chang@umbc.edu

Network Flow Applications

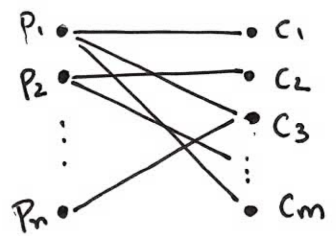
- Bipartite Matching
- Project Selection
- Baseball Elimination
- Assignment Problem (w/ lower bounds)

Network Flow Application: Bipartite Matching.

Customers: $p_1, \dots, p_n$

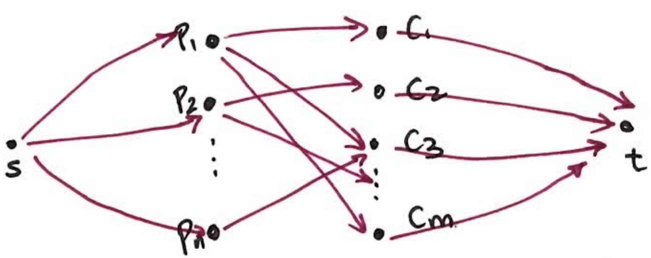
Cars: $c_1, \dots, c_m$

Edge from p_i to c_j indicates that p_i is interested in c_j and can afford it.

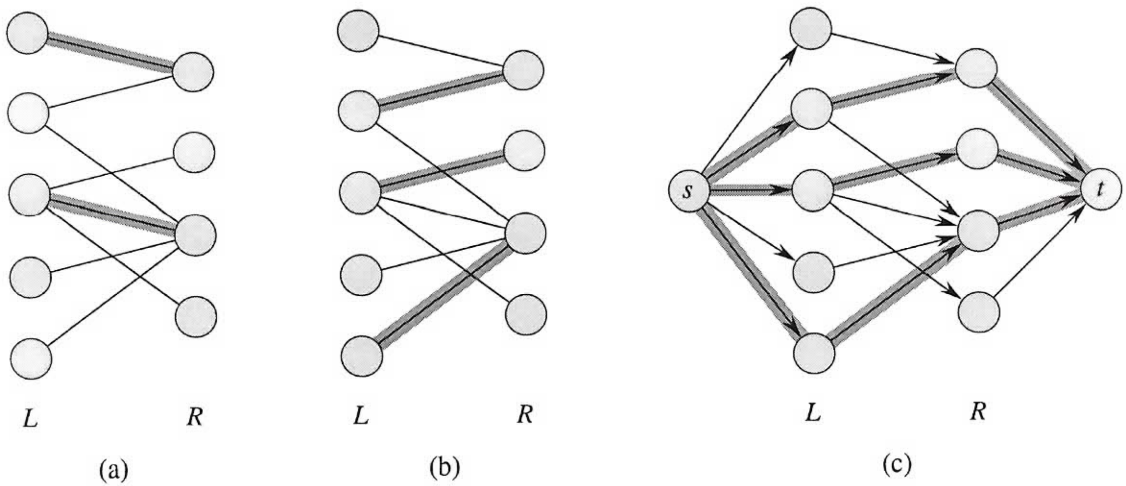


<u>Claim 1:</u> If there is a way to sell k cars to k customers, then there is a flow f with $|f| = k$

<u>Claim 2:</u> If there is an integer flow with $|f| = k$, then there is a way to sell k cars to k customers.



all edges have capacity = 1


L
R
(a)
L
R
(b)
s
t
L
R
(c)

Project Selection Problem:

Select projects to maximize profit subject to dependency constraints

Set of projects $P_1, \ldots, P_n$.

make money — lose money

Projects either generate revenue or incur expenses.

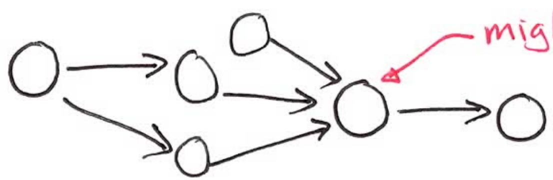
Projects have dependencies, given as a DAG

$P_i \rightarrow P_j$ means P_i depends on P_j

do P_j before P_i

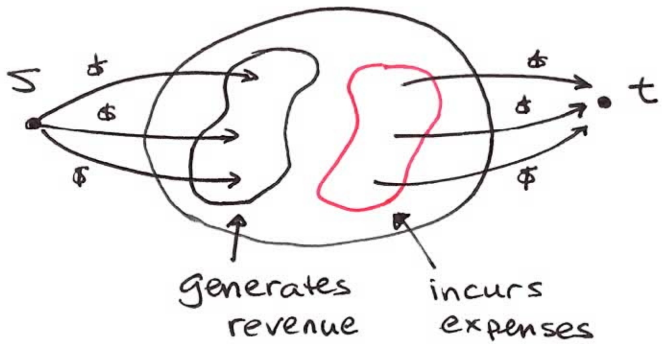
& must pick P_j if you pick P_i

Multiple dependencies allowed



might want to select this one, even if it incurs expenses

Convert into a max flow problem



For each revenue generator P_i, add edge (s, P_i).
set $c(s, P_i) = \text{revenue}(P_i)$.

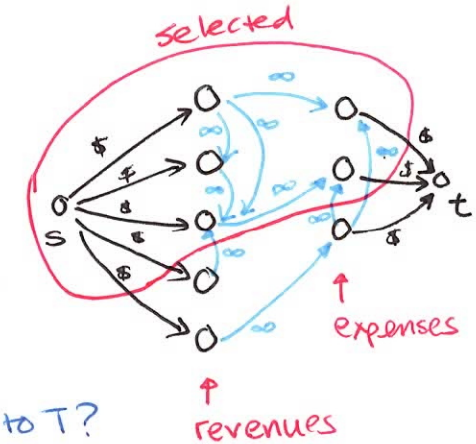
For each ~~expense incurrer~~ money loser P_i, add edge (P_i, t)
set $c(P_i, t) = \text{expense}(P_i)$.

For each dependency (P_i, P_j), set $c(P_i, P_j) = \infty$.

Compute max flow of network.

Let (S,T) be the min cut found.

Selected projects $= S - \{s\}$.



<u>Claim:</u> selected projects maximize profit

Which edges can cross the cut from S to T?

None of the dependency edges with ∞ capacity.

(Cannot have ∞ in min cut.)

so selected projects do not depend on projects not selected.

$c(S,T)$ = revenues of projects not selected + exp. of projects selected

R = sum of all revenues

= R − revenues of projects selected + expenses of projects selected

= R − profit

∴ minimize cut = maximize profit

Baseball Elimination [Kleinberg-Tardos]

Current Standing

New York	92
Baltimore	91
Toronto	91
Boston	90

5 Games left

NY v. B'more
NY v. Toronto
B'more v. Toronto
B'more v. Boston
Toronto v. Boston

Boston can't win or tie for first place:

- must win its last 2 games to get 92 pts and tie NY
- NY must lose its 2 games v.s. B'more & Toronto
- Then, B'more & Toronto each has 92 pts
- But, one of B'more & Toronto will win game vs each other and have 93 pts

Baseball Elimination [Kleinberg-Tardos]

Current Standing	
New York	92
Baltimore	91
Toronto	91
Boston	90

5 Games left

- NY v. B'more
- NY v. Toronto
- B'more v. Toronto
- B'more v. Boston
- Toronto v. Boston

Simpler explanation:

- Boston can get a maximum of 90 + 2 = 92 pts
- NY, B'more & Toronto already have 92+91+91 = ~~272~~ 274 pts
- There are 3 games w/o Boston remaining
- One of NY, B'more & Toronto will have > (~~272~~ 274 + 3) ÷ 3 = 92.33.. pts

Example 2:

New York	90
Baltimore	88
Toronto	87
Boston	79

Remaining Games:

Boston v. NY 4x	B'more v. NY 1x
Boston v. B'more 4x	B'more v. Toronto 1x
Boston v. Toronto 4x	NY v. Toronto 6x

You can always find a "short" explanation, but must consider subsets of teams. E.g. Can Boston win?

NY + B'more + Toronto = 90+88+87 = 265

+ 8 remaining game w/o Boston = 273

273 ÷ 3 = 91 ← doesn't tell us much

max for Boston

= 79+12

= 91

Boston cannot win

NY + Toronto = 90+87 = 177

+ 6 remaining games = 177+6 = 183

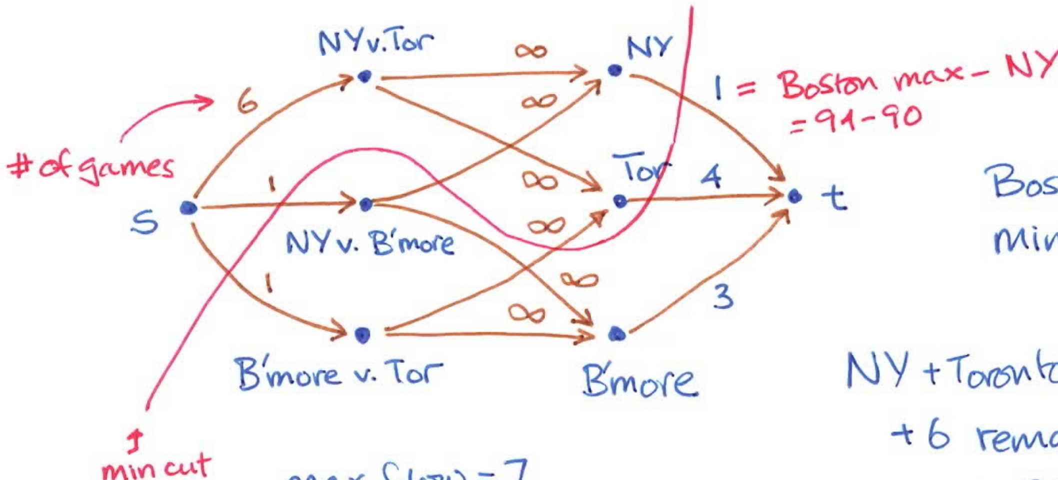
183 ÷ 2 = 91.5

Example 2:

New York	90
Baltimore	88
Toronto	87
Boston	79

Remaining Games:

Boston v. NY 4x	B'more v. NY 1x
Boston v. B'more 4x	B'more v. Toronto 1x
Boston v. Toronto 4x	NY v. Toronto 6x



max flow = 7

Can Boston win?

Boston's max $= 79 + 12 = 91$

min cut has capacity 7

$NY + Toronto = 90 + 87 = 177$

$+ 6$ remaining games $= 183$

$183 \div 2 = 91.5$

Boston cannot win

Example 2:

New York	90
Baltimore	88
Toronto	87
Boston	79

Remaining Games:

Boston v. NY 4x

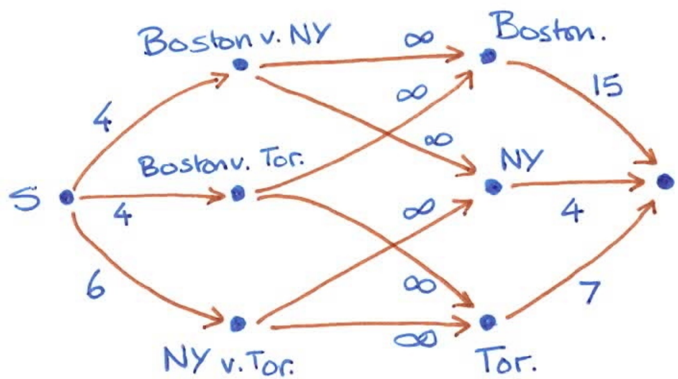
Boston v. B'more 4x

Boston v. Toronto 4x

B'more v. NY 1x

B'more v. Toronto 1x

NY v. Toronto 6x



max flow = 14

Baltimore can win:

pts = 88 + 6pts = 94 pts

Boston wins all its games

pts = 79 + 8 = 87 pts

Toronto wins all its games v. NY

pts = 87 + 6 = 93 pts

NY stays at 90pts

Z = team under consideration

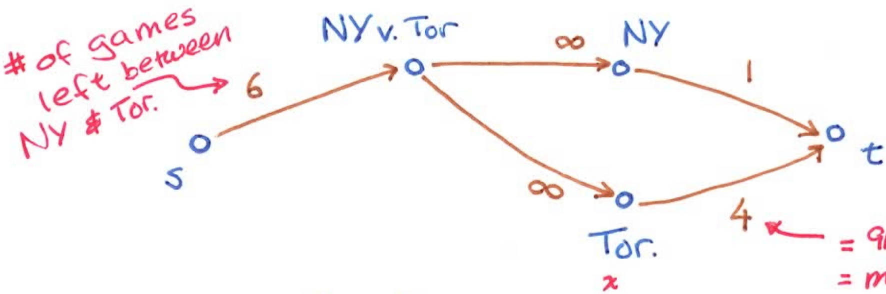
m = maximum possible pts for z

= current pts + # of remaining games

assume that z wins all of these

g^* = # of games not involving z

Idea: distribute pts from game to teams



Z = Boston

$m = 79 + 12 = 91$

$= 91 - 87$

= max number of pts Tor can win without beating Boston's max

$= m - w_x$

x = some other team

w_x = current points of x

g_{xy} = # of game left between team x & team y

General Construction:

Vertices: 1 vertex v_x for team x

1 vertex u_{xy} for pair of teams x & y

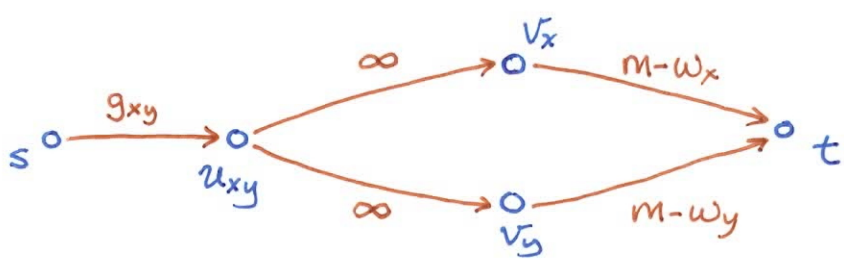
source s & sink t

Edges: $c(s, u_{xy}) = g_{xy}$ = # of games left between x & y

$c(u_{xy}, v_x) = \infty$

$c(u_{xy}, v_y) = \infty$

$c(v_x, t) = m - w_x$ = # of games x can win & not beat z



$g^* =$ number of games left not involving z.

Question: Is max flow $= g^*$?

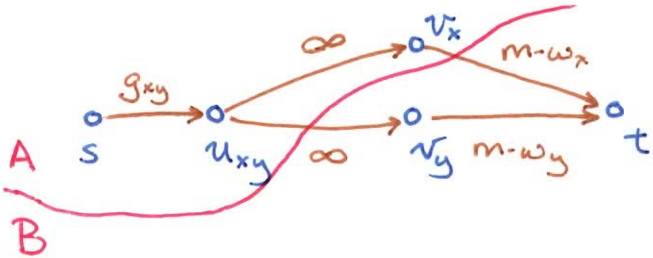
Yes = z can win or tie for 1st place
because pts from remaining games
can be distributed to teams
without any team exceeding z's max. pts.

No = z cannot win ... because?

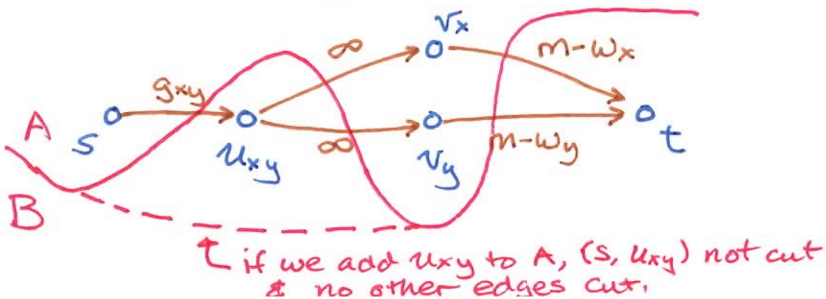
There is always a short proof.

Let (A, B) be the mincut:

- Cannot have $u_{xy} \in A$, but $v_x \notin A$ or $v_y \notin A$. An edge with ∞ capacity would cross (A, B)

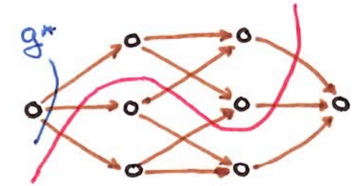


- If $v_x \in A$ & $v_y \in A$, then $u_{xy} \notin A$ means (A, B) not mincut.



$(A \cup \{u_{xy}\}, B - \{u_{xy}\})$ is a smaller cut, because (s, u_{xy}) not cut.

Let $T = \{x \mid v_x \in A\}$ = teams on A side of mincut.



$$c(A,B) = \sum_{\{x,y\}\not\subseteq T} g_{xy} + \sum_{x\in T} (m - w_x)$$

$$= g^* - \sum_{\{x,y\}\subseteq T} g_{xy} + m|T| - \sum_{x\in T} w_x$$

Then, $$\sum_{x\in T} w_x + \sum_{\{x,y\}\subseteq T} g_{xy} = m|T| + \overbrace{g^* - c(A,B)}^{>0} > m|T|$$

cut around s

min cut

So, $$\frac{1}{|T|}\left(\sum_{x\in T} w_x + \sum_{\{x,y\}\subseteq T} g_{xy} \right) > m.$$

there exists a subset of teams T

average over T

points teams in T already have

points from games left

max pts z can have

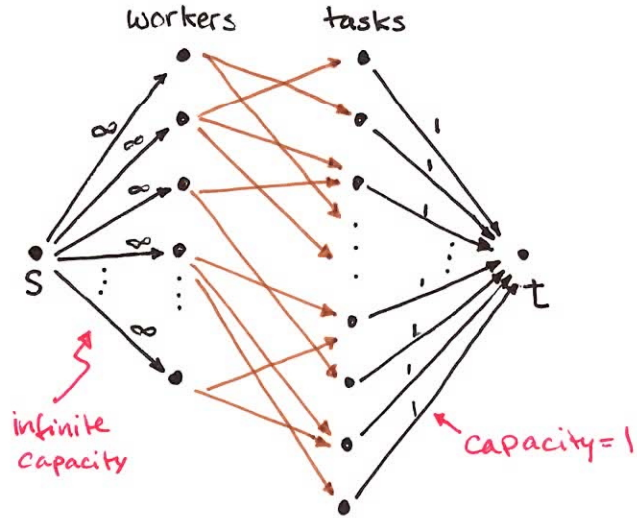
Assignment Problem: Basic Version

n workers: $w_1, w_2, \ldots, w_n$
m tasks: $t_1, t_2, \ldots, t_m$

Each worker is competent for a subset of the tasks.

Assign workers to tasks so each task is assigned a competent worker.

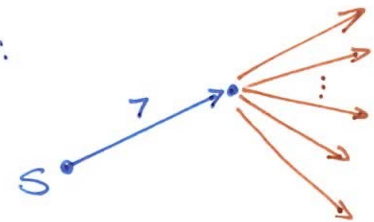
Q: does max flow saturate all edges into t?



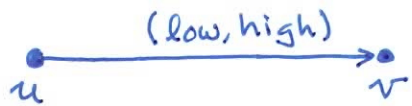
edge (w_i, t_j)
if worker i is competent in task j
capacity = 1

What if we want to prevent an individual worker from performing too many or too few tasks?

Limit worker to 7 tasks:



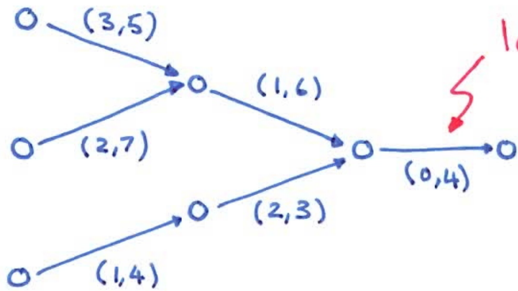
Need flow networks with lower bounds if we want to require a worker perform at least some number of tasks.



$low(u,v) \leq f(u,v) \leq high(u,v)$

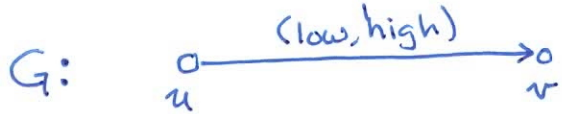
same as capacity

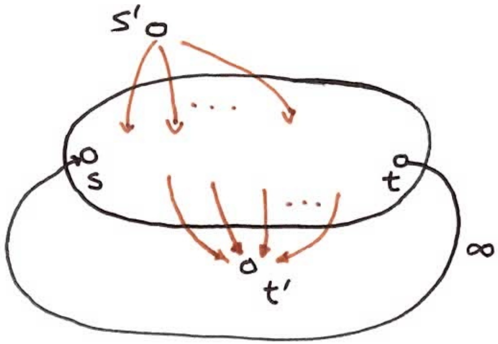
Some flow networks with lower bounds have no solution:



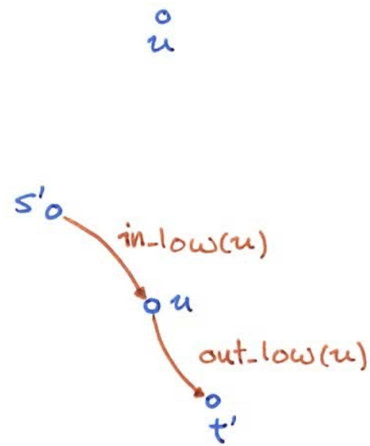
lower bounds on other edges require that at least 7 units flow thru this edge, but that exceeds the edge's capacity

Idea: convert flow networks w/ lower bounds into normal flow networks.





Create new source s'
& new sink t'
add edge w/ ∞ cap. between t & s

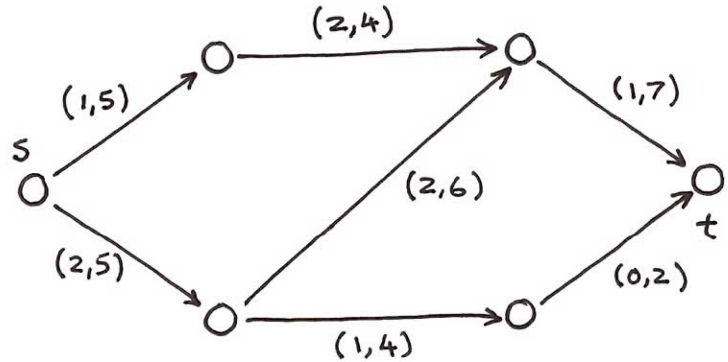


$$\text{in\_low}(u) = \sum_{v \in V} \text{low}(v,u)$$

= sum lower bounds of edges into u

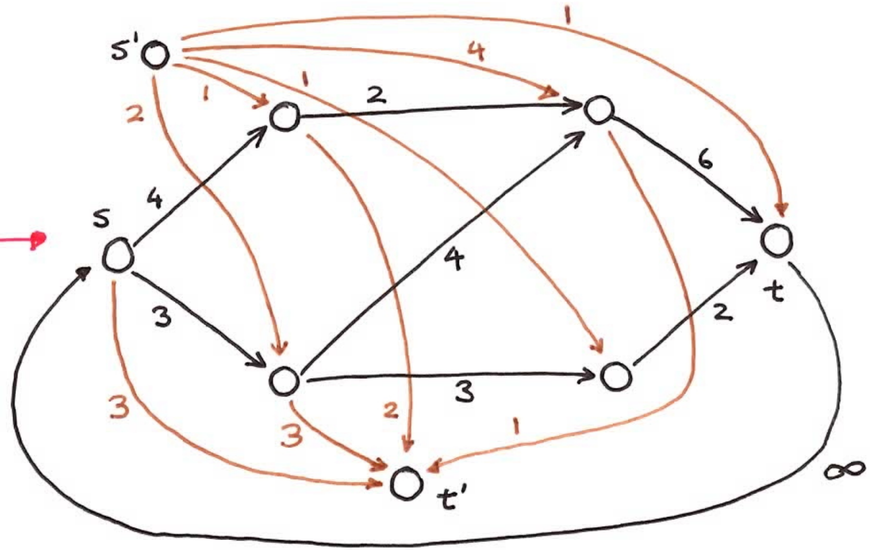
$$\text{out\_low}(u) = \sum_{v \in V} \text{low}(u,v)$$

= sum lower bounds of edges out of u



(low, high)

assume no edges into original source



Claim: network G with lower bounds has a legal solution iff normal network G' has max flow that saturates edges out of s'.

← equivalently, all edges into t' saturated.

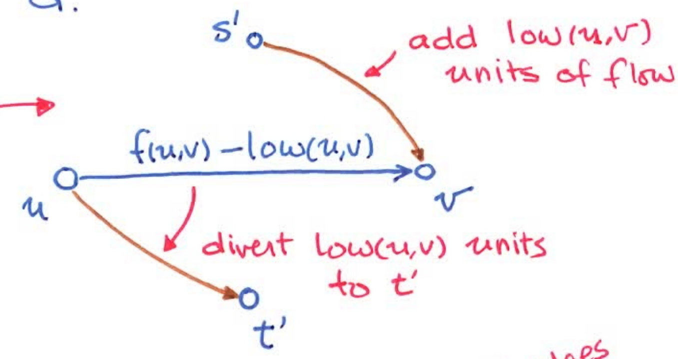
Pf:

($\Rightarrow$) Let f be a legal flow in G.

G: $u \xrightarrow{f(u,v)} v$

$low(u,v) \leq f(u,v) \leq high(u,v)$

flow is conserved!



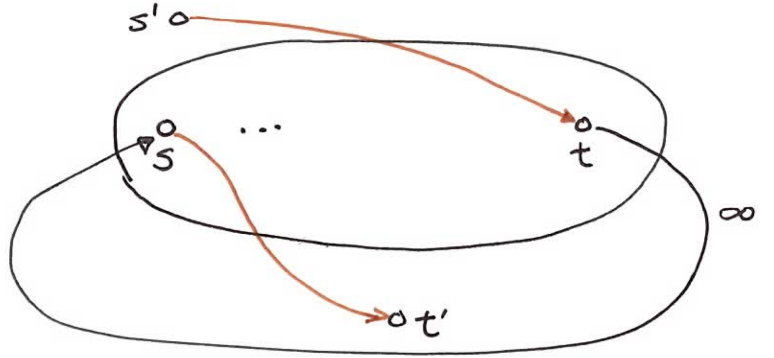
total units added to (s',v) for all edges
$$= \sum_{x \in V} low(x,v) = in\_low(v)$$

total units diverted to (u,t') for all edges
$$= \sum_{x \in V} low(u,x) = out\_low(u)$$

hence all edges out of s' are saturated, and we have max flow

Special cases: s & t

There are no edges into s or out of t



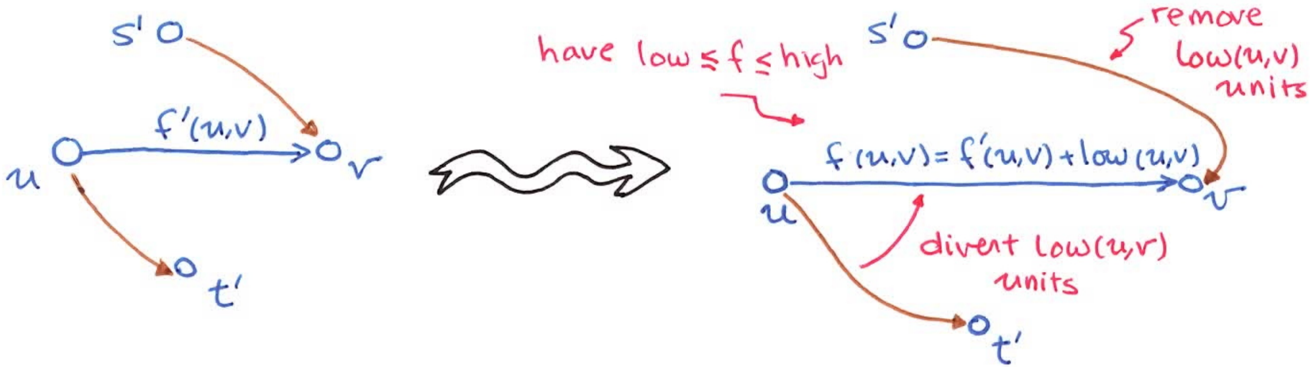
Flow comes into t with nowhere to go.
Flow out of s, but none coming in.
} flow conservation means these amounts are equal

Send this flow from t to s using special ∞-capacity edge.

Pf of Claim continued

($\Leftarrow$) [Need to show if max flow in G' saturates all edges out of s', then G has a legal flow]

Reverse process. Let f' be a max flow in G'



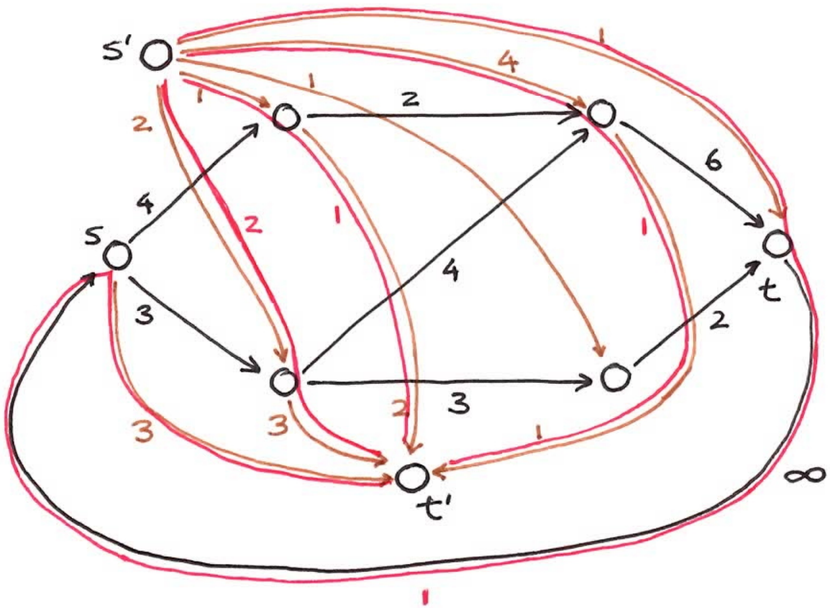
when all edges processed, out_low(u) units diverted from (u,t') and in_low(v) units removed from (s',v).

Flow does not involve s' & t'. Remove edge (t,s).

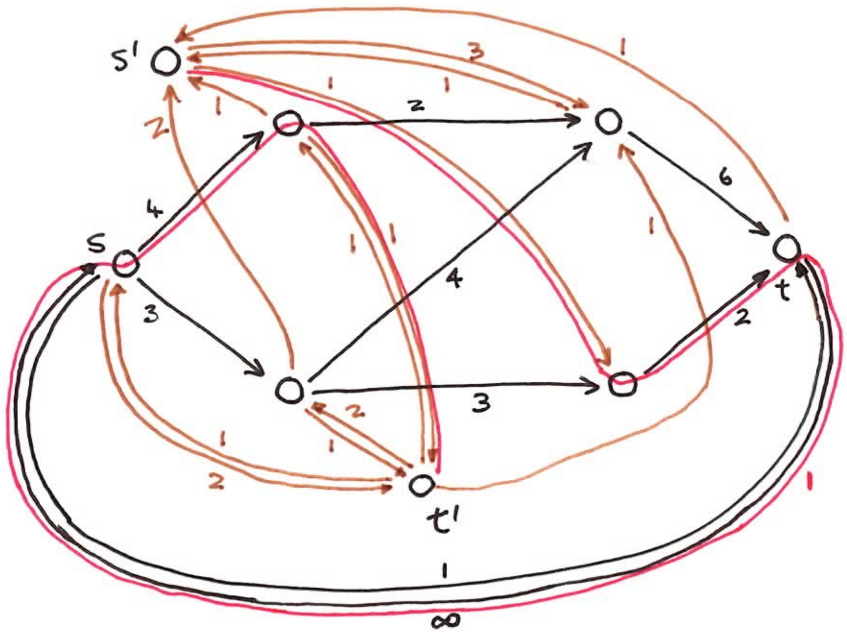
Now we have legal flow in G.

END OF CLAIM ☒

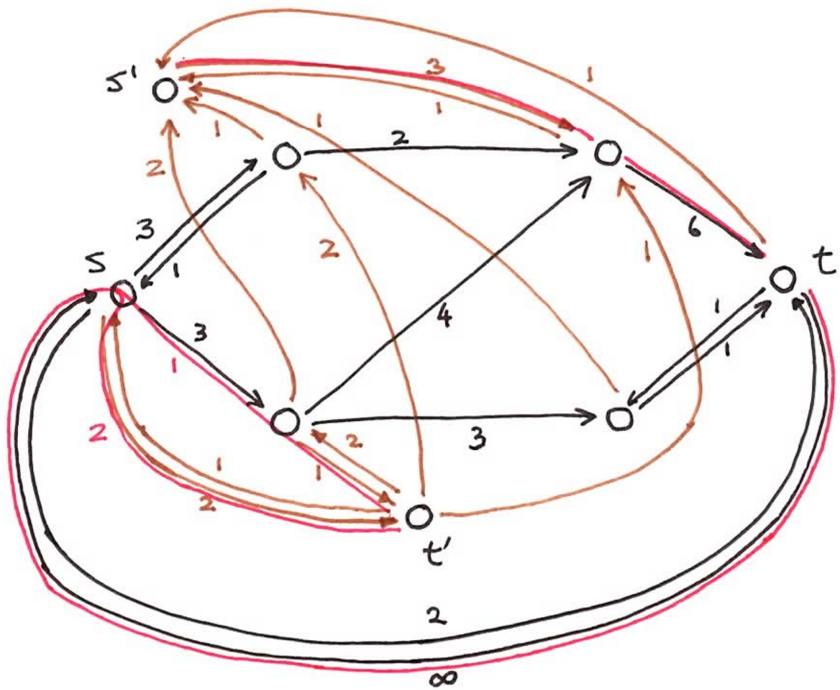
Finding max flow in G'.



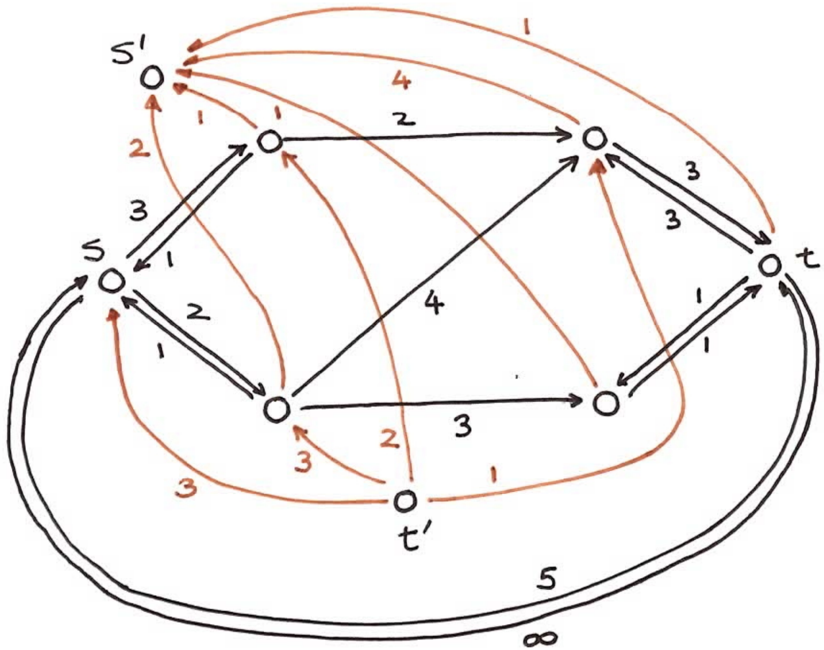
4 augmenting paths with path flows 2, 1, 1, 1



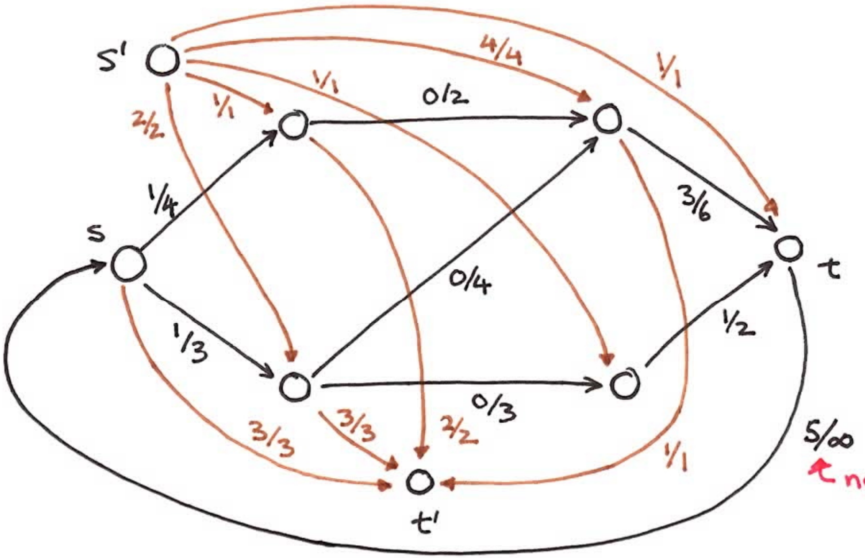
1 augmenting path with path flow 1



2 augmenting paths
with path flows
1, 2



final residual graph has no augmenting paths



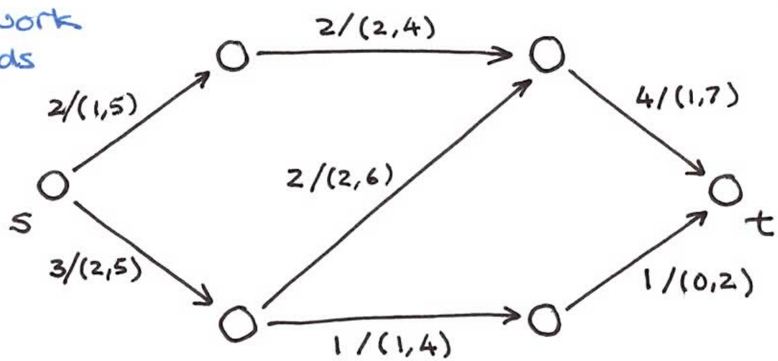
final flow:
edges out of s'
and into t'
are saturated

check flow conserv.

needed to conserve flow

Corresponding flow
in original network
with lower bounds

Add lower bounds
to flow above.



notation:
flow/(low, high)

legal flow,
but not max!

Residual Graph for flow networks w/ lower bounds

$G:$ $u \xrightarrow{5/(2,9)} v$

Can send 4 more units from u to v.
How many can we send back to u from v? 3 [must preserve lower bound]

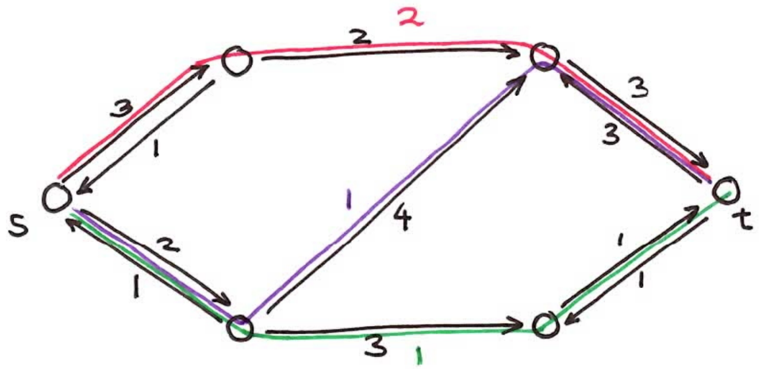
$G_f:$
$$\begin{tikzcd} u \arrow[r, shift left, "4"] & v \arrow[l, shift left, "3"] \end{tikzcd}$$

$$c_f(u,v) = \begin{cases} high(u,v) - f(u,v) & \text{if } (u,v) \in E \\ f(u,v) - low(u,v) & \text{if } (v,u) \in E \\ 0 & \text{otherwise} \end{cases}$$

Augment with paths as before.

Max flow is achieved when no more augmenting paths.

Residual graph respecting lower bounds



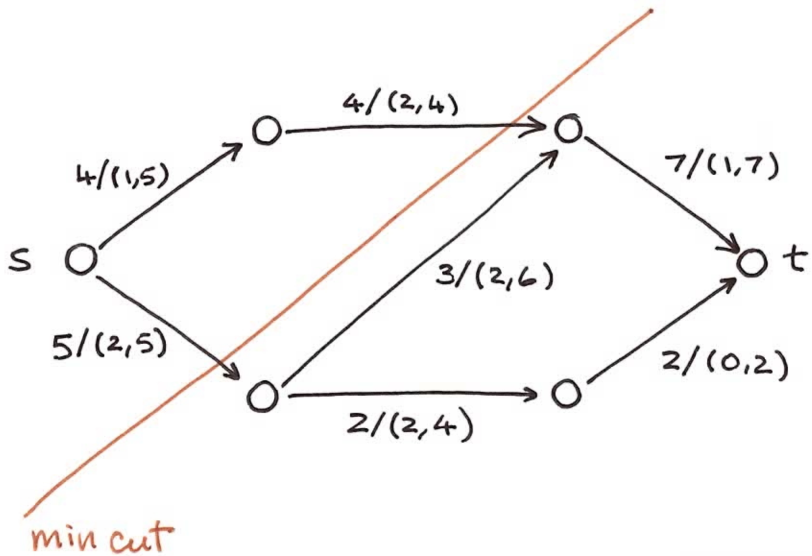
3 augmenting paths

1st path, flow = 2

2^{nd} path, flow = 1

3^{rd} path, flow = 1

Final flow is max.



flow/(low,high)