HW 8 out later today, due next Tue

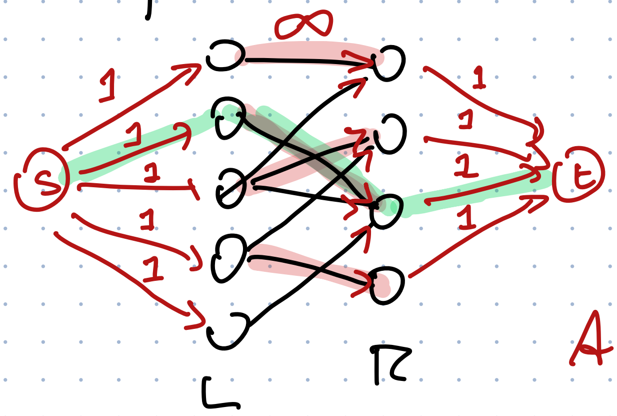
Office hours → basement

VOTE if you can

Encourage others to vote if you can't

Applications of maximum Flow

– Bipartite max matching



Given bipartite $G=(L \sqcup R, E)$

Find max # edges s.t. each vertex touches at most one

Alg: add s, t

add $s \to l_i$ cap 1

$r_j \to t$ cap 1

direct $l_i \to r_j$ cap ∞

Ford-Fulkerson $O(E \cdot |F^*|) = \boxed{O(E \cdot V)}$ → Find max integer flow

return $M = \{ l_i r_j \mid F^*(l_i \to r_j) = 1 \}$

Exam scheduling

We need to schedule exams for

c classes —— $Num[i]$ = # students taking class i

t available times

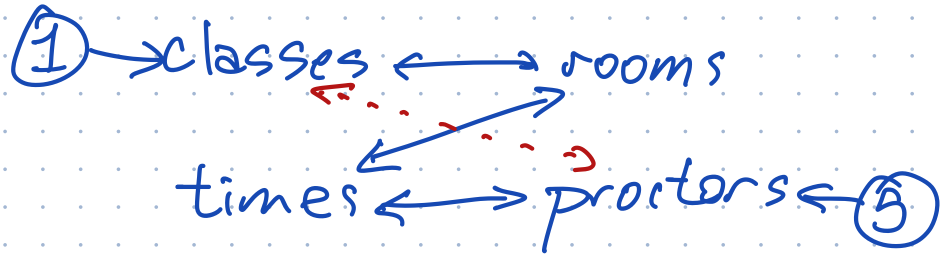
each used ≤ 10 times

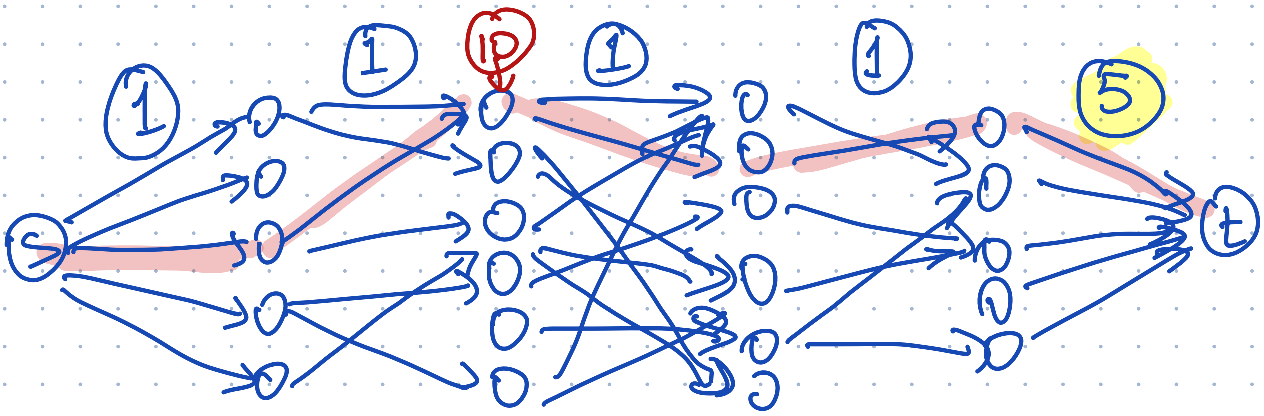
r available rooms —— $Cap[j]$ = # students fit in room j

p available proctors —— $Avail[k, \ell]$ = True iff proctor k is available at time ℓ

Each proctor can be used ≤ 5 times

We need to choose tuples (class, time, room, proctor) for each exam.





classes rooms times proctors

all Num≤Cap all Avail

Compute max integer flow in G

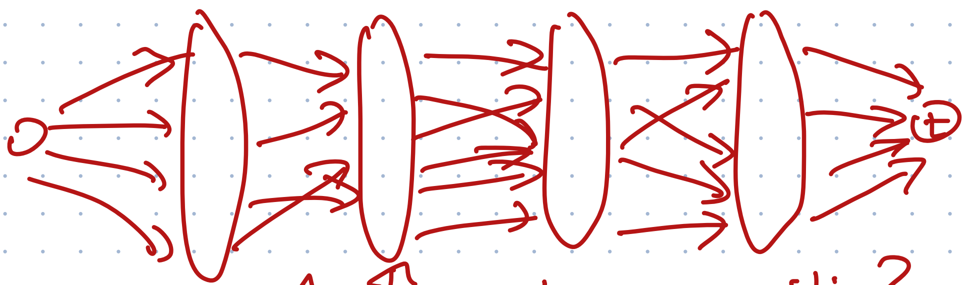
Decompose flow into paths

Each path s → class → room → time → proc → t describes an exam

If $|F^*| < c$ report FAIL

"Tuple selection"

Layers = resources?



vertex capacities?

which edges in each layer?

capacities?

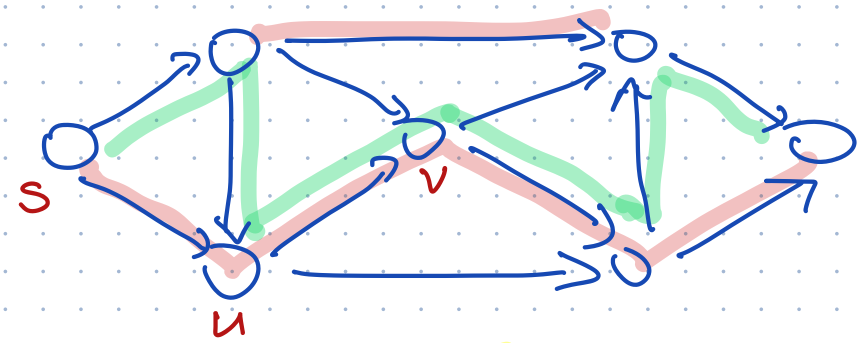
Max Flow → $\boxed{\text{paths = selected tuples}}$!

Disjoint Path Covers

Given a dag $G=(V,E)$

Find a collection of simple paths s.t. each vertex lies on exactly one path

min # paths



$s \to u$
$u \to v$
$\vdots$

$\boxed{\text{Assign}}$ successors to as many vertices as possible

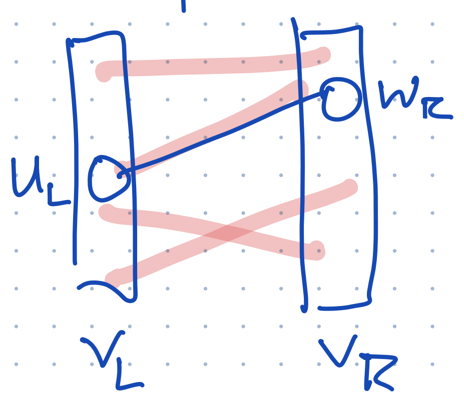
v can be succ. of u
iff $u \to v \in E$
v is not succ of any other node

$\#\text{paths} = \#\text{nodes w/o succ's}$ [last nodes in paths]

$= V - \#\text{nodes with succ's}$

$\min \#\text{paths} = \max \#\text{succs}.$

Reduce to bipartite matching



$u_L w_R \in E'$

$\Longleftrightarrow$

$u \to w \in E$

path cover in G = matching in G' ← Finding max matching

$O(V'E') = \underline{O(VE) \text{ time}}$

Fake profs

Array $C[1..n]$ of classes

$C[i].start$ } times
$C[i].end$
$C[i].loc$

Array $T[1..m, 1..m]$ — transit times

$T[j,k]$ = time to walk from loc j to loc k.

Hire as few profs as possible

Reduce to disjoint path cover

$G = (V, E)$

$V = n$ classes

$E = \{ i \to j \mid C[i].end + T[C[i].loc, C[j].loc] \leq C[j].start \}$

This is a dag!

Path = viable schedule for one prof

Disj Path Cover = overall schedule

\# paths = # profs

$O(VE) = O(n \cdot n^2) = \boxed{O(n^3) \text{ time}}$