Network Flow and the Ford-Fulkerson Algorithm

Announcement: Prelim 1 will be administered online in at least two time slots. We will be running a poll to identify suitable time slots. There will be 15 minutes at the end to scan and upload your solutions.

Theme of network flow: Pack as many paths into a graph as possible.

Maximum # paths from s to t that we can pack into this graph without exceeding edge capacities?
At this point, to make further progress, we need to backtrack and re-route flow to free up extra capacity.

Would like to add flow on this brown path without overloading (x/t).

Can some of the flow on this path be rerouted to make room for brown?
Formalizing the problem.

A flow network is a directed graph $G$ with:
- vertex set $V$ having two special vertices $s$ (source), $t$ (sink)
- edge set $E$ having capacities $c(e) > 0$.
  (for this lecture $c(e)$ will be integers).
- every vertex belongs to at least one edge
- $s$ has no incoming edges, $t$ no outgoing.

A flow in a flow network is an assignment of a number $f(e)$ to each edge $e$ satisfying

- \[\text{[flow conservation]}\quad \text{if } v \notin \{s,t\} \text{ then } \sum_{e \text{ into } v} f(e) = \sum_{e \text{ from } v} f(e)\]

- \[\text{[capacity constraints]}\quad 0 \leq f(e) \leq c(e) \quad \forall e.\]

(Notion: if we pack paths from $s$ to $t$ into $G$ then $f(e)$ counts # paths using $e$.)**
Def. The value of a flow is the quantity
\[ \nu(F) = \sum_{e \text{out of } s} f(e) \]

The maximum flow problem is:
given flow network \( G, s, t, c \)
find a flow of maximum value.
In general, $G_f$ has two types of edges:

- **Forward edge** $(uv)$ if $e = (uv) \in E(G)$ satisfies $f(e) < c(e)$ (residual capacity $c(e) - f(e)$).

- **Backward edge** $(vu)$ if $e = (vu) \in E(G)$ satisfies $f(e) > 0$ (residual can, $f(e)$).

Where is the unused capacity? Where can we free up capacity by removing flow? [signify using reversed edges]
Augmenting a path \( P \)

If \( G \) is a flow network
\( f \) is a flow
\( G_f \) is the residual graph
\( P \) is an \( s-t \) path in \( G_f \)

\( P \) is called an augmenting path for \( f \)

\[ \text{AUGMENT} (G, f, P) : \text{modify } f \text{ using } P \text{ to increase flow value.} \]

let \( b(f, P) = \min \{ \text{residual cap of } e \mid e \in P \} \)

"bottleneck value of \( P \)"

for each edge \( (u, v) \in P \):

if \( e = (u, v) \) is a forward edge:
\[ f'(e) = f(e) + b(f, P) \]

if \( (u, v) \) is a backward edge:
\[ \text{let } e = (v, u) \]
\[ f'(e) = f(e) - b(f, P) \]

for each edge \( e \) not in \( P \):
\[ f'(e) = f(e) \]

return \( f' \).
This procedure modifies $f$ to a new flow $f'$ whose value is
\[ \nu(f') = \nu(f) + b(f,P). \]