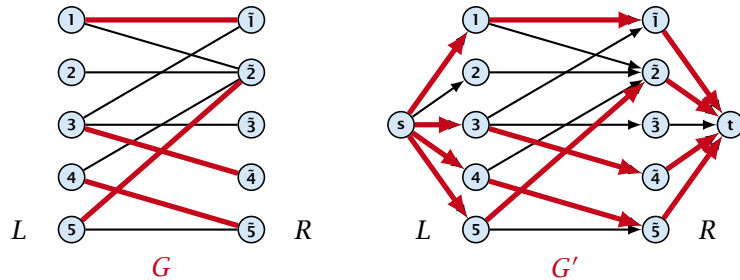


Proof

Max cardinality matching in $G \leq$ value of maxflow in G'

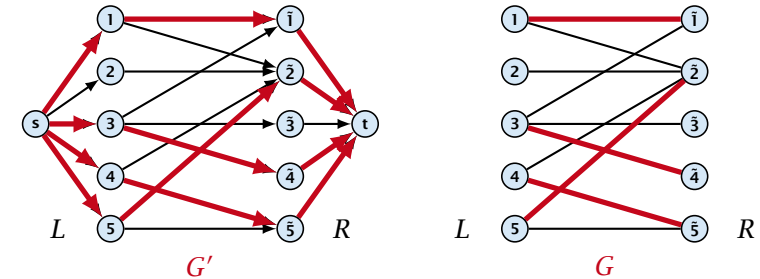
- ▶ Given a maximum matching M of cardinality k .
- ▶ Consider flow f that sends one unit along each of k paths.
- ▶ f is a flow and has cardinality k .



Proof

Max cardinality matching in $G \geq$ value of maxflow in G'

- ▶ Let f be a maxflow in G' of value k
- ▶ Integrality theorem $\Rightarrow k$ integral; we can assume f is 0/1.
- ▶ Consider $M =$ set of edges from L to R with $f(e) = 1$.
- ▶ Each node in L and R participates in at most one edge in M .
- ▶ $|M| = k$, as the flow must use at least k middle edges.



13.1 Matching

Which flow algorithm to use?

- ▶ Generic augmenting path: $\mathcal{O}(m \text{val}(f^*)) = \mathcal{O}(mn)$.
- ▶ Capacity scaling: $\mathcal{O}(m^2 \log C) = \mathcal{O}(m^2)$.

Baseball Elimination

team i	wins w_i	losses ℓ_i	remaining games			
			Atl	Phi	NY	Mon
Atlanta	83	71	–	1	6	1
Philadelphia	80	79	1	–	0	2
New York	78	78	6	0	–	0
Montreal	77	82	1	2	0	–

Which team can end the season with most wins?

- ▶ Montreal is eliminated, since even after winning all remaining games there are only 80 wins.
- ▶ But also Philadelphia is eliminated. Why?

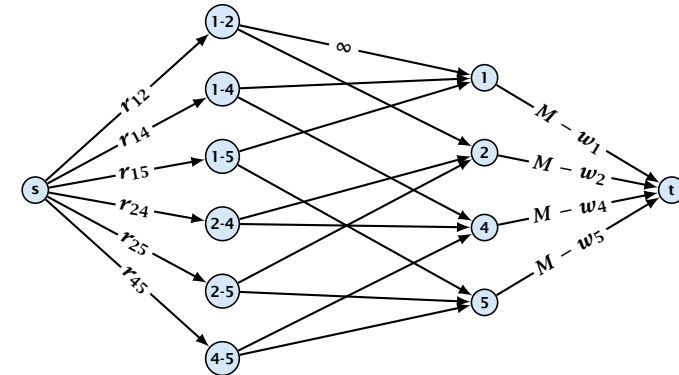
Baseball Elimination

Formal definition of the problem:

- ▶ Given a set S of teams, and one specific team $z \in S$.
- ▶ Team x has already won w_x games.
- ▶ Team x still has to play team y , r_{xy} times.
- ▶ Does team z still have a chance to finish with the most number of wins.

Baseball Elimination

Flow network for $z = 3$. M is number of wins Team 3 can still obtain.



Idea. Distribute the results of remaining games in such a way that no team gets too many wins.

Certificate of Elimination

Let $T \subseteq S$ be a subset of teams. Define

$$w(T) := \sum_{i \in T} w_i, \quad r(T) := \sum_{i, j \in T, i < j} r_{ij}$$

wins of teams in T
remaining games among teams in T

If $\frac{w(T) + r(T)}{|T|} > M$ then one of the teams in T will have more than M wins in the end. A team that can win at most M games is therefore eliminated.

Theorem 1

A team z is eliminated if and only if the flow network for z does not allow a flow of value $\sum_{i, j \in S \setminus \{z\}, i < j} r_{ij}$.

Proof (\Leftarrow)

- ▶ Consider the mincut A in the flow network. Let T be the set of team-nodes in A .
- ▶ If for a node $x-y$ not both team-nodes x and y are in T , then $x-y \notin A$ as otherwise the cut would cut an infinite capacity edge.
- ▶ We don't find a flow that saturates all source edges:

$$\begin{aligned} r(S \setminus \{z\}) &> \text{cap}(A, V \setminus A) \\ &\geq \sum_{i < j: i \notin T \vee j \notin T} r_{ij} + \sum_{i \in T} (M - w_i) \\ &\geq r(S \setminus \{z\}) - r(T) + |T|M - w(T) \end{aligned}$$

- ▶ This gives $M < (w(T) + r(T))/|T|$, i.e., z is eliminated.

Baseball Elimination

Proof (\Rightarrow)

- ▶ Suppose we have a flow that saturates all source edges.
- ▶ We can assume that this flow is **integral**.
- ▶ For every pairing x - y it defines how many games team x and team y should win.
- ▶ The flow leaving the team-node x can be interpreted as the additional number of wins that team x will obtain.
- ▶ This is less than $M - w_x$ because of capacity constraints.
- ▶ Hence, we found a set of results for the remaining games, such that no team obtains more than M wins in total.
- ▶ Hence, team z is not eliminated.

Project Selection

Project selection problem:

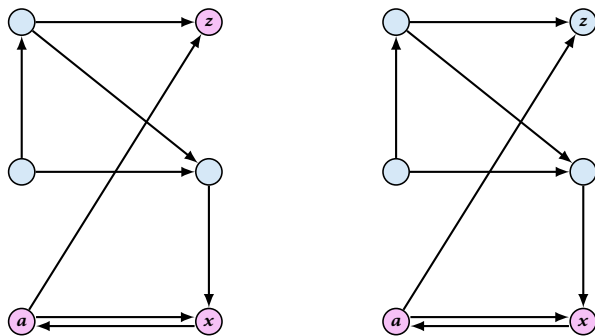
- ▶ Set P of possible projects. Project v has an associated profit p_v (can be positive or negative).
- ▶ Some projects have requirements (taking course EA2 requires course EA1).
- ▶ Dependencies are modelled in a graph. Edge (u, v) means “can’t do project u without also doing project v .”
- ▶ A subset A of projects is **feasible** if the prerequisites of every project in A also belong to A .

Goal: Find a feasible set of projects that maximizes the profit.

Project Selection

The prerequisite graph:

- ▶ $\{x, a, z\}$ is a feasible subset.
- ▶ $\{x, a\}$ is infeasible.



Project Selection

Mincut formulation:

- ▶ Edges in the prerequisite graph get infinite capacity.
- ▶ Add edge (s, v) with capacity p_v for nodes v with positive profit.
- ▶ Create edge (v, t) with capacity $-p_v$ for nodes v with negative profit.

