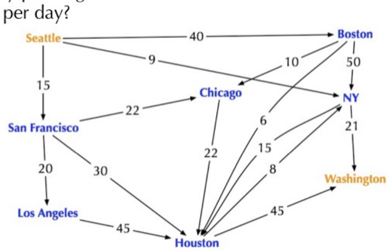
Maximal Flow

Problem

Maximum Flow

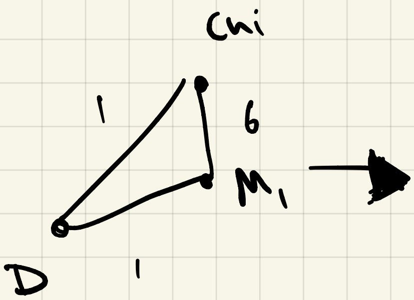
- Airline has a network of city-to-city routes
- Each route labeled w/ max number passengers that can be sent per day
- Q: How many passengers could be sent from Seattle to Washington per day?



Seth Pettie

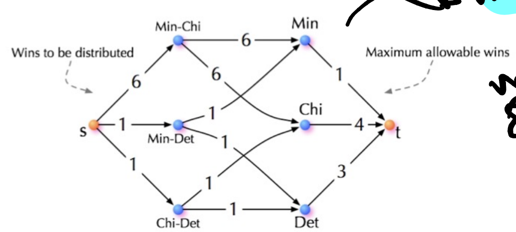
Maximum Flow

- Input: "flow network" (directed graph) $G = (V, E, c)$
  - V — vertices
  - $E \subseteq V \times V$ — edges
  - $c : E \to \mathbb{Z}^+$ — *capacity* function → rate at which material passes through edge
  - Two distinguished vertices $s, t \in V$ (source s, sink t)
- $c(u, v)$ = the rate that material can "flow" from u to v
- The problem: what is the maximum rate that material can flow from s to t?



- Model this problem as a flow network:
  - 1 unit of flow = 1 win
  - c(s, Min-Chi) = #(Minnesota-Chicago) games to be played
  - c(Chi, t) = maximum # allowable wins for Chicago

→ max flow here is 7



Seth Pettie

req 8 game distribution

Flow

$f : V \times V \to \mathbb{Z}^+$ is a flow if:

↳ Capacity: $f(u,v) \le c(u,v)$

↳ Conservation: $\forall u \in V \setminus \{s,t\}$ ~~$\sum_{(x,u)\in E} f(x,u) = \sum_{(u,x)\in E} f(u,x)$~~

$f(u,v) = -f(v,u)$ ← skew sym

$\sum_{(u,x)\in E} f(u,x) = 0$

$V(f)$ = total flow leaving s = total flow entering t.

Seth Pettie

Iterative Improvement

- The Ford-Fulkerson Method:
  - Find a path from s to t
  - Push as much flow through this path as possible
  - Find another path from s to t with *free capacity* on all the edges
  - Push as much flow through….

subtract 2 from all edges in path and rinse & repeat

- We have to be careful in implementing this idea! (What does "*free capacity*" mean?)

↳ if a path 100 → 2 → 5 → 8

↳ bottleneck

flow to this path is 2

Seth Pettie

Residual Networks

and with reversed edges

- A flow f in G defines a *residual network* G_f
  - $c_f(u,v)$ represents the amount of additional flow that could be sent through (u,v) without violating the capacity constraint
  - $c_f(u,v) = c(u,v) - f(u,v)$
  - Only edges w/ ***positive*** capacity appear in G_f

Ford-Fulkerson Method:

- Start w/ initial flow $f(u,v) = 0$ for all u, v $(G_f = G)$
- Repeat: ← How many times?
  - Find path P from s to t in G_f ← *How?*
  - $f' :=$ maximum legal flow along P
  - $f := f + f'$ ← Does adding these flows make sense?
- Until there is no path from s to t in G_f ← Does this eventually happen?
- Return f ← Is f necessarily a maximum flow?

Adding flows is legal

- CLAIM
  - Let f be a legal flow in G and f' be a legal flow in G_f
  - Let $f'' = f + f'$ i.e. $(f''(u,v) = f(u,v) + f'(u,v))$
  - Then f'' is a legal flow in G and $v(f'') = v(f) + v(f')$

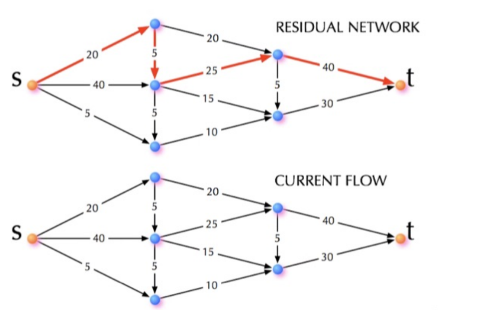
f'' should abide capacity, symmetry, flow cons

$$f''(u,v) = f'(u,v) + f(u,v)$$
$$\le \underbrace{(c(u,v) - f(u,v))}_{\text{flow cap const. in } G_f} + f(u,v) \le c(u,v) \quad \triangleright.$$

e.g. initially $G_f = G$

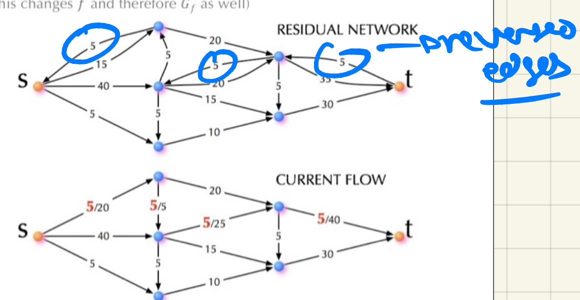
Ford-Fulkerson in Action

- Find an s-t path in G_f (using depth first search)



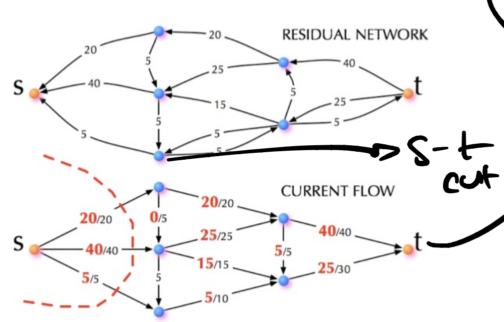
Ford-Fulkerson in Action

- Add as much flow as possible through the path (5)
  (This changes f and therefore G_f as well)



Ford-Fulkerson in Action

- All edges leaving s are filled to capacity!

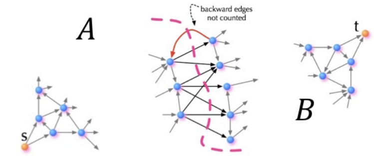


▷ this is optimal since everything that goes out of s goes into t (65)

S-T cut

s-t Cuts

- (A, B) is an ***s-t* cut** if $s \in A$ and $t \in B$ and $A \cup B = V$.
- $c(A, B)$ = total capacity of edges from A to B
- For any flow f and *s-t* cut (A, B), $v(f) \le c(A, B)$.



Seth Pettie

TFAE

Max-Flow — Min-Cut Theorem

- The following are equivalent:
  (1) f is a maximum flow in G
  (2) There is no path from s to t in G_f
  (3) $v(f) = c(A, B)$ for some *s-t* cut (A, B)

Will show
(1) => (2) by contrapos
(2) => (3)
(3) => (1)

(1) => (2)
!(2) => !(1) as we can add on a flow in G_f to f increases combined flow

(3) => (1) As capacity of every s-t cut upper bounds every s-t flow!

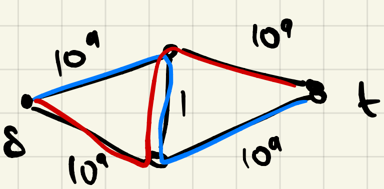
(2) => (3) look at 0-capacity edges in G_f ←
(i) i.e. let $A \to$ DFS edges in G_f source s (reachable from s)
(ii) $B = V \setminus A$
(iii) $\forall (u,v) \in out(A,B)$ $f(u,v) = c(u,v)$
$\forall (u,v)$ st $u \in B, v \in A$ $f(u,v) = 0$
$v(f) = c(A,B)$

→ flow-cut duality.

Efficiency of Ford-Fulkerson

- $O(|E|)$ time to find a path in G_f w/ depth first search.
- Each augmenting path increases f by ≥ 1
  (Remember we assumed int. capacities. This is not true in general!)
  $\Rightarrow$ At most $O(|E| \cdot v(f))$ time in total
- Could it ever actually be this bad?

Yes! it can consider



doing blue, red... is dumb.
takes $O(10^9 |E|)$ time.

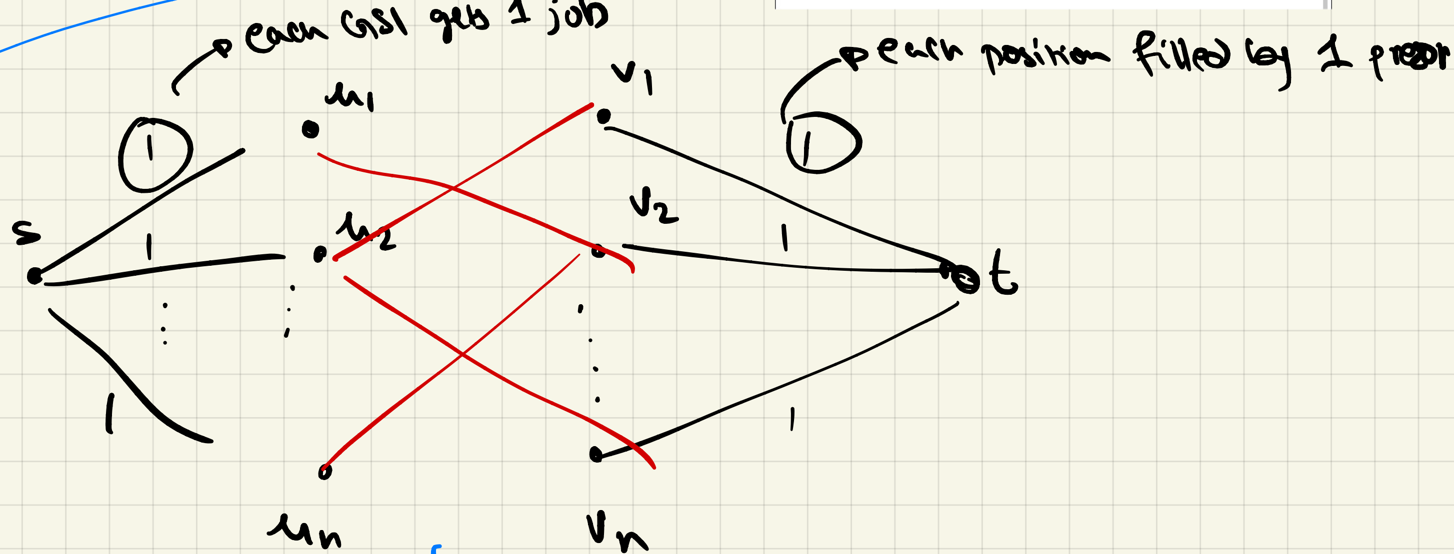
e.g. 1

Maximum Matching

- n GSIs, n GSI positions. Each GSI is qualified for a ***subset*** of the positions.
- **Problem:** find a maximum-size matching from GSIs to GSI positions.
- Create *undirected* graph: $G = (V, E)$ → GSI
  - $V = \{u_1, \dots, u_n\} \cup \{v_1, \dots, v_n\}$ → position
  - $E = \{\{u_i, v_j\} \;:\; \text{GSI } u_i \text{ is qualified for position } v_j\}$.

Maximum Matching

- n GSIs, n GSI positions. Each GSI is qualified for a ***subset*** of the positions.
- **Problem:** find a maximum-size matching from GSIs to GSI positions.
- Transform into *directed* flow graph $G = (V, E, c)$
  - $V = \{u_1, \dots, u_n\} \cup \{v_1, \dots, v_n\} \cup \{s, t\}$
  - $E = \{(u_i, v_j) \;:\; \text{GSI } u_i \text{ is qualified for position } v_j\}$. $\cup \{(s, u_i), (v_i, t) \;:\; 1 \le i \le n\}$
  - $c(s, u_i) = c(v_i, t) = 1$ for all edge $(s, u_i), (v_i, t) \in E$.
  - $c(u_i, v_j) = \infty$ for all other edges $(u_i, v_j) \in E$.



→ any capacity above 1 for these is good. But use ∞ for now.

OBS: if $c : E \to \mathbb{Z}^+$ Then Ford Fulkerson will only give integer flows

→ Above, assignments require integer flow
→ can have a GSI teach $1/2$ of a course although thats a legal flow

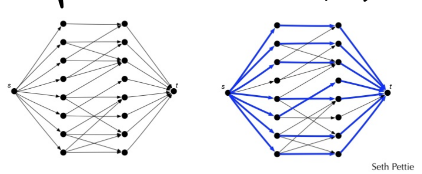
→ If find perfect matching the $s \to u_i$, $v_j \to t$ both give a minimal s-t cut
→ flow is n!

→ else No perfect matching → infinite edge capacity shows that no edge exists from a cut.

Consequences of Max-Flow Min-Cut

- **Hall's Thm:** Let $G = (V, E)$ be a bipartite graph. If G does not have a perfect matching, then $\exists S \subset V.\ |N(S)| < |S|$. $\quad N(S) = \{v : u \in S, \{u, v\} \in E\}$

$\exists$ perf match $\iff |N(S)| \ge |S|$



Seth Pettie

Making Ford-Fulkerson run forever

- The ***maximum flow*** is ***finite***
- We want Ford-Fulkerson to make an ***infinite number of augmentations***. How?

→ might never halt

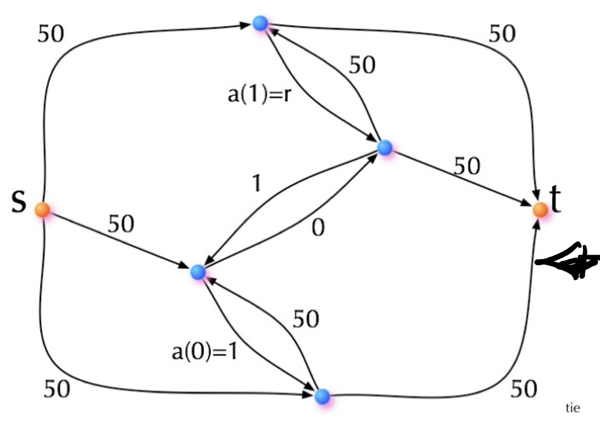
- One way:
  - 1^{st} augmentation adds 1 unit of flow
  - 2^{nd} augmentation adds 0.5 units of flow
  - 3^{rd} augmentation adds 0.25 units of flow
  - …

will use a Fibonacci-like seq.

- A Fibonacci-like sequence:
  - $a(0) = 1$
  - $a(1) = r$
  - $a(n) = a(n-2) - a(n-1)$ for $n \geq 2$
- We want to ***choose r*** so that $a(n) = r^n < 1$
  - $a(0) = 1 = r^0$ ✓
  - $a(1) = r = r^1$ ✓
  - By definition: $a(n) = a(n-2) - a(n-1)$
  - We need: $r^n = r^{n-2} - r^{n-1}$ or $r^2 = 1 - r$
  - Choose: $r = \frac{\sqrt{5}-1}{2} \approx 0.618$

Seth Pettie

IDEA: Make Ford-Fulkerson Compute a(0),a(1),a(2),…



bad graph