1 Oct 2021

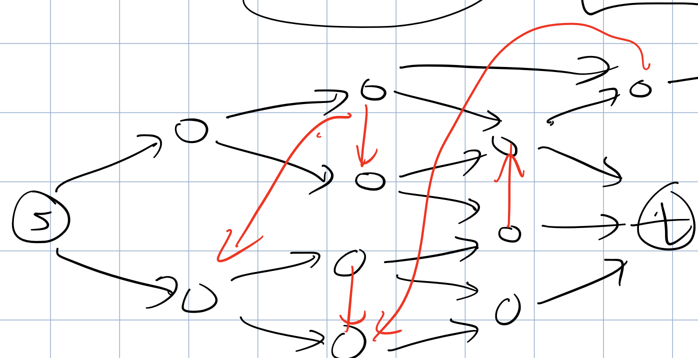
Dinitz Algorithm

Recall: Edmonds-Karp repeatedly chooses aug'l path with fewest edges.

$$d(v) \triangleq \text{min \# edges in } s-v \text{ path in } G_f.$$

"Advancing edge" (u,v): $d(v) = d(u)+1$

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. $\equiv \boxed{f(u,v) = c(u,v)}$



Black edges = advancing

Analyzing Edmonds-Karp: in every iteration at least one edge (u,v) becomes saturated.

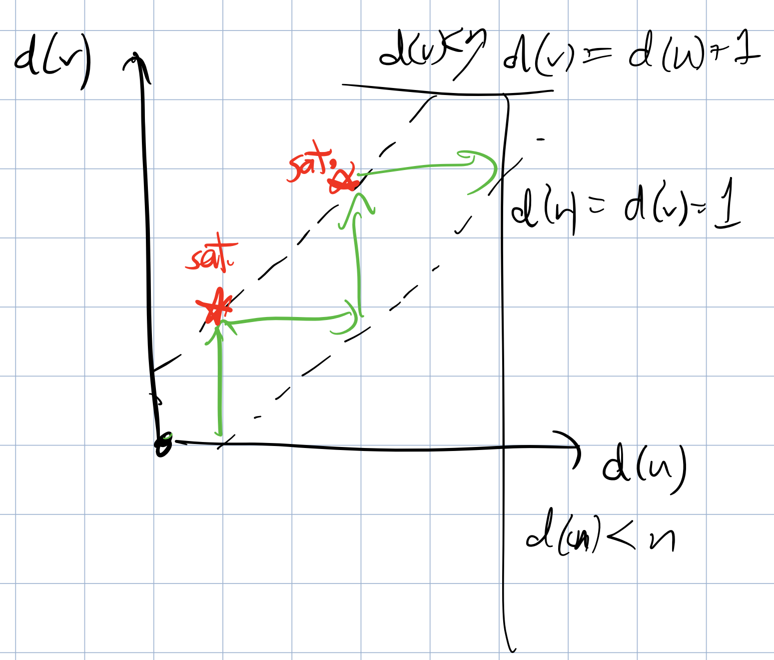
How many times can (u,v) become saturated?

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

At that time $d(u) = d(v) + 1$.

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

$\Longrightarrow$ at that time $d(v) = d(u) + 1$.



$d(u)$ increases by ≥ 2 between any pair of saturations $\Longrightarrow$ at most $\frac{n}{2}$ times.

Graph has n vertices, m edges.
In G_f there are $\leq 2m$ pairs that ever occur as edges.

$\Longrightarrow \leq 2m \cdot \left(\frac{n}{2}\right) = mn$ iterations

One iteration takes $O(m)$.

Total running time $O(m^2 n)$ arith. ops.
"Strongly polynomial": # of arithmetic operations bounded by 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 on 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

$f \leftarrow f + f'$

endwhile

output f

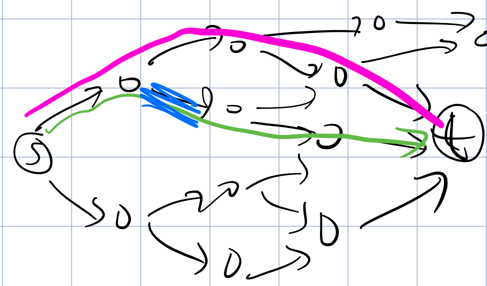
Every loop iteration strictly increases $d(t)$

$\implies$ at most $n-1$ iterations.

To compute a blocking flow in $O(mn)$...

let $H = (V, E_{adv})$ where

$E_{adv} = \{$ advancing edges in $G_f \}$.



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

while $V(H)$ non-empty

let u = 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

elif u has no outgoing edges
    remove u from $V(H)$
    remove all of u's incoming edges
        from $E(H)$
    pop stack
else there is an outgoing edge (u,v)
    push v onto stack
endwhile
output f'

Invariant: if we delete u, then every path $u \to t$ in H contains an edge (v,w) with $f'(v,w) = c_f(v,w)$.

Running time: Every augmenting step takes $O(n)$, saturates at least one edge.

$\Longrightarrow$ $O(mn)$ time on augmenting.

Removing vertices, edges:

$O(m+n)$ time in total.

Stack height $\leq n$

$\Longrightarrow$ at most $n-1$ stack pushes take place before we

either augment or pop.
$O((m+n) \cdot n)$ stack pushes
each taking $O(1)$ time.

in all, blocking flow takes

$$O(mn + n^2 + m + n)$$
$$= O(mn)$$

to compute blocking flow.

Dinitz is $O(mn^2)$

E-K $O(m^2 n)$