Weighted, directed graph.

Edge weights = cost
- Paths in series: +
- Paths in parallel: min

Edge weights = capacity
- Paths in series: min (bottleneck)
- Paths in parallel: max
  - ↳ pick single path → Dijkstra, Floyd-Warshall
  - OR: + ↳ pick both paths → max flow

Flow network = weighted, directed graph
- one source vertex + one target vertex



Flow = f(e) on each edge, ≤ capacity c(e)

flow in = flow out @ each vertex.

---

every edge has some remaining capacity.

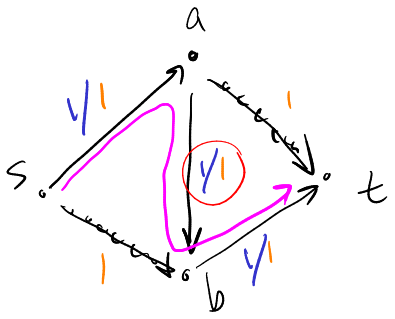
GREEDYFLOW(G):

while there is an unsaturated path P from s → t:

α ← find min remaining capacity of any edge in P

Add α to the flow on every edge of P.

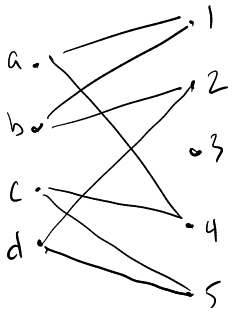




Max flow applications!

① Bipartite matching.

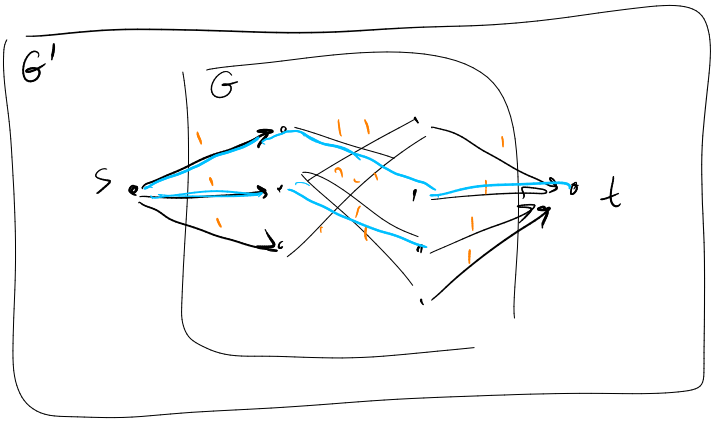
Recall bipartite graph:



Matching = set of edges such that no 2 edges share an endpoint.

Q: size of a maximum matching?

Idea:

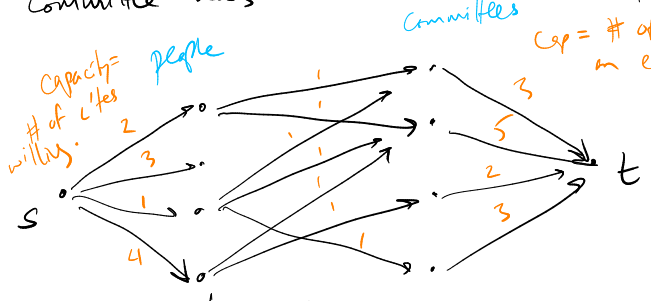


Claim: valid flows of value k in G' are in 1-1 correspondence with valid matchings of size k in G.

Corollary: a max flow $\leftrightarrow$ max matching.

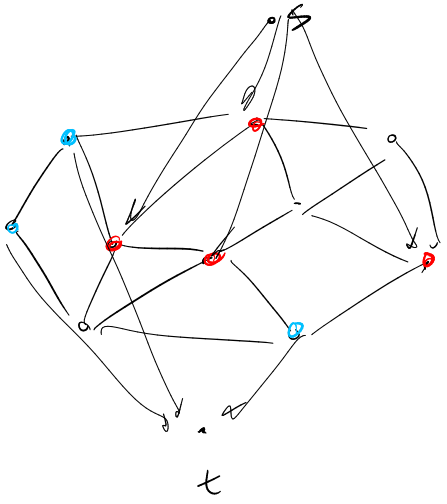
② Committee assignment problem

- People
- Committees
- Each person has
  - certain committees they are willing to be on
  - Max # of committees
- Each committee has an ideal # of people.



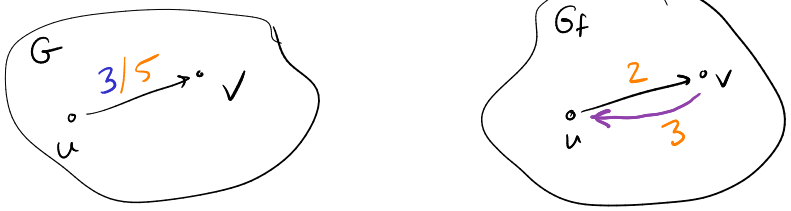
edge $p \to c$ = p willing to be on c

Evacuation planning:

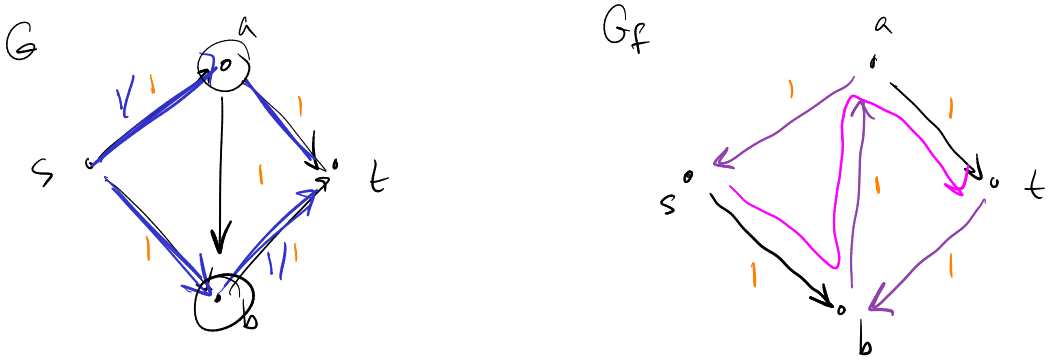


Def'n Given a flow network G with a flow f, the residual network G_f is a weighted, directed graph defined as follows:

- The vertices of G_f are the same as the vertices of G.
- For every edge $u \xrightarrow{e} v$ which is not saturated (i.e. $f(e) < c(e)$), we add an edge $u \to v$ in G_f with weight $(c(e) - f(e))$.



- For every edge $u \xrightarrow{e} v$ with $f(e) > 0$, we add a "backwards" edge $u \leftarrow v$ with weight $f(e)$.



FORD–FULKERSON(G):

while there is any path P from $s \to t$ in G_f:

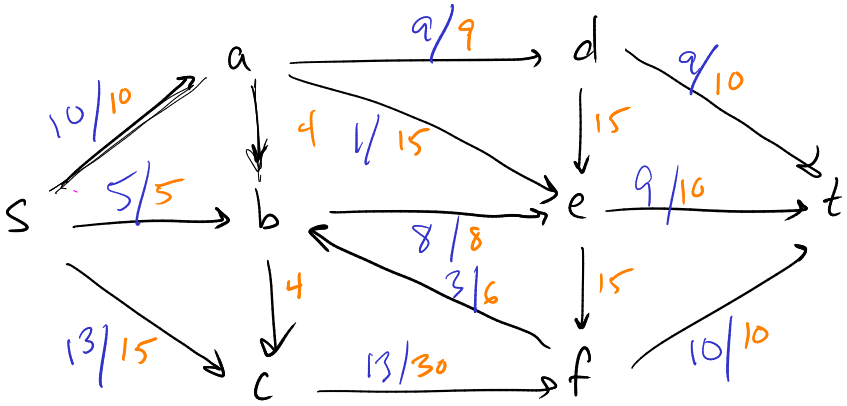
- $\alpha \leftarrow$ min weight along P in G_f
- for each edge $e \in P$:
  - if e is a "forwards edge", add α to flow on corresponding edge in G
  - if e is a "backwards edge", subtract α " " " "

$\Theta(E)$. worst-case

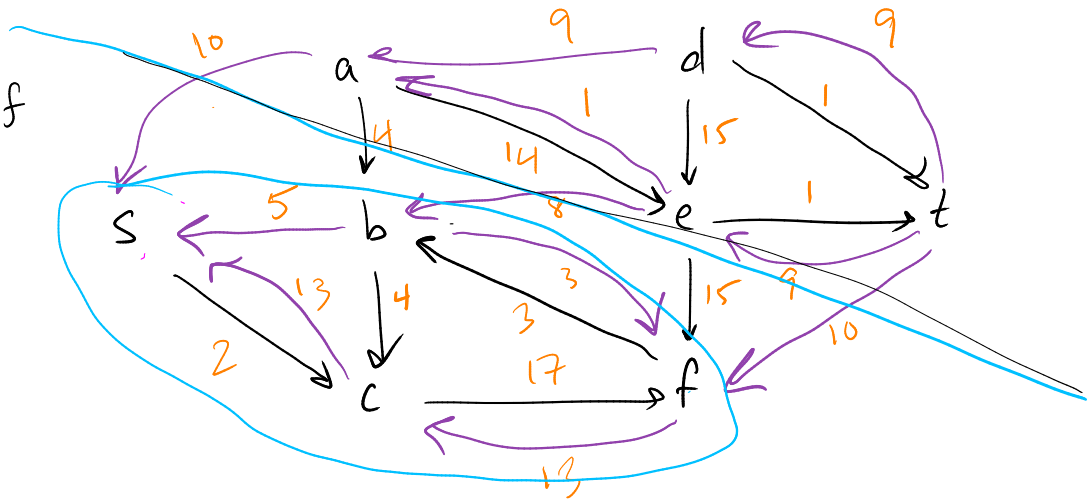
← time to find P: $\Theta(V+E)$?

# of loops bounded by capacity from s.

G



G_f

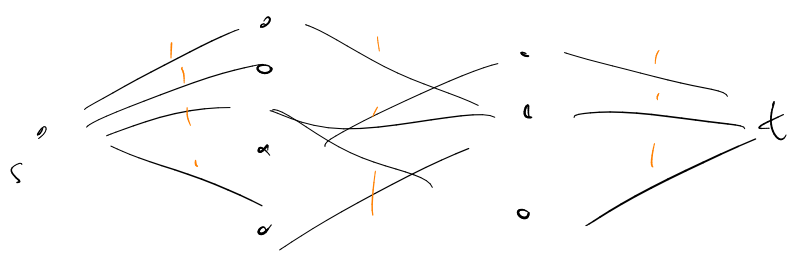


How long does Ford-Fulkerson take?

- At most C loops where C = capacity out of s.
- Each loop takes $\Theta(V+E) + \Theta(E)$ time to find path + update flow + G_f.

Hence F-F takes $O(CE)$.

e.g. bipartite matching:



$C = O(V)$

Hence F-F takes

$O(CE) = O(VE)$.

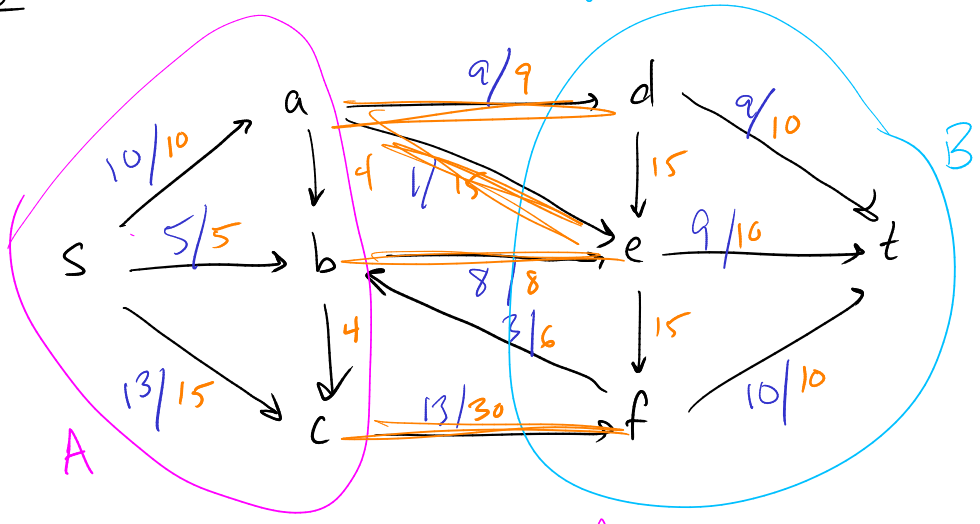
Defn Given a flow network $G = (V, E)$, an **s-t cut** is a partition of V into two sets A and B such that $s \in A$ and $t \in B$.

Defn The **capacity** of an s-t cut is

$$c(A, B) = \sum_{e : A \to B} c(e).$$

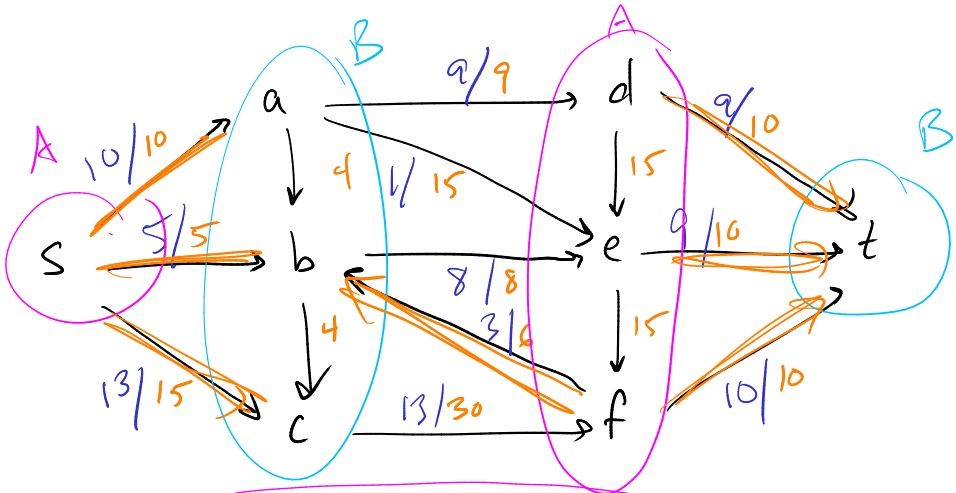
all edges from A to B.

eg.



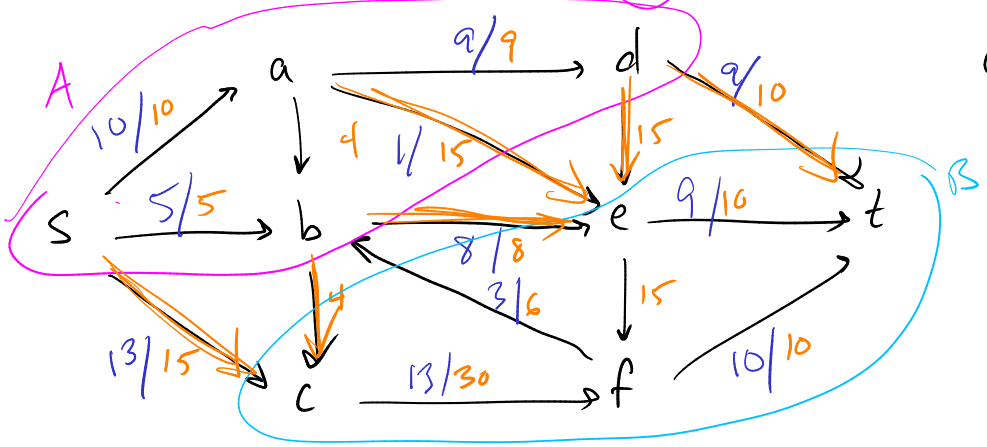
$c(A,B) = 9 + 15 + 8 + 30 = 62$

$v(A,B) = 9 + 1 + 8 + 13 - 3 = 28.$



$c(A,B) = 10 + 5 + 15 + 6 + 10 + 10 + 10 = 66$

$v(A,B) = 10 + 5 + 13 + 3 + 9 + 9 + 10 - 9 - 1 - 8 - 13 = 28$



$c(A,B) = 15 + 4 + 8 + 15 + 15 + 10 = 67$

$v(A,B) = 13 + 8 + 1 + 9 - 3 = 28.$

Defn A **minimum cut** is an s-t cut with **smallest possible capacity**.

Defn Given a flow network and a flow f and an s-t cut (A, B), the **net flow** of (A, B) with respect to f is

$$v(A, B) = \left[\sum_{e: A \to B} f(e)\right] - \left[\sum_{e: B \to A} f(e)\right]$$

Lemma (Flow value lemma)

The net flow for any s-t cut equals the value of the flow,

$$v(A, B) = v(f).$$

Proof

$$\begin{aligned} v(f) &= f^{out}(s) && \{\text{definition}\} \\ &= \sum_{v \in A} \left(f^{out}(v) - f^{in}(v)\right) && \{0 \text{ for every } v \in A \text{ except } s\}. \\ &= \sum_{e: A \to B} f(e) - \sum_{e: B \to A} f(e) \\ &= v(A, B) \end{aligned}$$

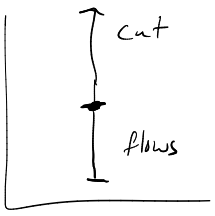
Lemma (Bottleneck lemma) Let f be any flow and (A, B) any s-t cut. Then

$$v(f) \leq c(A, B).$$

Proof.

$$\begin{aligned} v(f) &= v(A, B) && \{\text{flow value lemma}\} \\ &= \sum_{e: A \to B} f(e) - \sum_{e: B \to A} f(e) \\ &\leq \sum_{e: A \to B} f(e) && \{\text{don't subtract} \to \text{bigger}\} \\ &\leq \sum_{e: A \to B} c(e) && \{\text{valid flow} \to f(e) \leq c(e)\}. \\ &= c(A, B) && \{\text{defn}\}. \end{aligned}$$

Corollary. If $v(f) = c(A,B)$, then f is a max flow and (A,B) is a min cut.



Theorem (max flow/min cut). Let f be any flow on network G. The following are equivalent:

1. $v(f) = c(A,B)$ for some s-t cut (A,B).
2. f is a max flow.
3. There are no $s \to t$ paths in the residual network G_f.

Proof.

(1) $\Rightarrow$ (2) : this is the corollary from above.

(2) $\Rightarrow$ (3) : ie. if f max flow, then there are no $s \to t$ paths in G_f. The contrapositive is: if there is an $s \to t$ path in G_f, then f is not a max flow.

An $s \to t$ path in G_f is an augmenting path — we can add (+ subtract) flow along the path to increase the value of the flows, so f is not a max flow.

(3) $\Rightarrow$ (1) ie. no $s \to t$ paths in G_f, then $v(f) = c(A,B)$ for some s-t cut (A,B).

Let A be the set of all vertices reachable from s, and $B = V - A$. Claim: every edge from $A \to B$ is full and every edge $B \to A$ is empty.

- If an edge $A \to B$ were not full, there would be a forwards edge in the residual network, but by defn we cannot reach B from A.
- If an edge $B \to A$ were not empty, there would be backwards edge.

Therefore $v(f) = v(A,B) = \underbrace{\sum_{e: A \to B} f(e)}_{\text{full}} - \underbrace{\sum_{e: B \to A} f(e)}_{\text{empty.}}$

$$= \sum_{e: A \to B} c(e) - 0$$

$$= c(A,B).$$