Recall: Edmonds-Karp repeatedly chooses augmenting path with fewest edges.

\[ d(v) = \min \# \text{edges in } s-v \text{ path in } G, \]

"Advancing edge" \((uv)\): \(d(v) = d(u) + 1\)

Main result from last time: \(d(v)\) never decreases.

From iteration \(i\) to iteration \(i+1\), it strictly increases unless there is a shortest path from \(s\) to \(v\) in \(G_f\) none of whose edges become saturated.

\[ f(\sum uv = c_{uv}) \]

Analyzing Edmonds-Karp: in every iteration at least one edge \((uv)\) becomes...
How many times can \((u,v)\) become saturated?

Whenever it becomes unsaturated after having been saturated, it means that \((v,u)\) was part of an augmenting path.

At that time \(d(n) = d(v) + 1\).

The next time \((u,v)\) gets saturated \((u,v)\) is on an augmenting path

\[ \Rightarrow \text{at that time } d(u) = d(v) + 1. \]

\[ d(u) \text{ increases by } \geq 2 \text{ between any pair of saturations} \Rightarrow \text{at most } \frac{n}{2} \text{ times}. \]
Graph has $n$ vertices, $m$ edges.
In $G_f$ there are $\leq 2m$ pairs that ever occur as edges.

$\implies \leq 2m \cdot \frac{\sqrt{m}}{\sqrt{2}} = mn$ iterations
One iteration takes $O(m)$.

Total running time $O(m \sqrt{n})$ arithmetic ops.
"Strongly polynomial": # of arithmetic operations bounded by $\text{poly}(n, m)$.

Dinitz's "blocking flow" algorithm

Def: A blocking flow from $s$ to $t$ in $G$ is a feasible flow that saturates at least one edge in every advancing $s-t$ path.
(advancing path $= $ path composed of advancing edges $= $ shortest $s-t$ path)

Dinitz Alg: $f = 0$

while $G_f$ contains an $s-t$ path

compute $f' = $ blocking flow in $G_f$
\[ f \leftarrow f + f' \quad \text{while} \quad \text{output} \quad f \]

Every loop iteration strictly increases \( d(t) \)
\[ \Rightarrow \text{at most} \quad n - 1 \text{ iterations} \]

To compute a blocking flow in \( O(mn) \)... let \( H = (V, E_{\text{adv}}) \) where
\[ E_{\text{adv}} = \{ \text{advancing edges in } G' \} \]

Initialize stack with \( \langle s \rangle \), \( f' = 0 \)

while \( V(H) \) non-empty
let \( u = \text{top of stack} \)
if \( u = t \) / stack is an aug path
increase \( f' \) on all edges of path until at least one is saturated.
pop vertices off stack until it contains no saturated edges.
\[ \text{elif } u \text{ has no outgoing edges, } \]
\[ \text{remove } u \text{ from } V(H) \]
\[ \text{remove all of } u\text{'s incoming edges from } E(H) \]
\[ \text{pop stack} \]
\[ \text{else there is an outgoing edge } (uv) \]
\[ \text{push } u \text{ onto stack} \]
\[ \text{endwhile} \]
\[ \text{output } f^? \]

\text{Invariant: if we delete } u, \text{ then every path } u \rightarrow t \text{ in } H \]
\[ f^!(v,w) = c(v,w). \]

\text{Running time: Every augmenting step takes } O(n), \text{ saturates at least one edge.} \]
\[ \Rightarrow O(mn) \text{ time on augmenting.} \]

\text{Removing vertices, edges: } O(mn) \text{ time in total.} \]
\[ \text{Stack height } 5n \Rightarrow \text{ at most } n-1 \text{ stack pushes take place before we} \]
either `augment` or `pop`

\[ O((m+n)n) \] stack pushes

each taking \( O(1) \) time.

in all, blocking flow takes

\[ O(mn + n^2 + m + n) \]

\[ = O(mn) \]

+ to compute blocking flow.

\[ \delta \] is

\[ O\left(\frac{n^2}{mn}\right) \]
\[ = O\left(\frac{m}{n}\right) \]