Edge-Disjoint Paths

Given directed graph $G=(V,E)$ & two vertices s & t.

Want a max size set of (s,t)-paths that don't share any edges.

Alg:

Give each edge a capacity of 1.

Compute a max (s,t)-flow.

$\mathcal{P}$: any set of k edge-disjoint paths

Claim: There exists a flow f s.t. $|f| = k$.

$f(e) = 1$ if e is in same path of $\mathcal{P}$

$f(e) = 0$ o.w.

f^*: max flow

capacities are integers, so we can assume f^* is integral

Claim: There is a set of $|f^*|$ edge-disjoint paths.

There is a path-cycle decomposition of f^*.
Each path has weight/value 1.

So there are $|f^*|$ of them.

& they are edge-disjoint.
o.w, we'd have flow ≥ 2
on an edge

So if k^* is max # paths
$|f^*| \geq k^*$ & we can grab
k^* paths from it, so
$k^* \geq |f^*|$
$\Rightarrow k^* = |f^*|$

Time: Orlin: $O(VE) + O(k^* E)$
If G is simple, $|f^*| \leq |V| - 1$

So just use Ford-Fulkerso

in $O(E|f^*|) = O(VE)$

$$\|$$

$O(k^* E)$

Bipartite Matching

Given an undirected bipartite graph $G=(L \cup R, E)$ (so no L-L or R-R edges)

A <u>matching</u> $M \subseteq E$ has no two edges of M sharing an endpoint.
(i.e. a subgraph where all

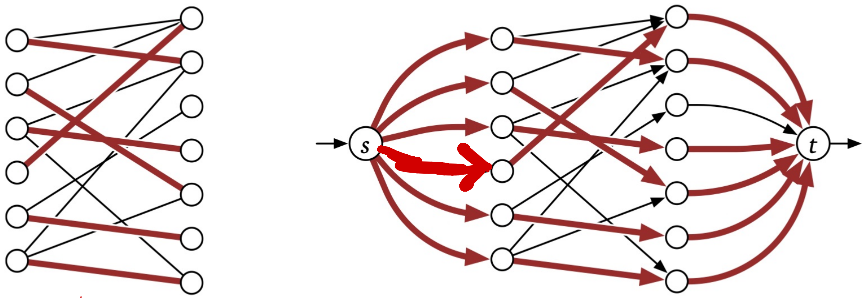
vertices have degree 1)

Maybe L is med studentsd
R is internships.

Goal: Find a matching
of max size.

Alg: Create G' from G by:
1) orient edges from L to R
2) add vertices s & t
3) add edge $s \to v$ for all $v \in L$
4) add edge $u \to t$ for all $u \in R$

5) Assign capacity 1 to all edges.



M

Claim: For any matching M in G, there is a feasible (s,t)-flow f_M in G' s.t.

$$|f_M| = |M|$$

For each edge $uv \in M$, set $f(s \to u) = f(u \to v) = f(v \to t) := 1$

Any edges we missed get flow 0.

f^*: max flow - assumed integral

M^*: max matching

we know $|f^*| \geq |M^*|$

Claim: There is a matching M' s.t. $|M'| = |f^*|$.

Take all $uv \in G$ s.t. $f^*(u \to v) = 1$.

Time with FF: $O(|f^*|E) = O(VE)$

Could change max # times
we use the edges or vertices
by changing capacities.

Assignment Problems

Project Selection

Given a directed acyclic graph $G=(V,E)$.

V: the projects

edge $u \to v$ whenever u depends upon v

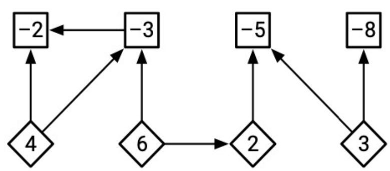
profits $\$\ V \to \mathbb{R}$

$\$(v)$ profit if we complete v

negative profit $\equiv$ cost

Goal: Select $X \subseteq V$ that includes dependencies for all $x \in X$.

Want to max $\sum_{x \in X} p(x)$.



Think of finding a cut (S, T) of Selected & Turned down projects.

Turn G into a flow network G'.

1) Add source s & target t.
2) For each v s.t. $\$(v) > 0$, add $s \to v$.

   For each u s.t. $\$(u) < 0$, add $u \to t$.

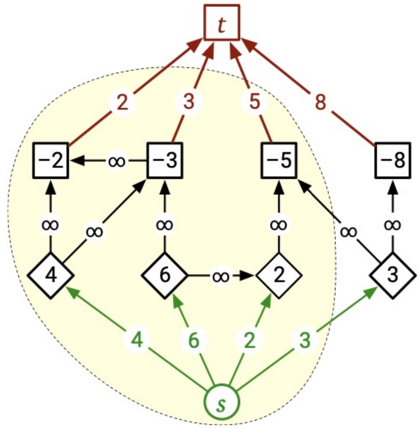
3) $c(s \to v) := \beta(v) \quad \forall\ \beta(v) > 0$

$c(u \to t) := -\beta(u) \quad \forall\ \beta(u) < 0$

$c(u \to v) := \infty \quad \forall\ u \to v \in G$

Let (S, T) be any finite capacity (s,t)-cut, we won't have $u \in S$, $v \in T$ if $u \to v \in G$, so

$X := S - s$ is a feasible solution to our problem.



$$P := \sum_{v} \max\{0, \$(v)\} = \sum_{\$(v)>0} \$(v)$$

For any $X \subseteq V$, let

$$\text{cost}(X) := \sum_{\substack{u \in X \\ \$(u)<0}} -\$(u) = \sum_{u \in X} c(u \to t)$$

$$\text{yield}(X) := \sum_{\substack{v \in X \\ \$(v)>0}} \$(v) = \sum_{v \in X} c(s \to v)$$

$$\text{profit}(X) := \sum_{v \in X} \$(v) = \text{yield}(X) - \text{cost}(X)$$

By def., $P = \text{yeild}(V)$

$$= \text{yield}(S) + \text{yield}(T)$$

Assuming $\|S, T\|$ is finite

$$\|S, T\| = \text{cost}(S) + \text{yield}(T)$$

$$\begin{aligned}
\text{profit}(S) &= \text{yield}(S) - \text{cost}(S) \\
&= P - \text{yield}(T) - \text{cost}(S) \\
&= P - \|S, T\|
\end{aligned}$$

So compute min cut to maximize profit.

Time: Orlin in $O(VE)$