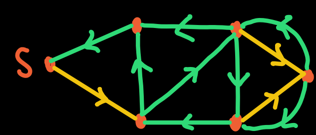
min s-t cut

Given s-t flow f, make digraph $D_f=(V, A_f\cup B_f)$ where

$A_f=\{a\in A : f(a) < c(a)\}$

$B_f=\{a^{-1} : a\in A,\ 0< f(a)\}$.

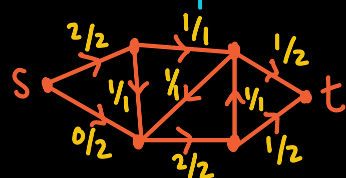


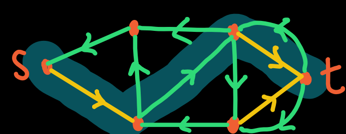


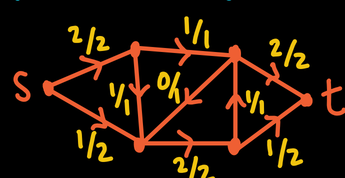
A_f B_f

An s-t path P is a <u>flow augmenting path</u>

Augment by $\alpha = \min\{c(a)-f(a) : a\in A_f\cap P\}\cup\{f(a^{-1}) : a\in B_f\}$

Ex: 





Max Flow Algorithm (Ford-Fulkerson)

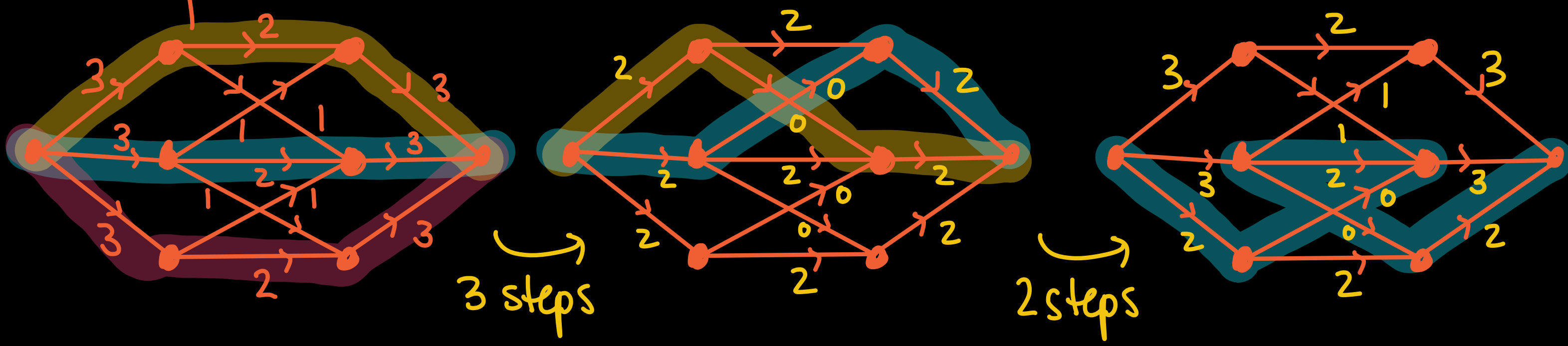
<u>Input</u>: $D=(V,A)$, $s,t\in V$, $c: A\to\mathbb{R}_{\geq 0}$

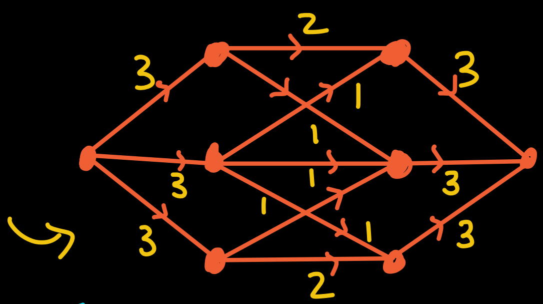
<u>Output</u>: an s-t flow f and s-t cut U with $\text{value}(f) = c(\delta^{out}(U))$

$\Rightarrow$ solves max flow, min cut!

Start with $f_0 : A \to \mathbb{R}_{\geq 0}$ with $f_0(a) = 0 \ \forall a \in A$. Repeat alg. to give increasing flows $f_1, \dots, f_N$ until it returns a cut $\mathcal{U}$.

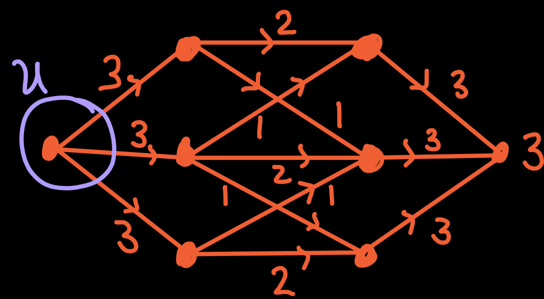
Example:





max flow value = 9

= min cut!



Thm If all capacities $c(a)$ are rational numbers, then this algorithm terminates (regardless of path choice).

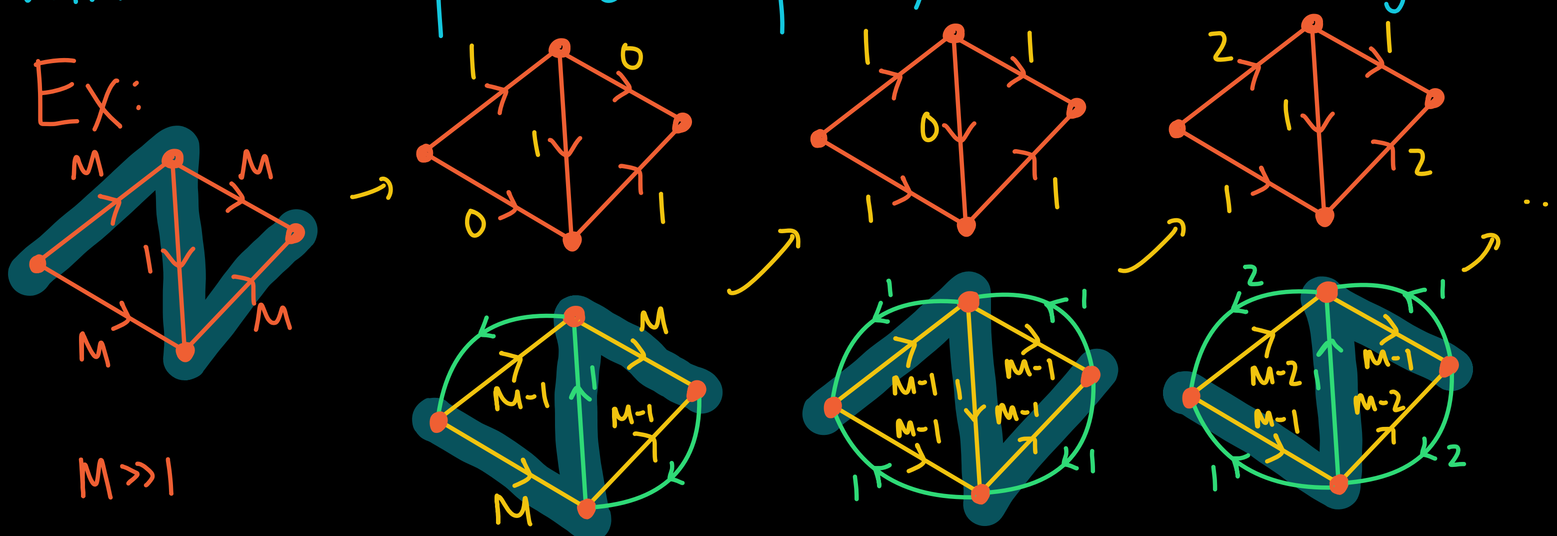
(Proof) $\exists K \in \mathbb{N}$ s.t. $K \cdot c(a) \in \mathbb{Z}_{\geq 0}$ for all $a \in A$.

Then at each step, increase α is a multiple of $1/K$.

$\Rightarrow \alpha \geq 1/K$. Then # steps bounded by $K \cdot c(\delta^{out}(s))$. $\square$

With arbitrary choice of paths, it can take this long!

Ex:



$M \gg 1$

Remarks: • Alg. doesn't always terminate for arbitrary $c: A \to \mathbb{R}_{\geq 0}$

• If capacities $c(a) \in \mathbb{Z}$, then $\exists$ an integer valued max flow.

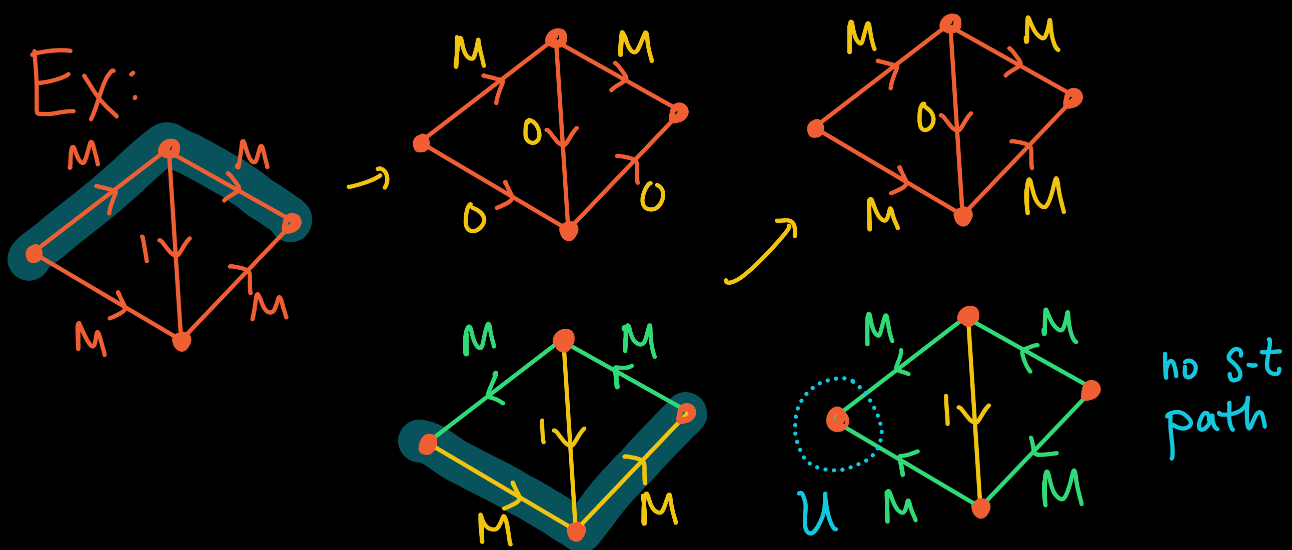
• If flow augmenting path is computed as the <u>shortest</u> s-t path in D_f then alg. runs in polynomial time ($O(|V| \cdot |A|^2)$ see §4.4) (improved to $O(|V| \cdot |A|)$ by Orlin 2013)

fewest edges

Max Flow Algorithm′ (Edmonds-Karp)

Always choose s-t path in D_f with fewest edges (computable in poly time with Dijkstra's alg., see §1.1)

Ex:



Claim: Edmonds-Karp terminates after $\leq 2|V| \cdot |A|$ augmentations. $\Longrightarrow$ run time $O(|V| \cdot |A|^2)$

<u>Main idea:</u> every edge $a \in A$ can appear as the bottleneck ($a \in P \cap A_f$, $\alpha = c(a) - f(a)$ or $\overleftarrow{a} \in P \cap B_f$, $\alpha = f(a)$) at most $|V|$ times.

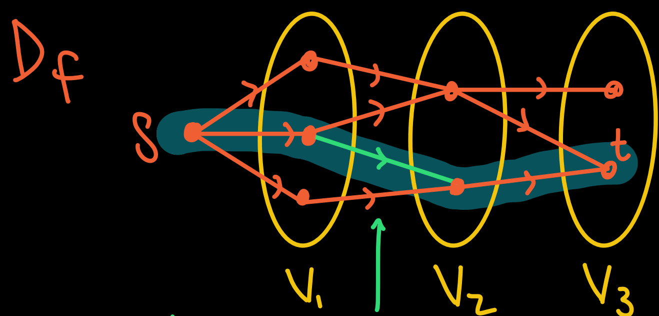
Step 1: Augmenting paths can only increase the distance between s and v in D_f

Step 2: If (v,w) was a bottleneck twice with flows f and later f', then s-v dist. in $D_f <$ s-v dist in $D_{f'}$

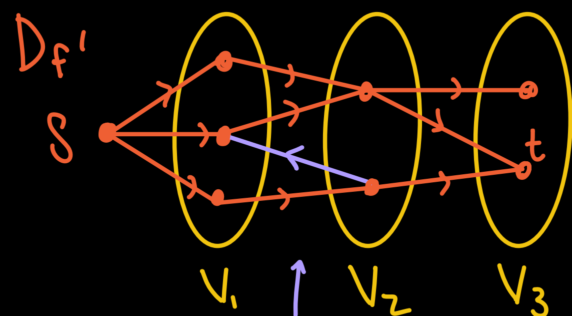
Since s-v dist. belongs to $\{1, \ldots, |V|-1, \infty\}$, it can strictly increase at most $|V|$ times

(Idea of Proof) $V_j = \{v \in V$ with s-v distance j in $D_f\}$

Chosen P path uses edges $V_j \to V_{j+1}$



bottleneck edge on P



augmenting by P reverses e in $G_{f'}$

Application 4.2 (Transportation problem)

m factories, n customers

Factory f_i can produce p_i tons of product/month

Customer c_j wants d_j tons of product/month

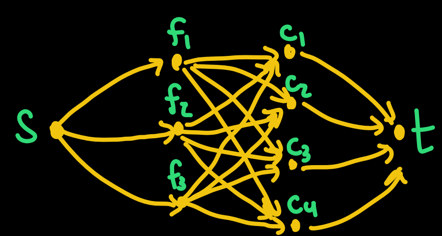
Can transport t_{ij} tons/month from factory f_i to customer c_j.

Can product demand be met?

Let $D=(V,A)$ with $V=\{s,t\}\cup\{f_1,\dots,f_m\}\cup\{c_1,\dots,c_n\}$

Arcs A: $s\to f_i$, $f_i\to c_j$, $c_j\to t$ for $i=1,\dots,m$, $j=1,\dots,n$

Capacity $c(s\to f_i)=p_i$, $c(f_i\to c_j)=t_{ij}$, $c(c_j\to t)=d_j$



Customer demand can be met $\iff \exists$ s-t flow under c of value $\sum_{j=1}^{n} d_j$