Advanced Algorithmic Techniques (COMP523)

Network Flows 3

Recap and plan

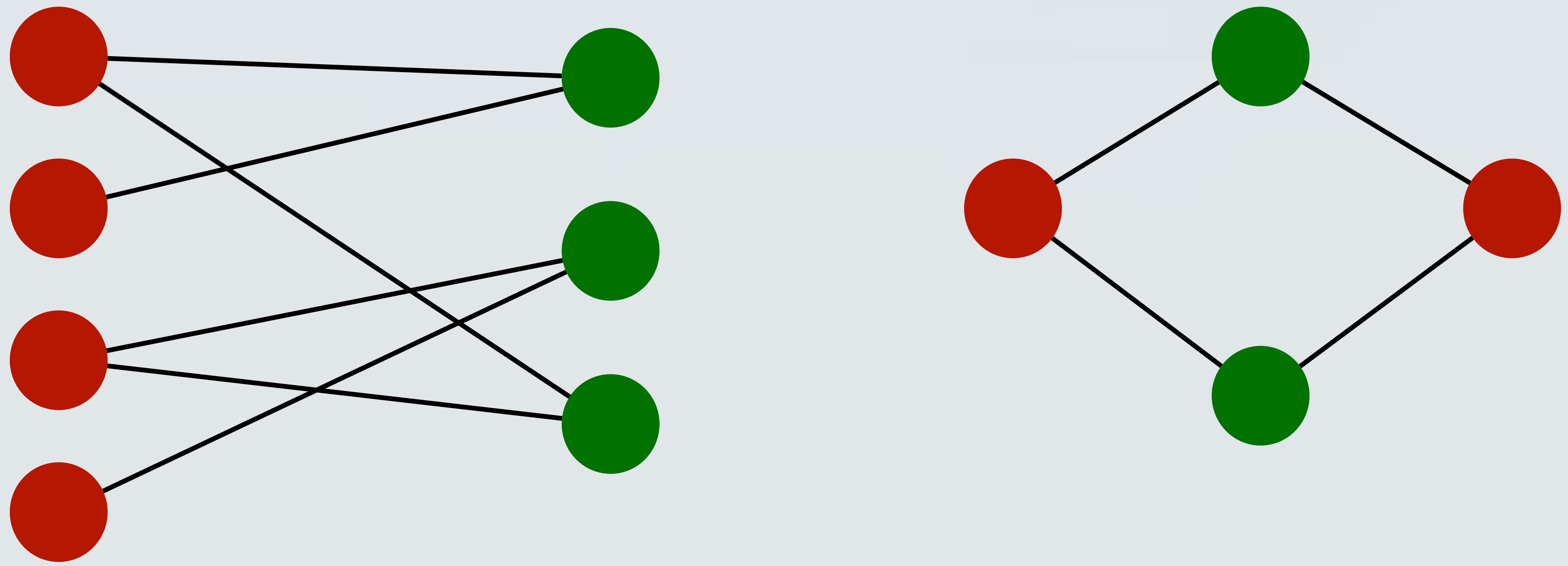
- **Last 2 lectures:**
  - Maximum Flow.
  - The Ford-Fulkerson Algorithm.
  - The Max-Flow - Min - Cut theorem.
  - The Edmonds-Karp algorithm.
- **This lecture:**
  - Modelling with flows.
  - Maximum Bipartite Matching.
  - Baseball Elimination.
  - Open-pit mining.

Bipartite Matching

- *Maximum Bipartite Matching* or Maximum matching on a bipartite graph G.

Bipartite graphs

- A graph G=(V,E) is bipartite *if any only if* it can be partitioned into sets A and B such that each edge has one endpoint in A and one endpoint in B.
  - Often, we write G=(A U B,E).

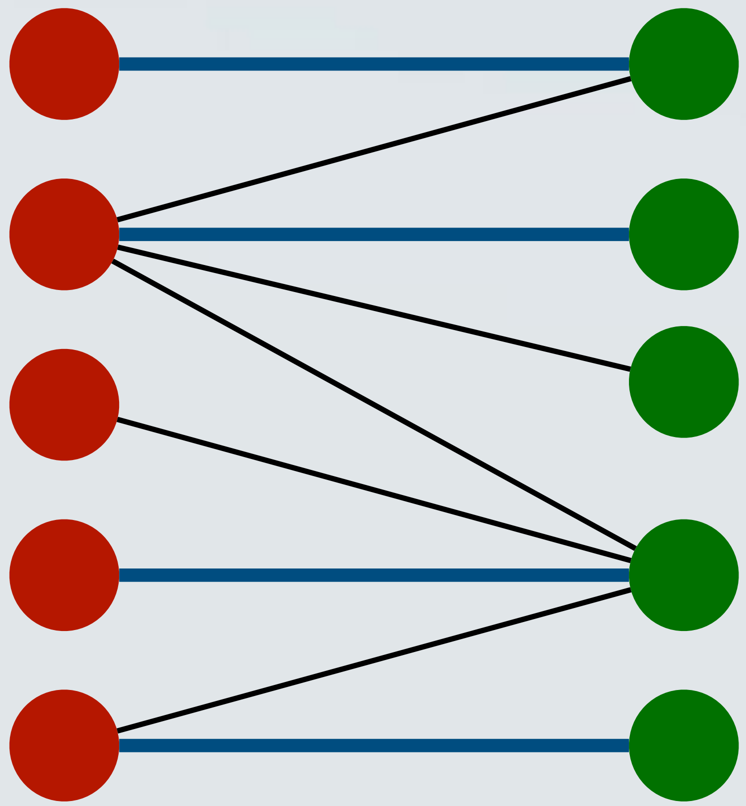


Bipartite Matching

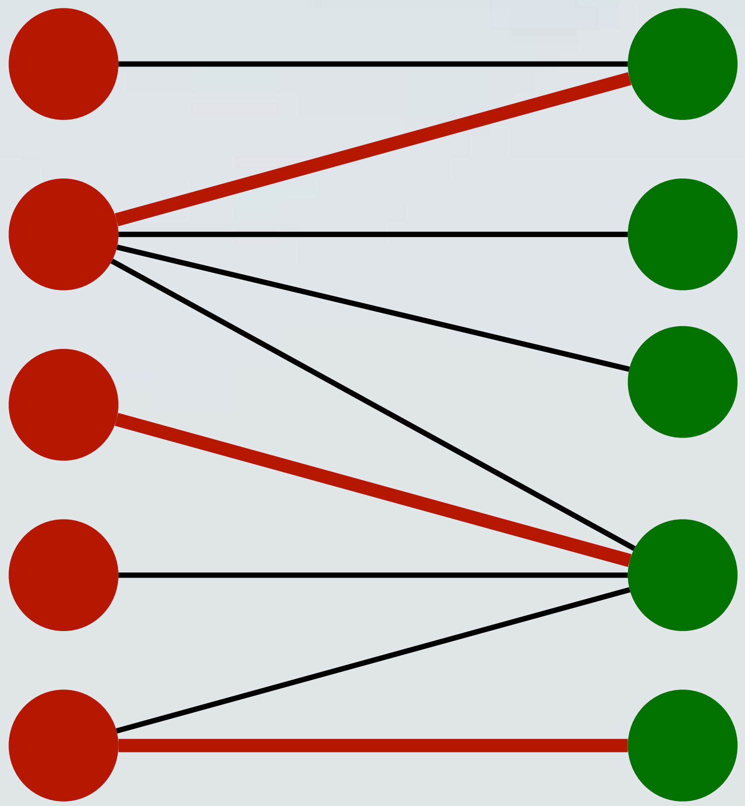
- *Maximum Bipartite Matching* or Maximum matching on a bipartite graph G.
  - Matching: A subset M of the edges E such that each node v of V appears in at most one edge e in E.
  - Maximum matching: A matching with maximum cardinality.(i.e., |M| is maximised).

Example

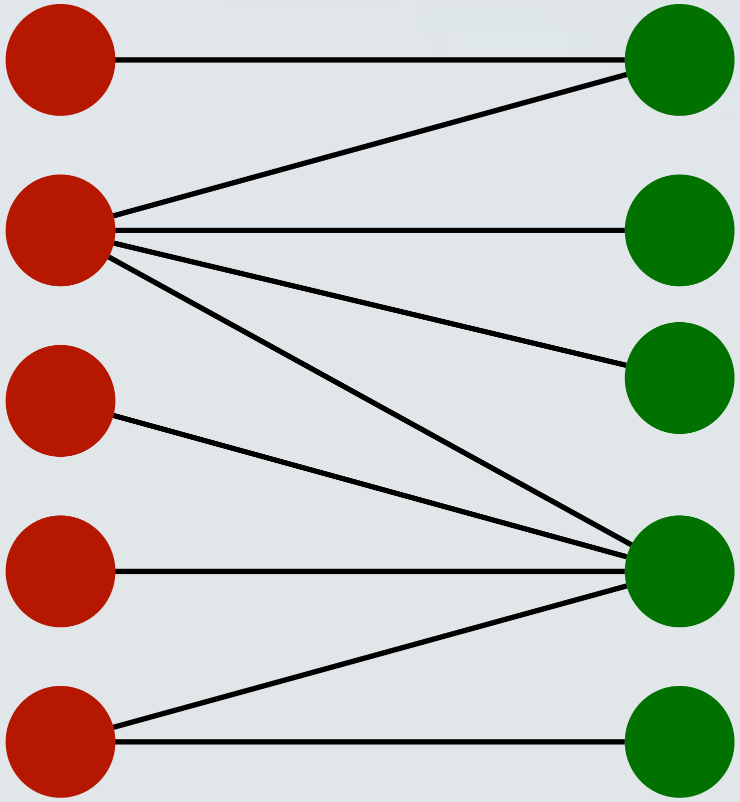
A maximum matching



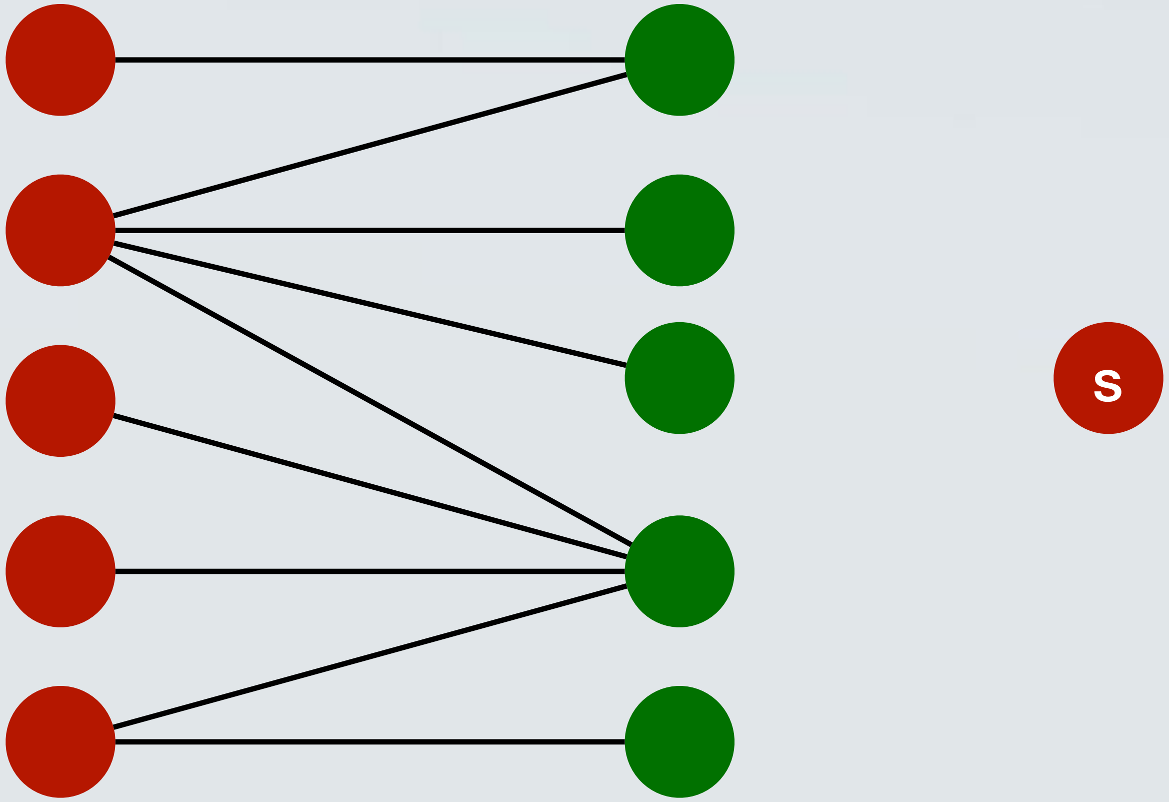
A maximal matching



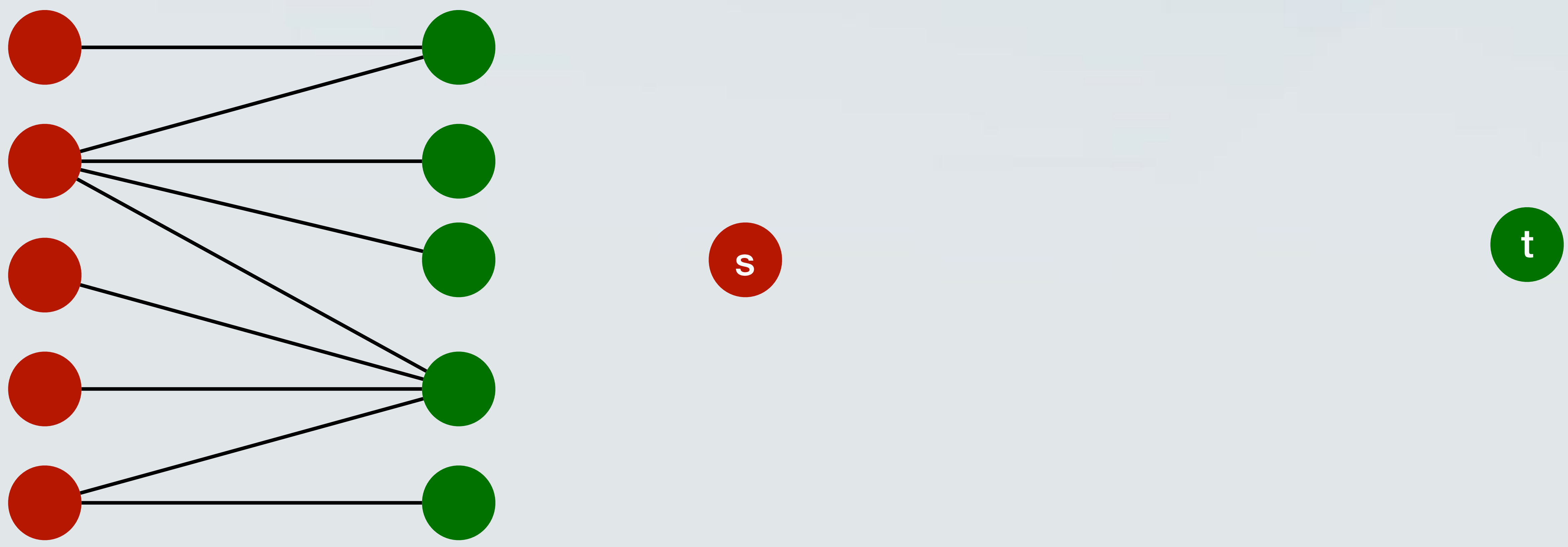
From matchings to flows



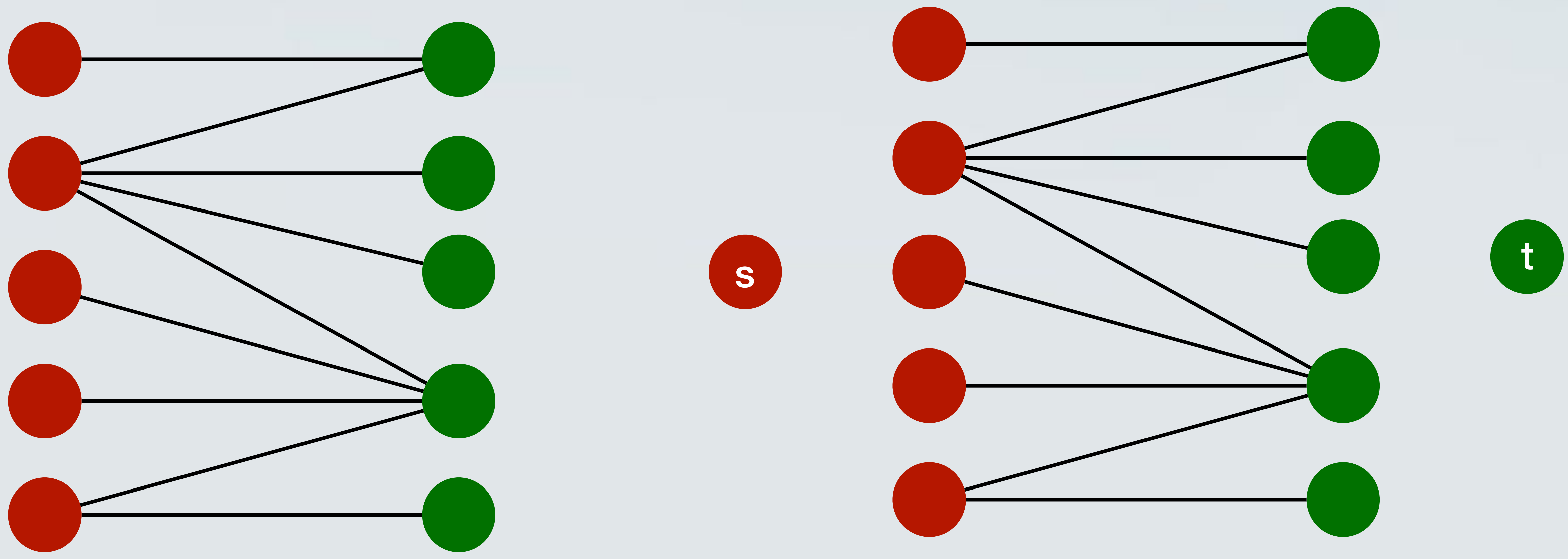
From matchings to flows



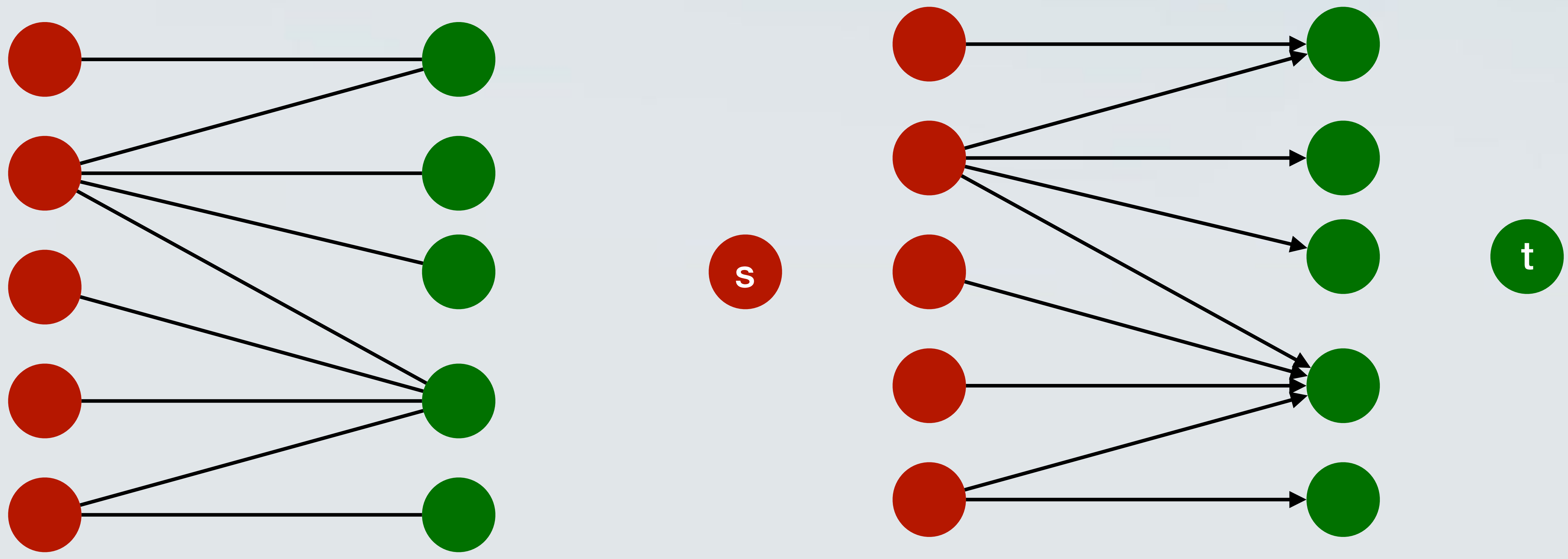
From matchings to flows



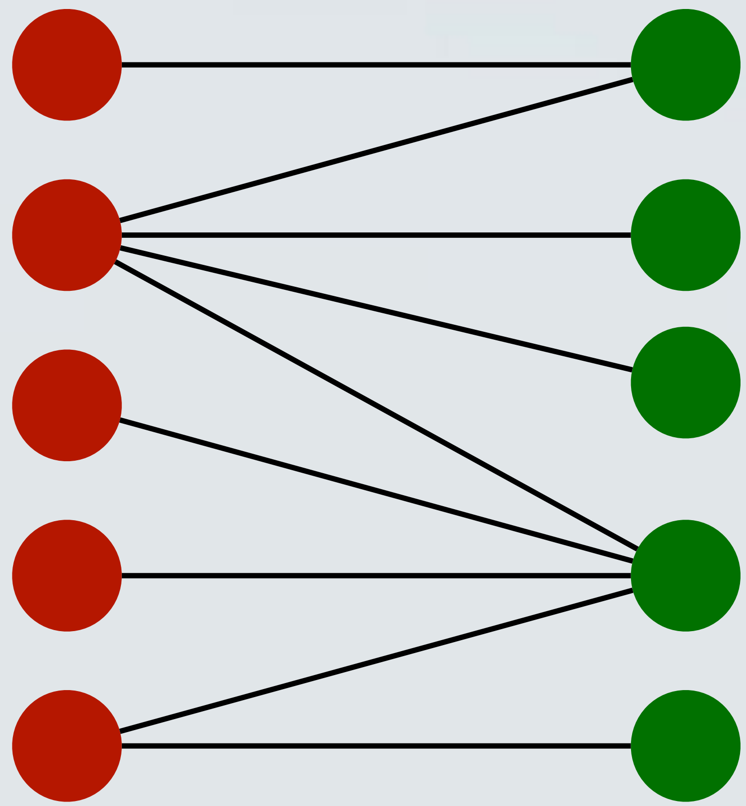
From matchings to flows

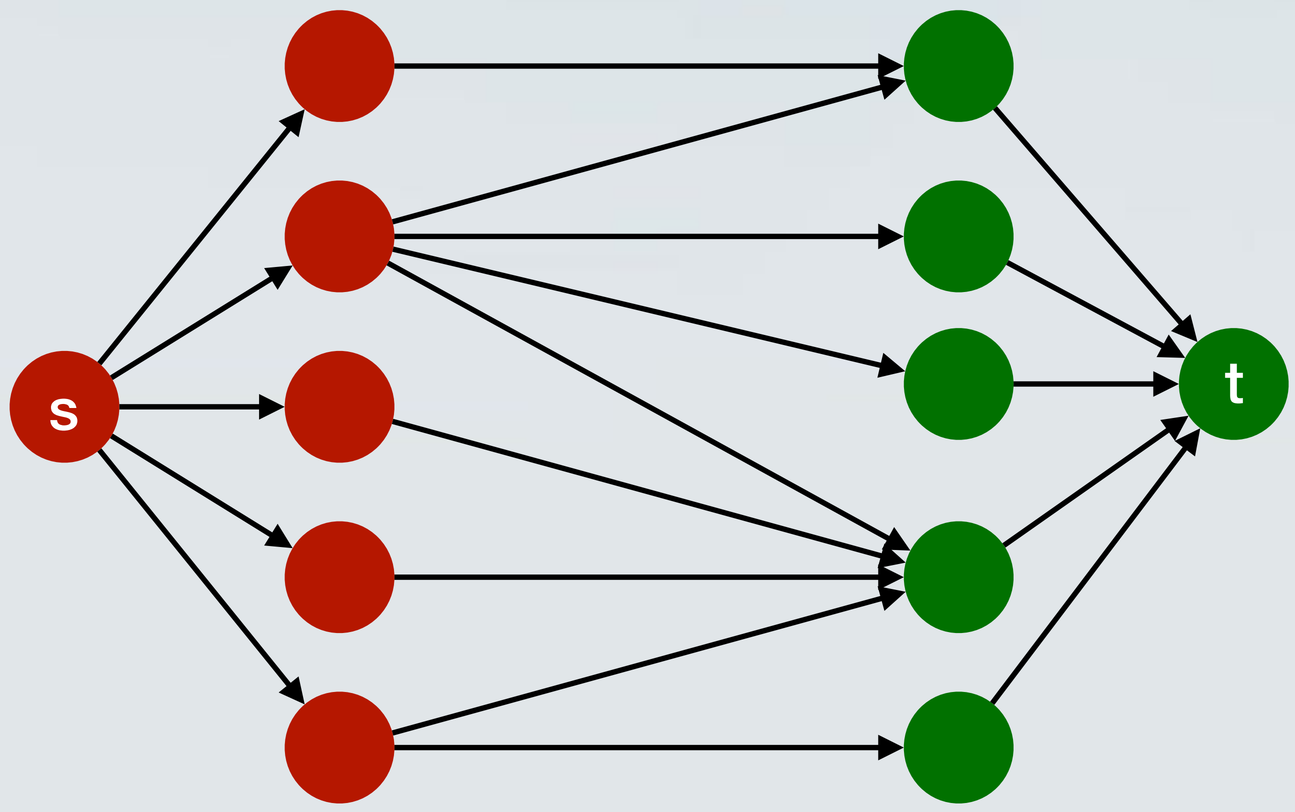


From matchings to flows

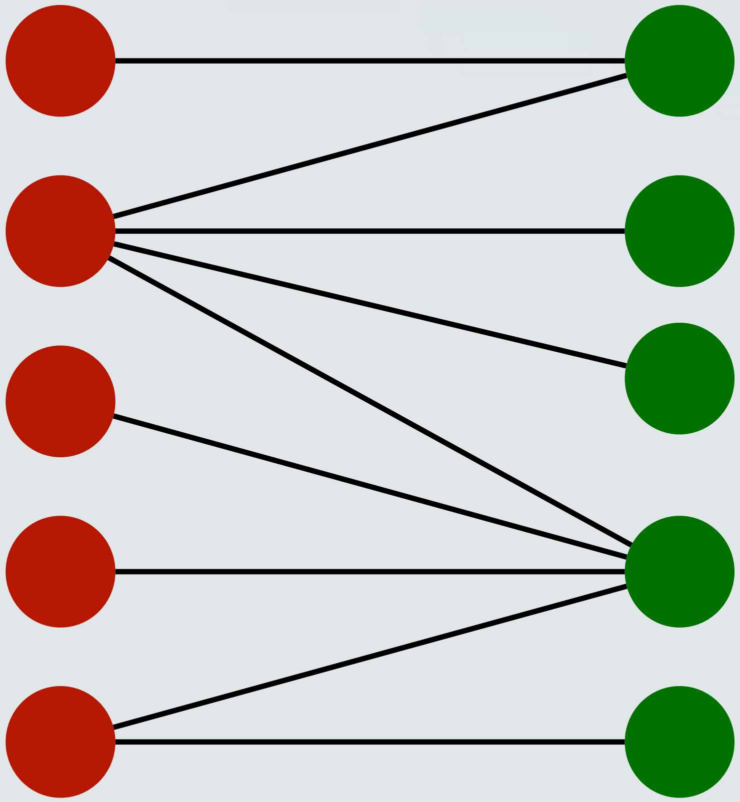


From matchings to flows

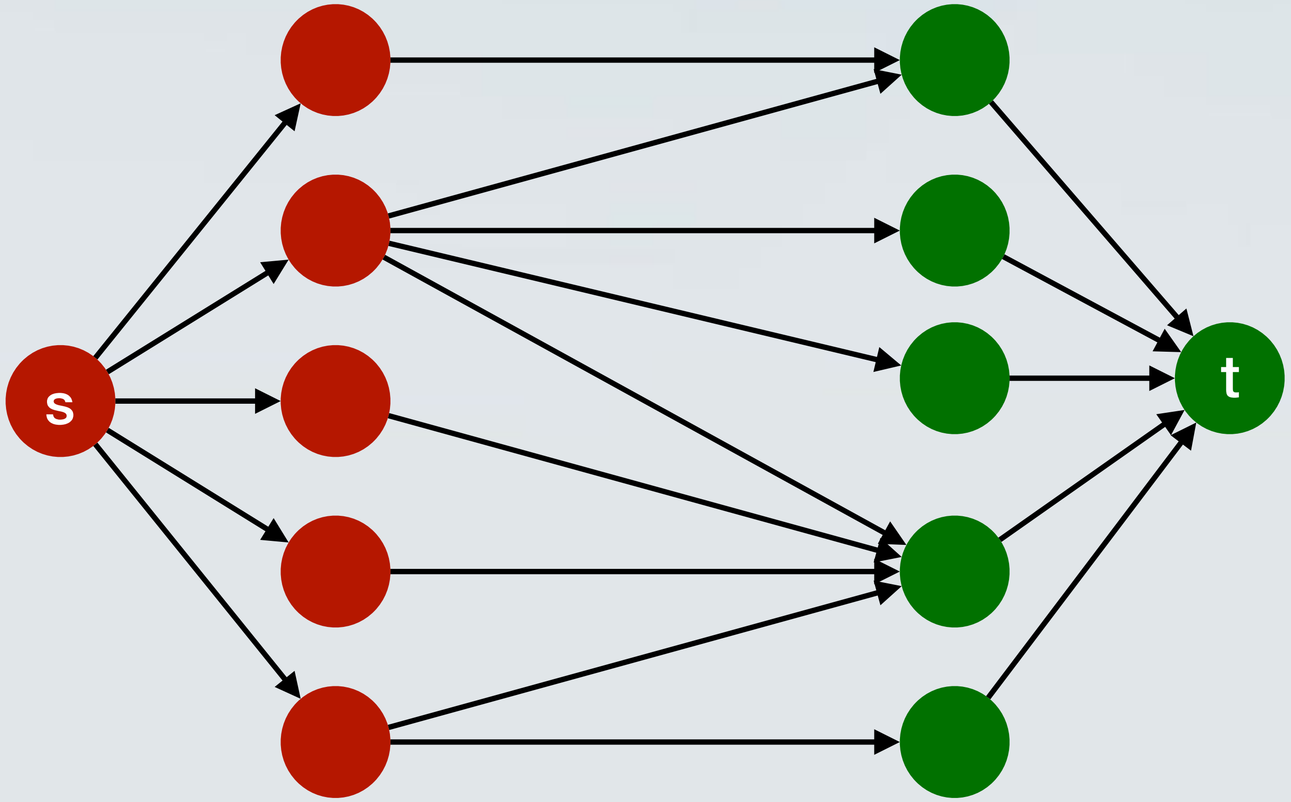




From matchings to flows



All capacities are set to 1.



From matchings to flows

- Claim: Assume that there is a matching M of size k on G. Then there is a flow f of value k in G^f.

From matchings to flows

- Claim: Assume that there is a matching M of size k on G. Then there is a flow f of value k in G^f.
  - Consider the matching

    $M = \{(u_1, v_1), (u_2, v_2), \ldots , (u_k, v_k)\}$

From matchings to flows

- Claim: Assume that there is a matching M of size k on G. Then there is a flow f of value k in G^f.

  - Consider the matching

    $M = \{(u_1, v_1), (u_2, v_2), \dots , (u_k, v_k)\}$

  - Consider the flow such that

    $f(s, u_i) = f(u_i, v_i) = f(v_i, t) = 1$ for all $i = 1, \dots , k$

    $f(e) = 0$, otherwise

From matchings to flows

- Claim: Assume that there is a matching M of size k on G. Then there is a flow f of value k in G^f.
  - Consider the matching

    $M = \{(u_1, v_1), (u_2, v_2), \ldots , (u_k, v_k)\}$

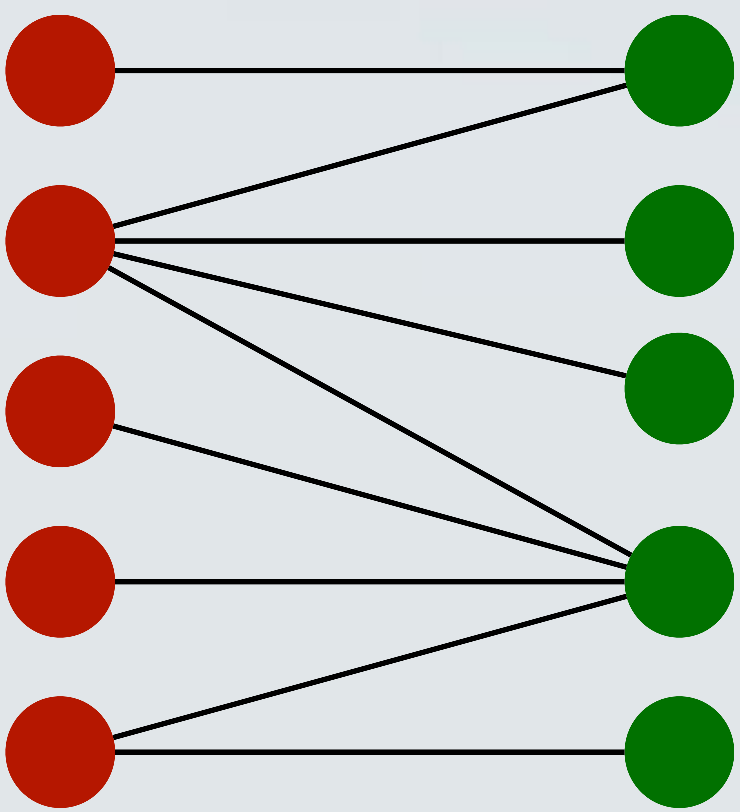
  - Consider the flow such that

    $f(s, u_i) = f(u_i, v_i) = f(v_i, t) = 1$ for all $i = 1, \ldots , k$

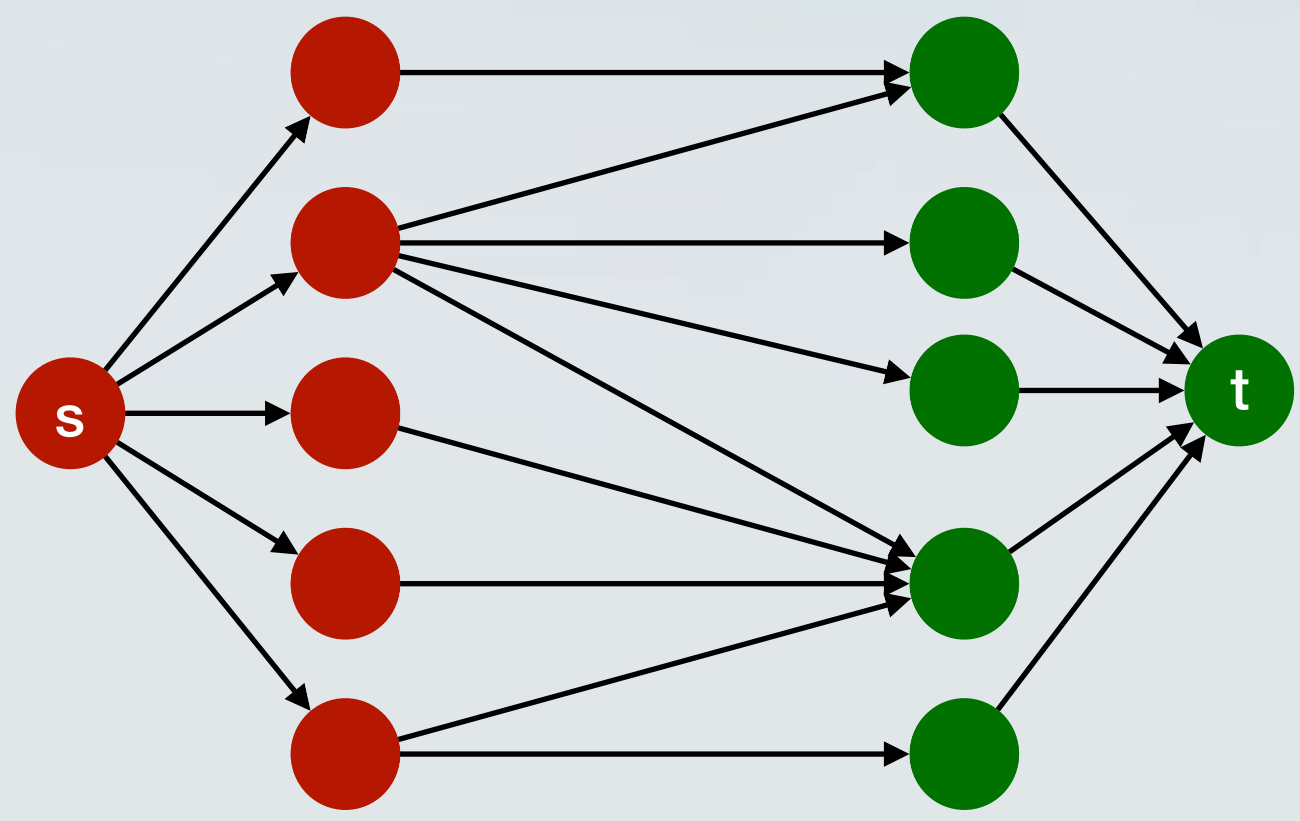
    $f(e) = 0$, otherwise

  - This is a feasible flow and obviously has value k.

From matchings to flows

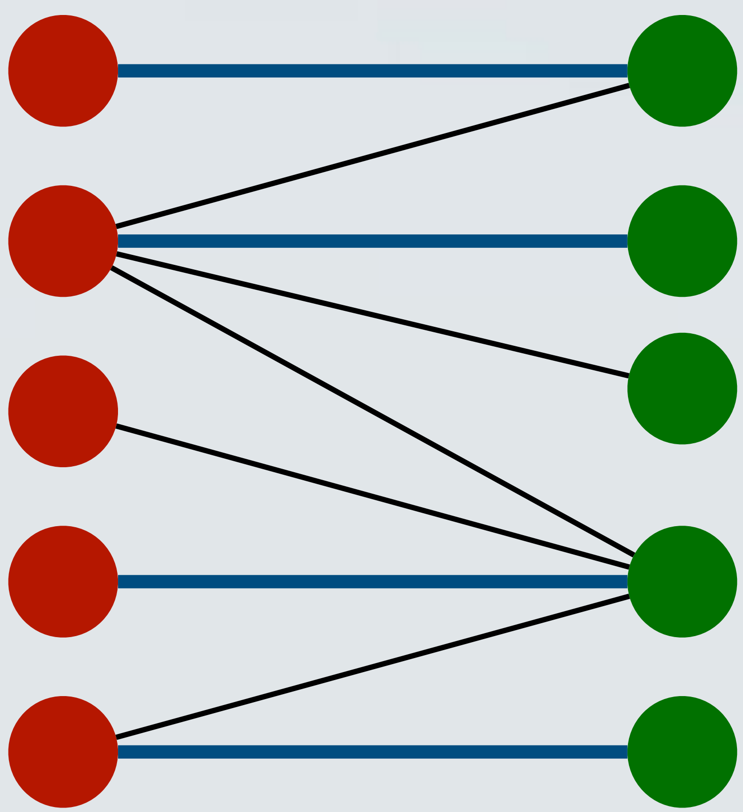


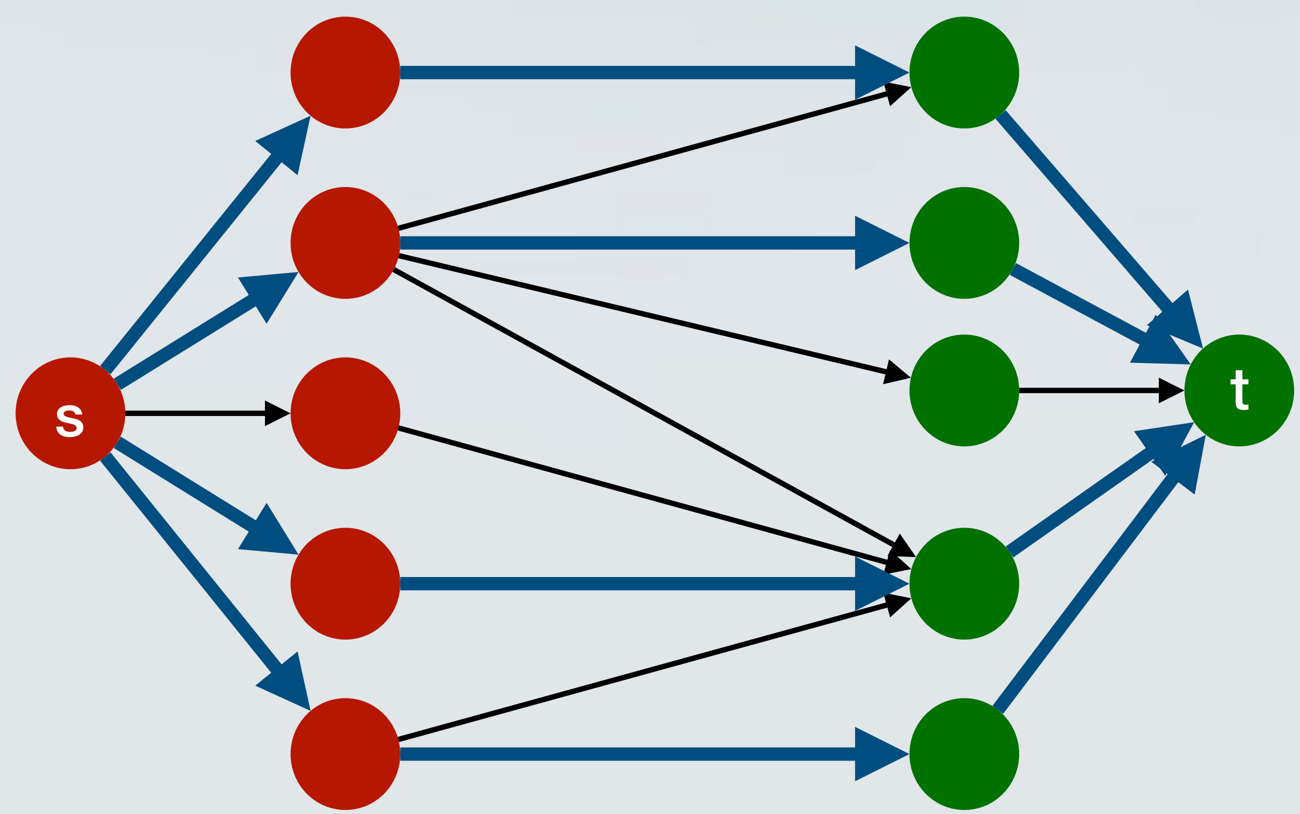
All capacities are set to 1.



From matchings to flows

All capacities are set to 1.





From flows to matchings

- Claim: Assume that there is a a flow f of value k in G^f. Then there is a matching M of size k on G.

From flows to matchings

- Claim: Assume that there is a a flow f of value k in G^f. Then there is a matching M of size k on G.
  - For an edge e, $f(e)$ is either *0* or *1*. (why?)

From flows to matchings

- Claim: Assume that there is a a flow f of value k in G^f. Then there is a matching M of size k on G.
  - For an edge e, $f(e)$ is either *0* or *1*. (why?)
  - Consider the set M' of edges with $f(e) = 1$.

From flows to matchings

- Consider the set M’ of edges with $f(e) = 1$.

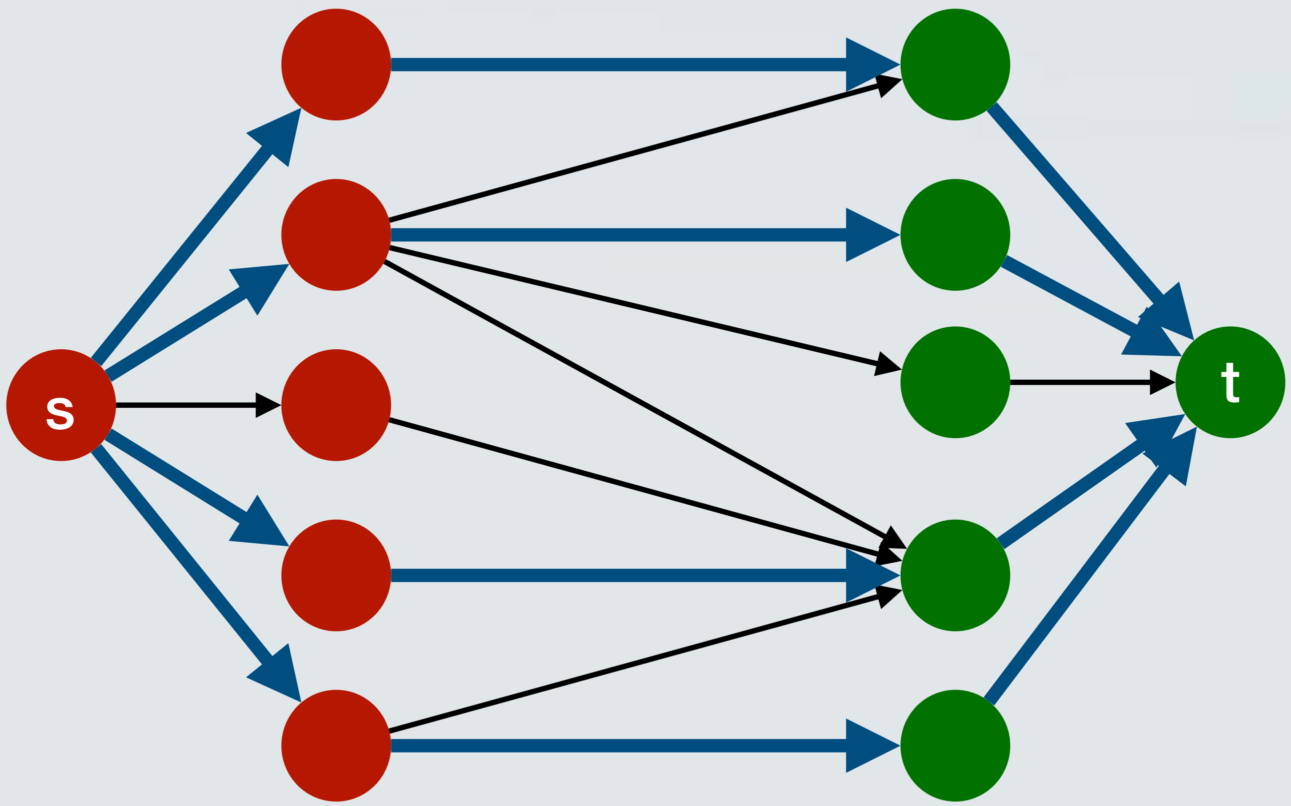
From flows to matchings

- Consider the set M’ of edges with f(e) = *1*.
  - Claim: |M’| = k.

From flows to matchings

- Consider the set M' of edges with $f(e) = 1$.
  - Claim: $|M'| = k$.

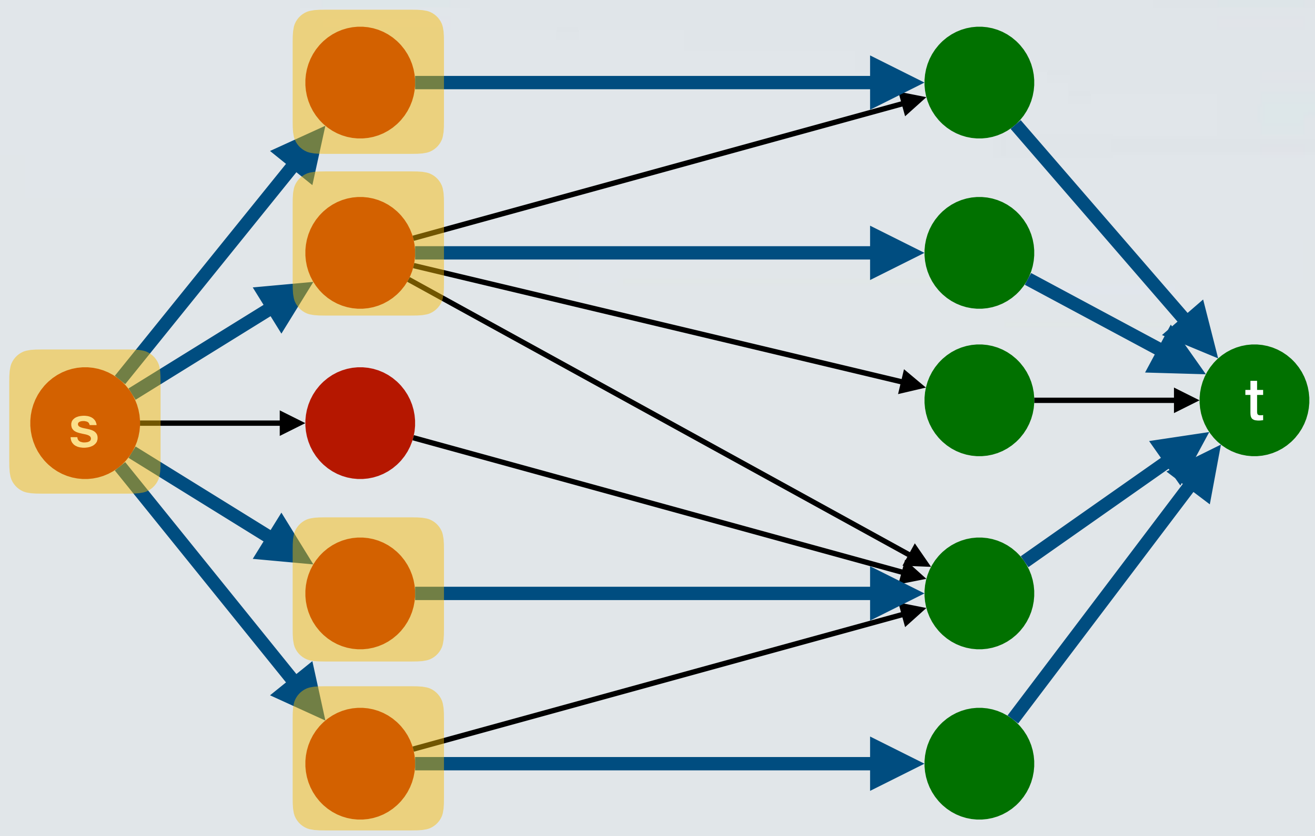
All capacities are set to 1.



From flows to matchings

- Consider the set M’ of edges with f(e) = *1*.
  - Claim: |M’| = k.

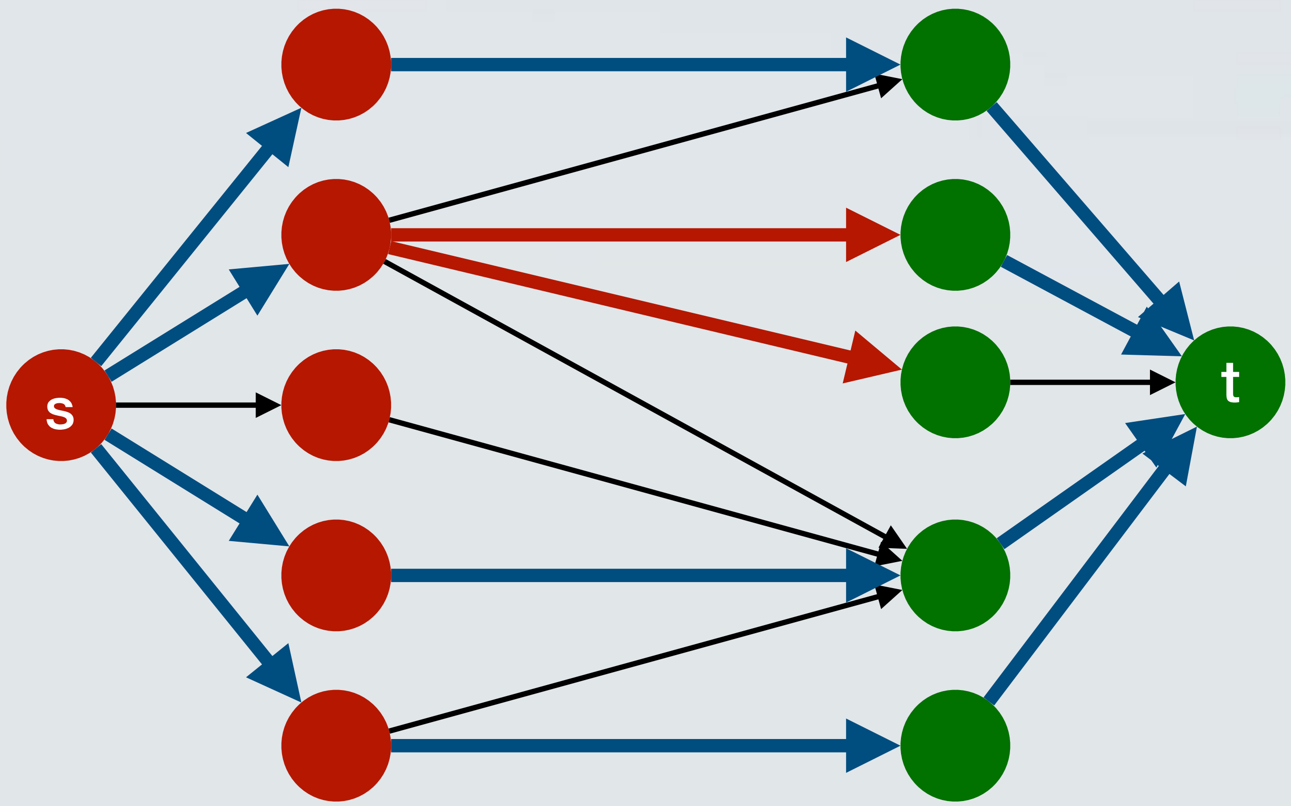
All capacities are set to 1.



From flows to matchings

- Consider the set M’ of edges with f(e) = *1*.
    - Claim: M’ is a matching.

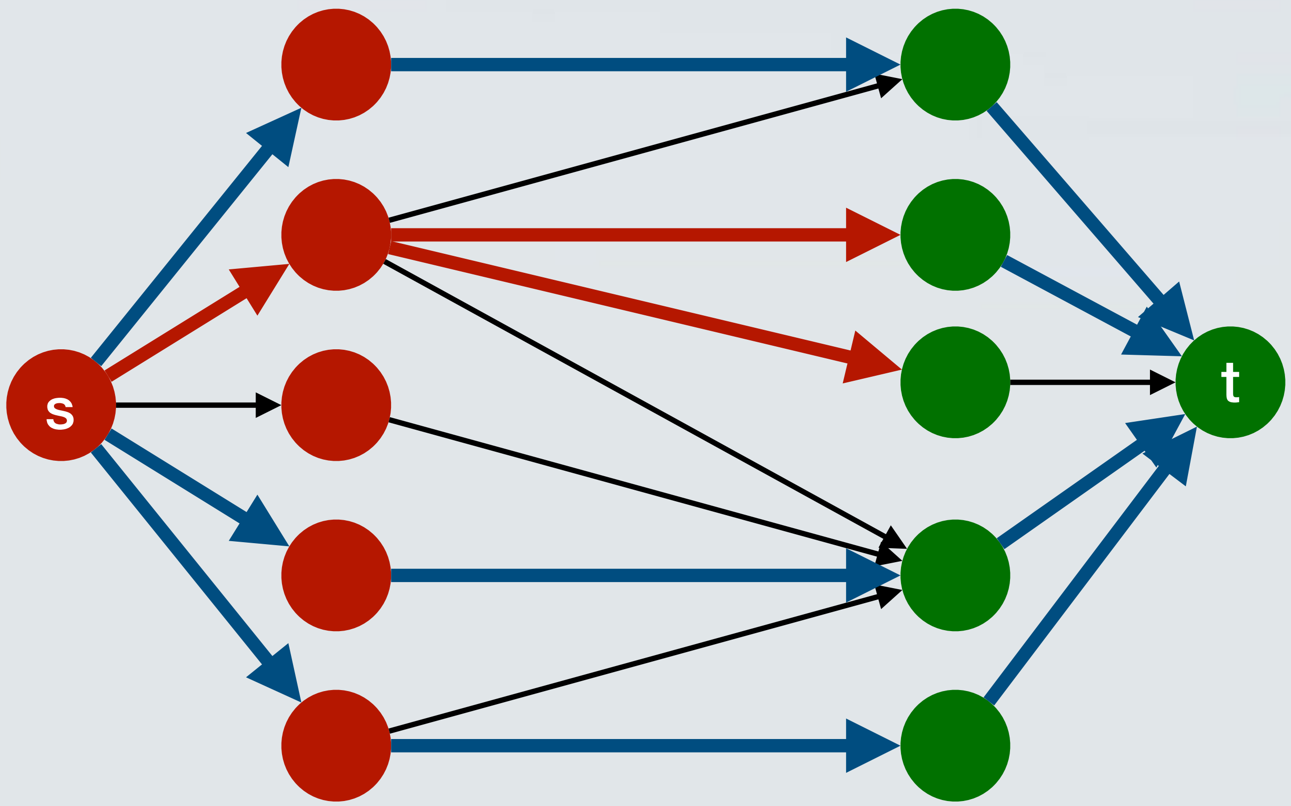
All capacities are set to 1.



From flows to matchings

- Consider the set M' of edges with f(e) = *1*.
  - Claim: M' is a matching.

All capacities are set to 1.



Maximum Flow and Maximum matching

- The size of the maximum matching M in G is equal to the value of the maximum flow f in G^f.
- The edges of M are the edges that carry flow from A to B in G^f.

Maximum Flow and Maximum matching

- The size of the maximum matching M in G is equal to the value of the maximum flow f in G^f.
- The edges of M are the edges that carry flow from A to B in G^f.
- What was the crucial part, that allows us to establish this?

Maximum Flow and Maximum matching

- The size of the maximum matching M in G is equal to the value of the maximum flow f in G^f.
- The edges of M are the edges that carry flow from A to B in G^f.
- What was the crucial part, that allows us to establish this?
  - The integrality theorem.

Running time

- What is the running time of the algorithm?
- By Edmonds - Karp, we get **O(**nm^2**).**

Running time

- What is the running time of the algorithm?
- By Ford-Fulkerson, we get **O(**mF**).**

Running time

- What is the running time of the algorithm?
- By Ford-Fulkerson, we get **O(**mF**).**
- How large is F here?

Running time

- What is the running time of the algorithm?
- By Ford-Fulkerson, we get **O(mF).**
- How large is F here?
    - It is at most max{|A|, |B|}.

Running time

- What is the running time of the algorithm?
- By Ford-Fulkerson, we get **O(**mF**).**
- How large is F here?
    - It is at most max{|A|, |B|}.
    - Running time **O(**nm**).**

Baseball Elimination

- In the baseball league, there are 4 teams with the following number of wins:`

New York	92
Baltimore	91
Toronto	91
Boston	90

- There are five games left in the season.
  - NY vs BLT, NY vs TOR, BLT vs TOR, BLT vs BOS, TOR vs BOS
- **Question:** Can Boston finish (possibly tied for) first?

Baseball Elimination

- In the baseball league, there are 4 teams with the following number of wins:`

Assume Boston wins all remaining games.

New York	92
Baltimore	91
Toronto	91
Boston	90

- There are five games left in the season.
  - NY vs BLT, NY vs TOR, BLT vs TOR, BLT vs BOS, TOR vs BOS
- **Question:** Can Boston finish (possibly tied for) first?

Baseball Elimination

- In the baseball league, there are 4 teams with the following number of wins:`

New York	92
Baltimore	91
Toronto	91
Boston	90

Assume Boston wins all remaining games.

New York must lose all remaining games.

- There are five games left in the season.
  - NY vs BLT, NY vs TOR, BLT vs TOR, BLT vs BOS, TOR vs BOS
- **Question:** Can Boston finish (possibly tied for) first?

Baseball Elimination

- In the baseball league, there are 4 teams with the following number of wins:`

New York	92
Baltimore	91
Toronto	91
Boston	90

Assume Boston wins all remaining games.

New York must lose all remaining games.

Baltimore and Toronto must win one game each.

- There are five games left in the season.
  - NY vs BLT, NY vs TOR, BLT vs TOR, BLT vs BOS, TOR vs BOS
- **Question:** Can Boston finish (possibly tied for) first?

Baseball Elimination

- In the baseball league, there are 4 teams with the following number of wins:`

New York	92
Baltimore	91
Toronto	91
Boston	90

Assume Boston wins all remaining games.

New York must lose all remaining games.

Baltimore and Toronto must win one game each.

Baltimore or Toronto must win one more game (BLT vs TOR).

- There are five games left in the season.
  - NY vs BLT, NY vs TOR, BLT vs TOR, BLT vs BOS, TOR vs BOS
- **Question:** Can Boston finish (possibly tied for) first?

Baseball Elimination

- In the baseball league, there are 4 teams with the following number of wins:`

New York	92
Baltimore	91
Toronto	91
Boston	90

Assume Boston wins all remaining games.

Baltimore and Toronto must win one game each.

New York must lose all remaining games.

Baltimore or Toronto must win one more game (BLT vs TOR).

- There are five games left in the season.
  - NY vs BLT, NY vs TOR, BLT vs TOR, BLT vs BOS, TOR vs BOS
- **Question:** Can Boston finish (possibly tied for) first?

The answer is no.

Baseball Elimination

- In the baseball league, there are 4 teams with the following number of wins:`

  New York 90
  Baltimore 88
  Toronto 87
  Boston 79

- There are five games left in the season.
  - NY vs BLT
  - NY vs TOR ***6 games***
  - BLT vs TOR
  - BOS vs ANY ***4 games (12 games total)***
- **Question:** Can Boston finish (possibly tied for) first?

Baseball elimination

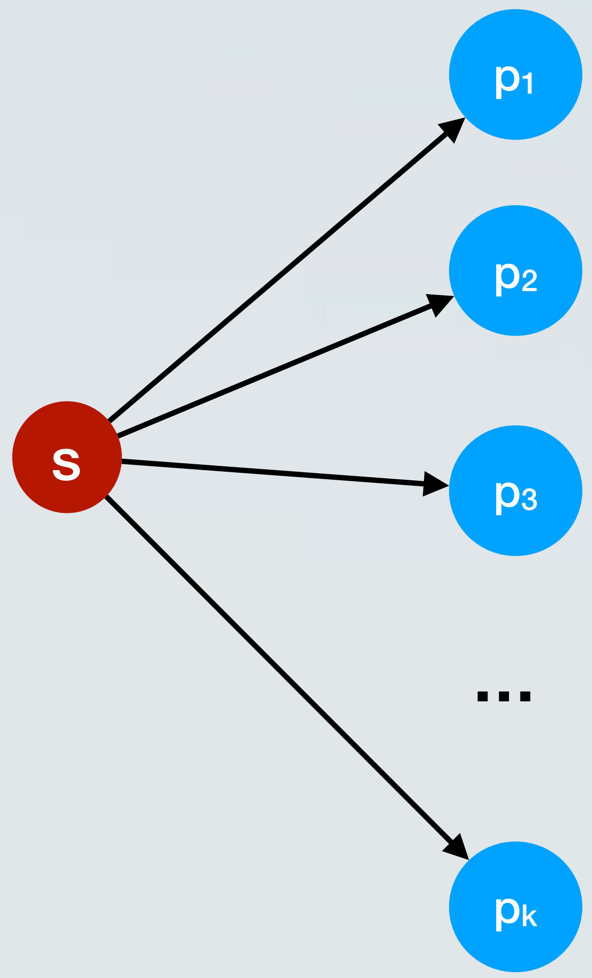
- Generally:
  - We have a set S of teams.
  - For each team x in S, the current number of wins is w_x.
  - For teams x and y in S, they still have to play g_{xy} games against each other.
  - We are given a designated team z.
  - Can z win the tournament (possibly in a tie?)

From baseball to flows

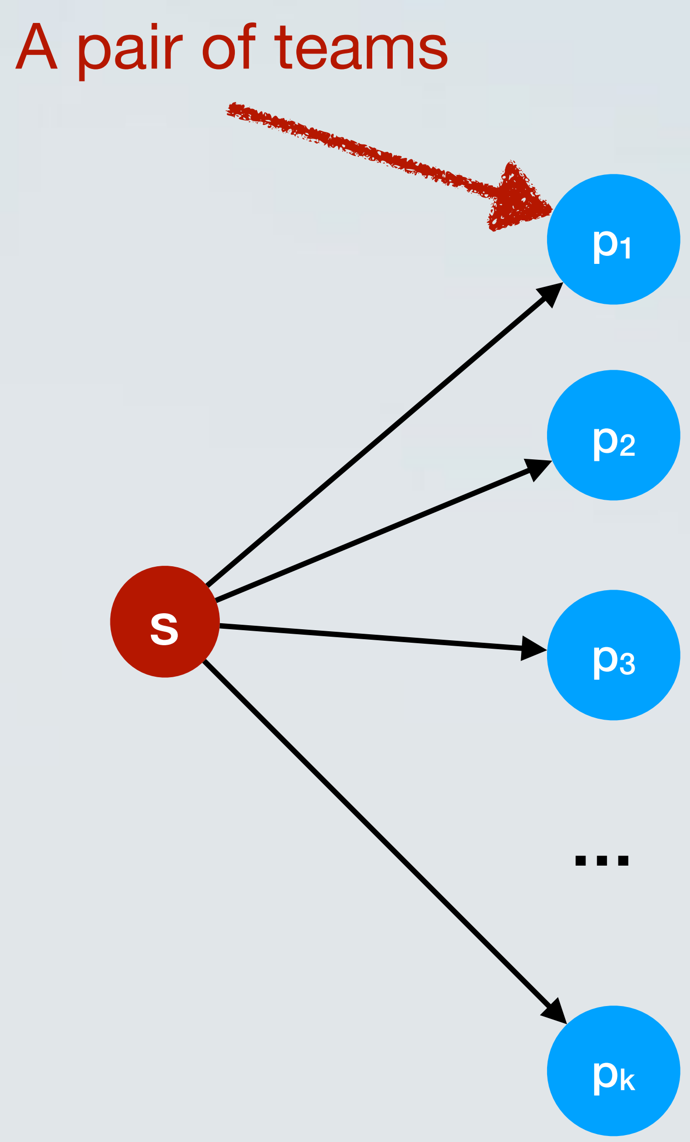
From baseball to flows

- Observation: If there is a way for z to be first, there is a way for z to be first *when winning all remaining games*.
  - Suppose that in the end, team z has m wins.
  - What are we looking for?
    - Is there an allocation of all the remaining g^* games (between the other teams) such that no team ends up with more than m wins?

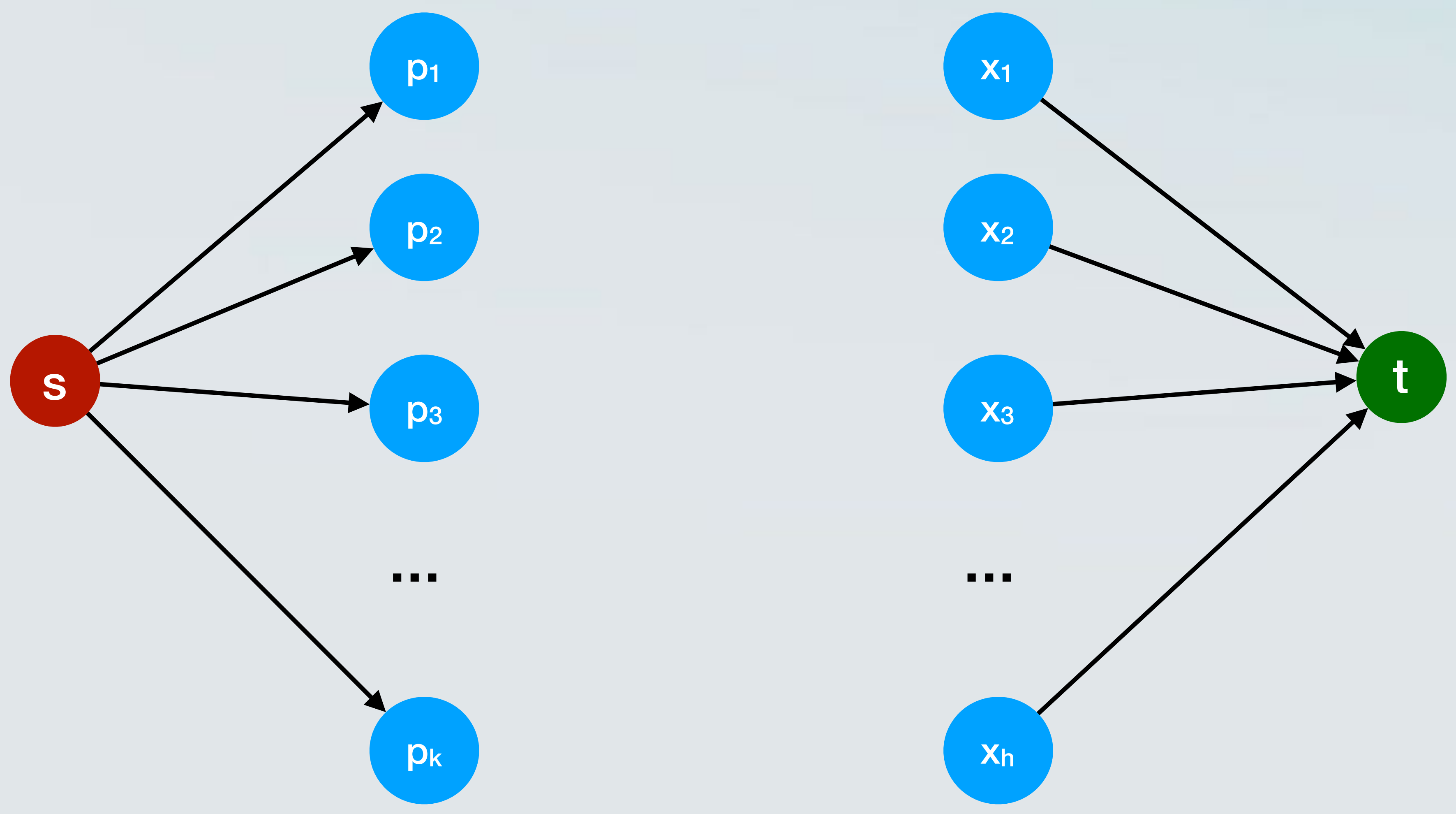
From baseball to flows



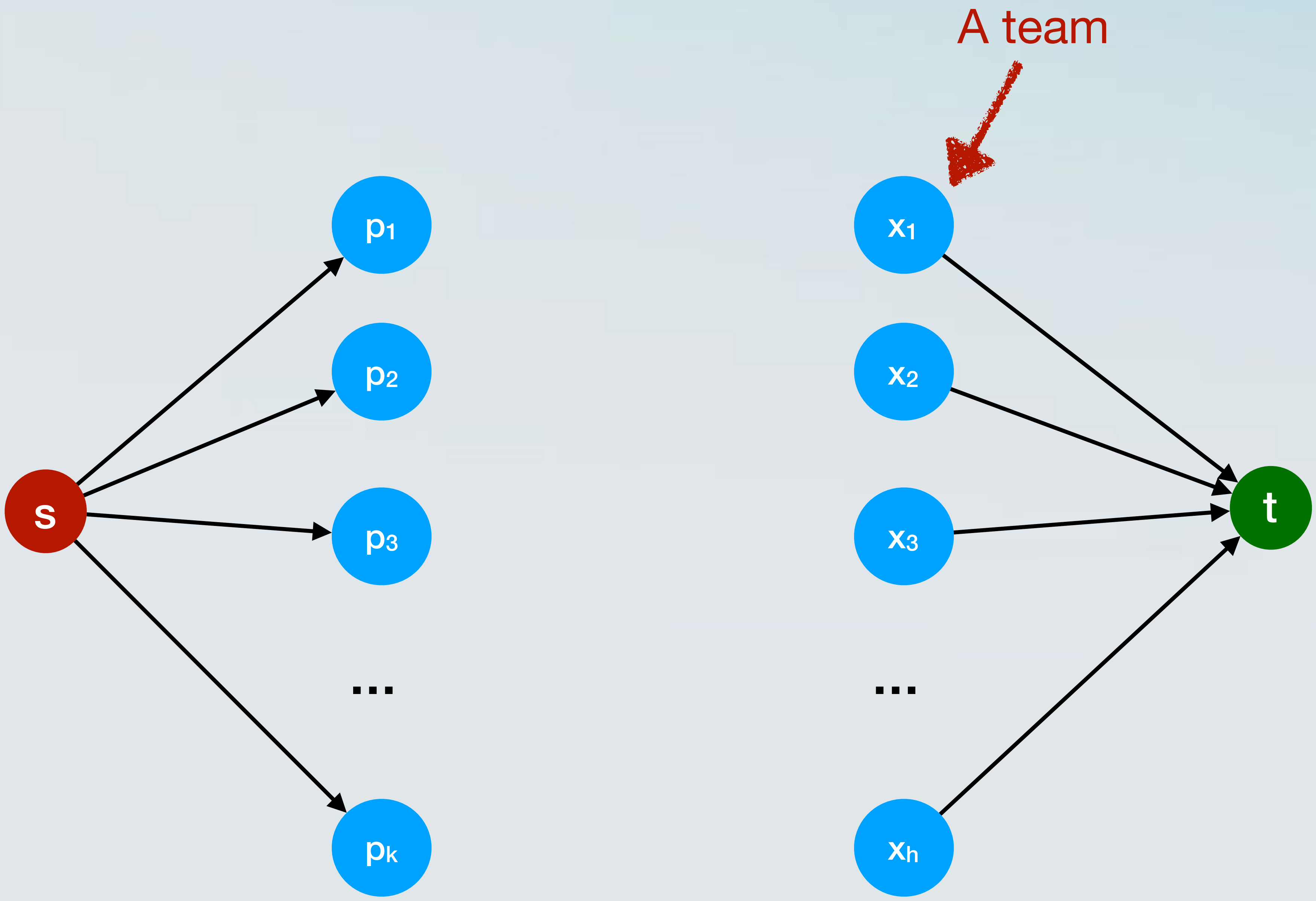
From baseball to flows



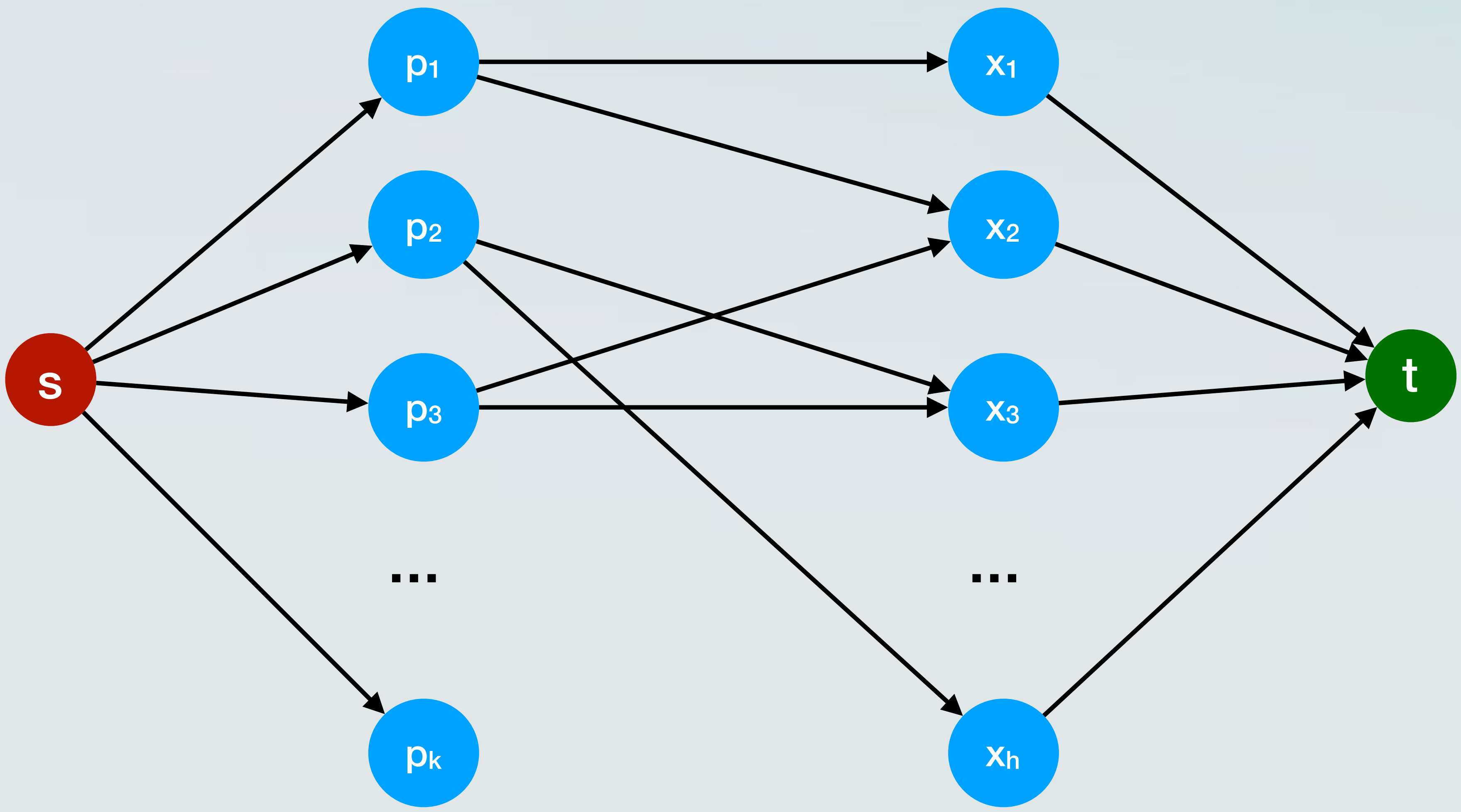
From baseball to flows



From baseball to flows

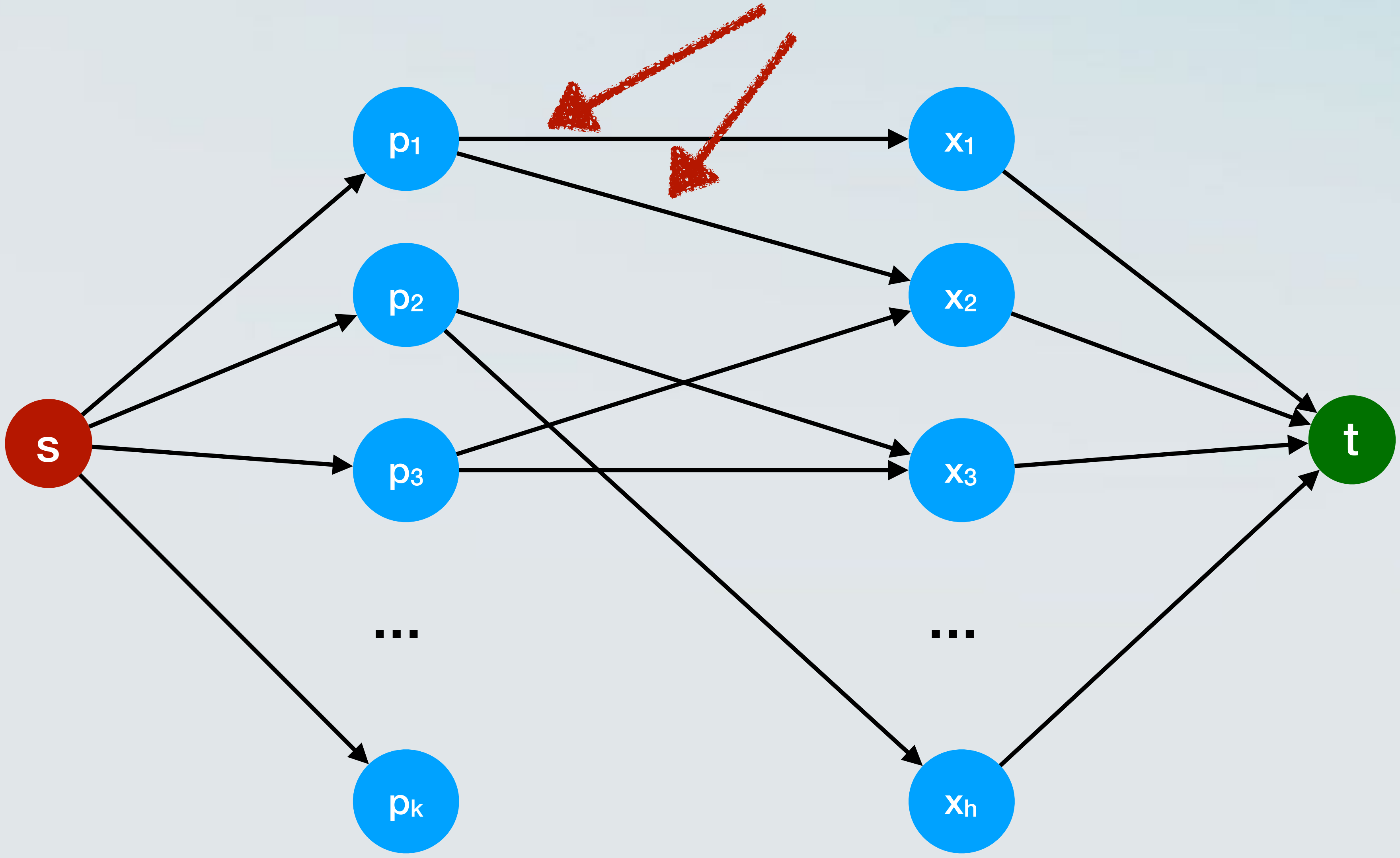


From baseball to flows



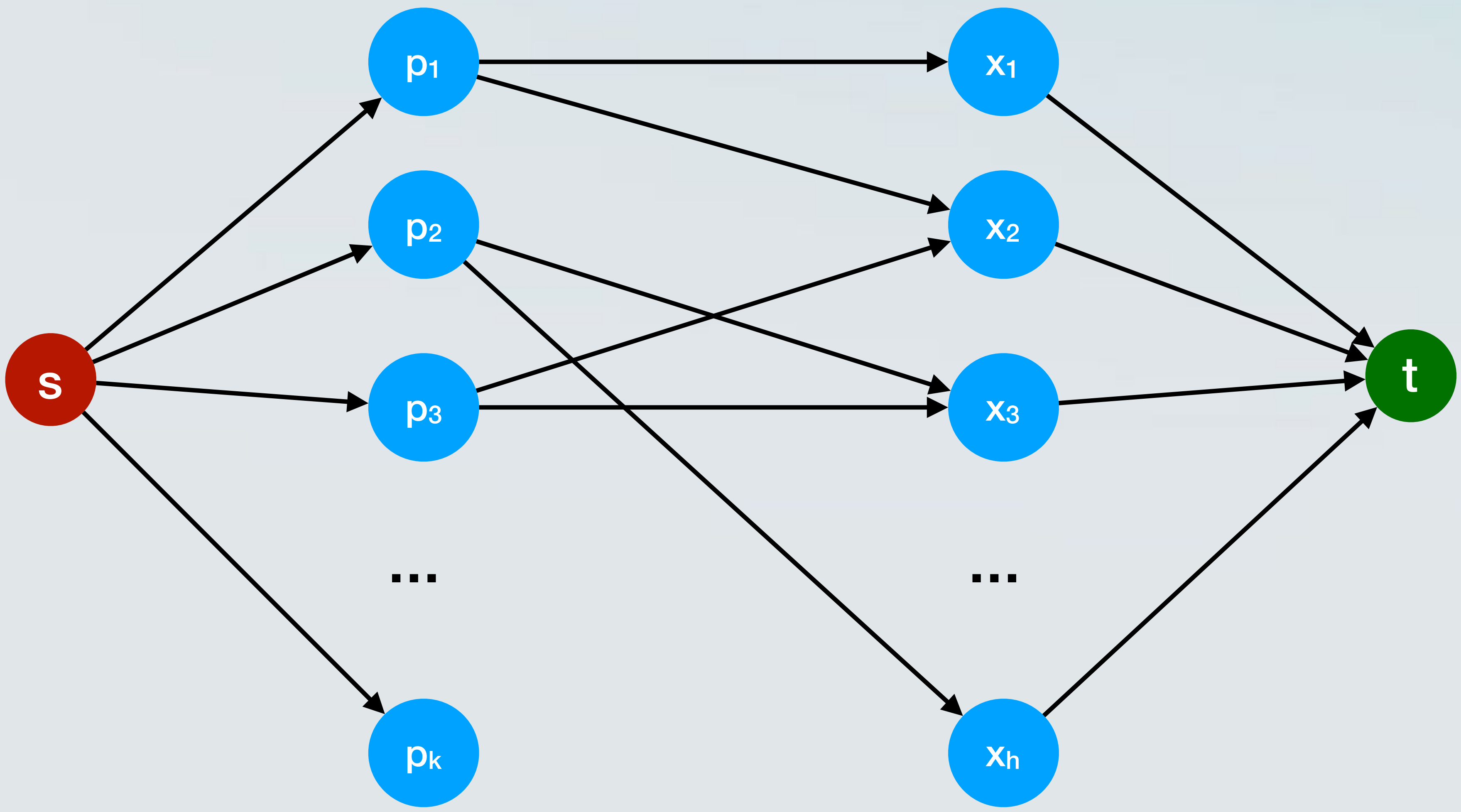
From baseball to flows

Two edges if teams in p_j still have games to play between them.



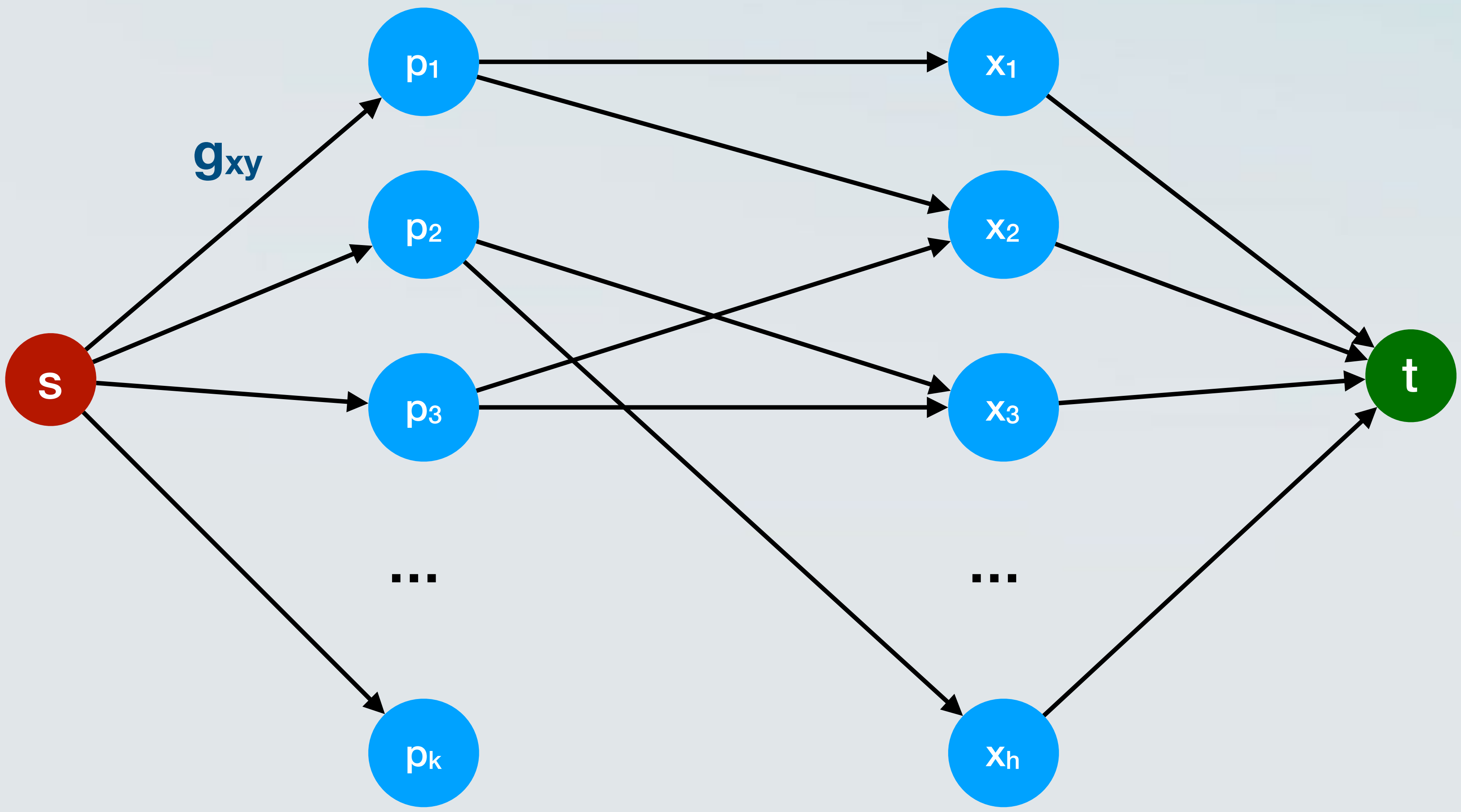
From baseball to flows

Let p_j = (x, y)



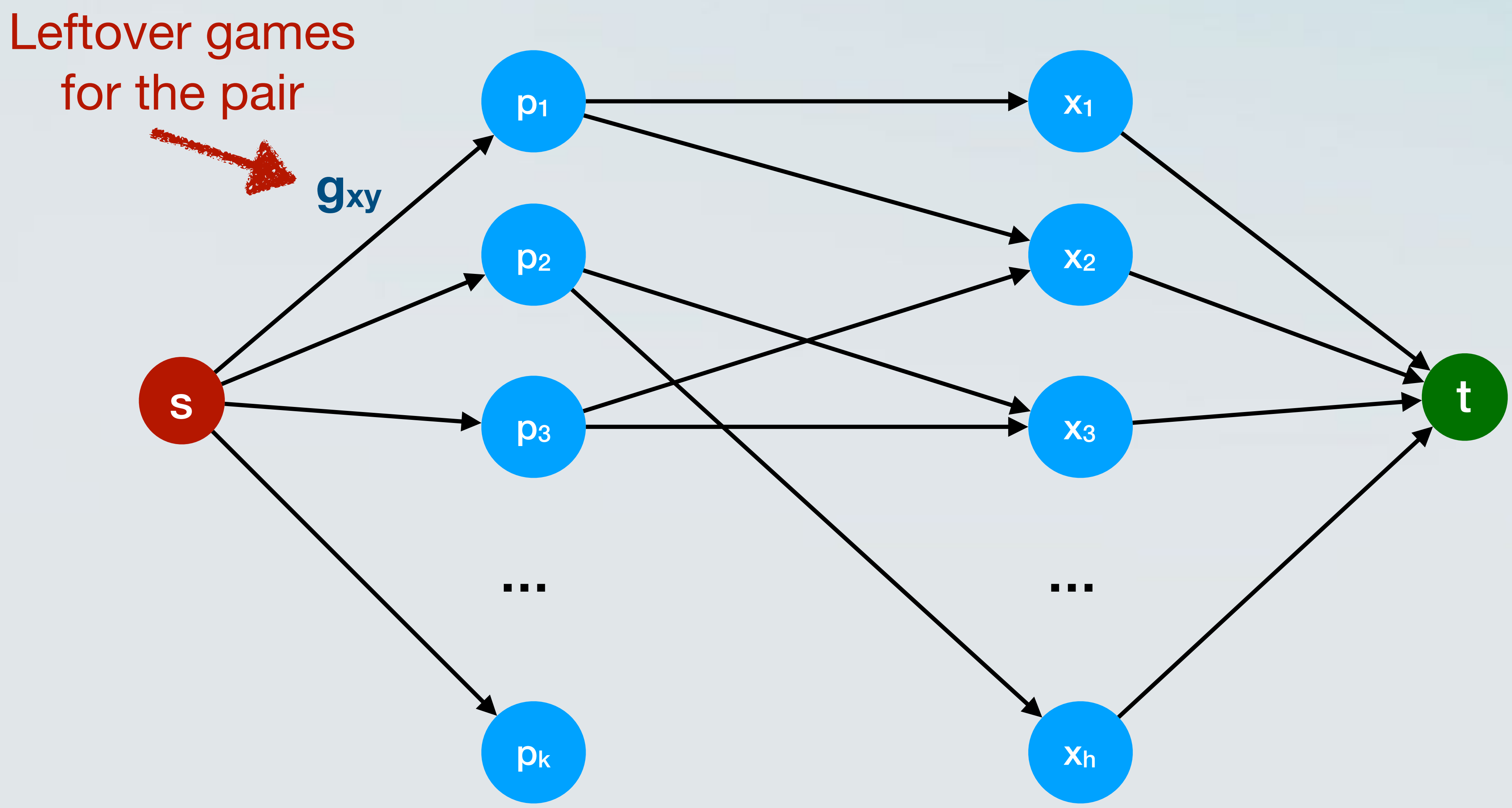
From baseball to flows

Let $p_j = (x, y)$

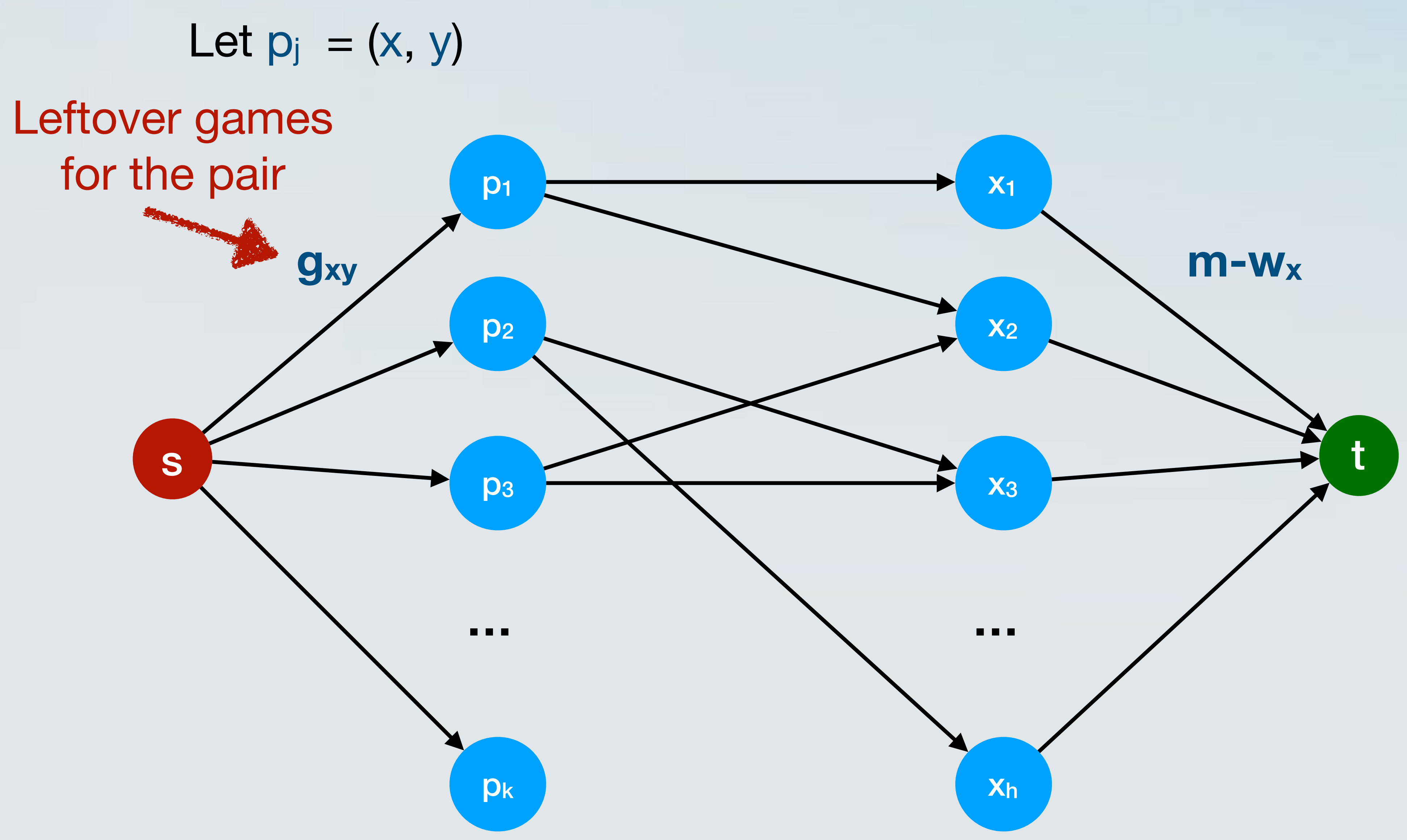


From baseball to flows

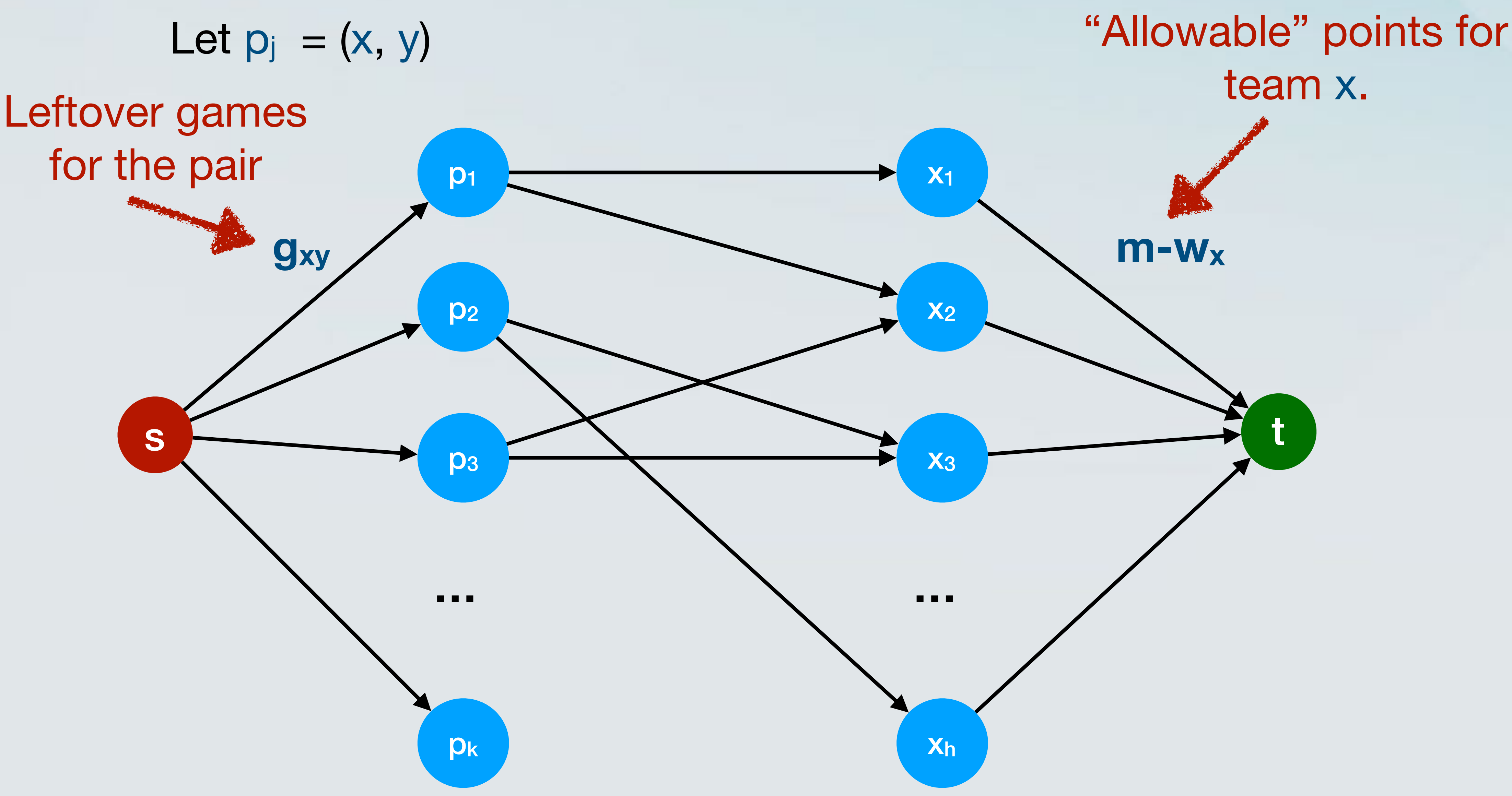
Let $p_j = (x, y)$



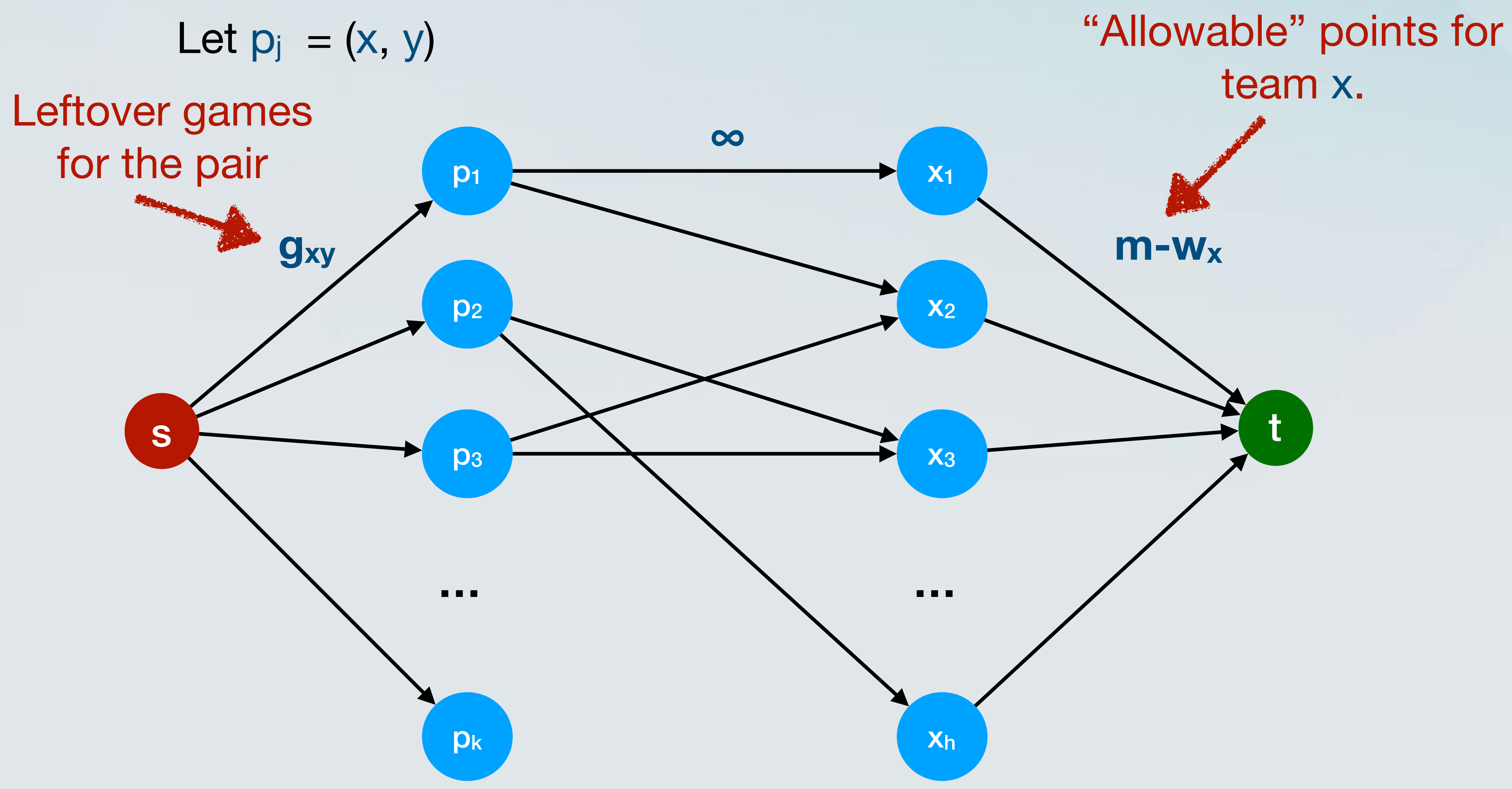
From baseball to flows



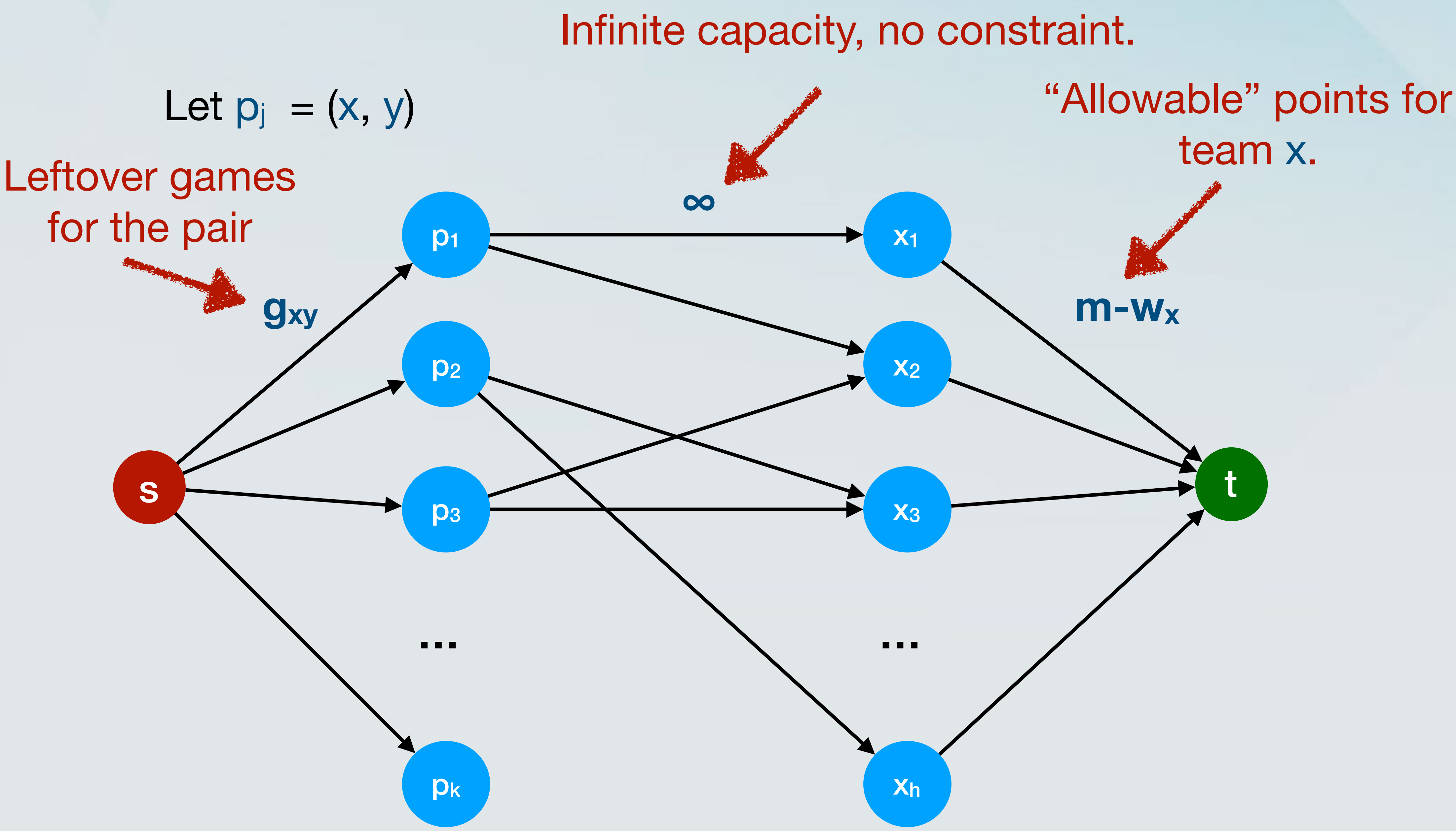
From baseball to flows



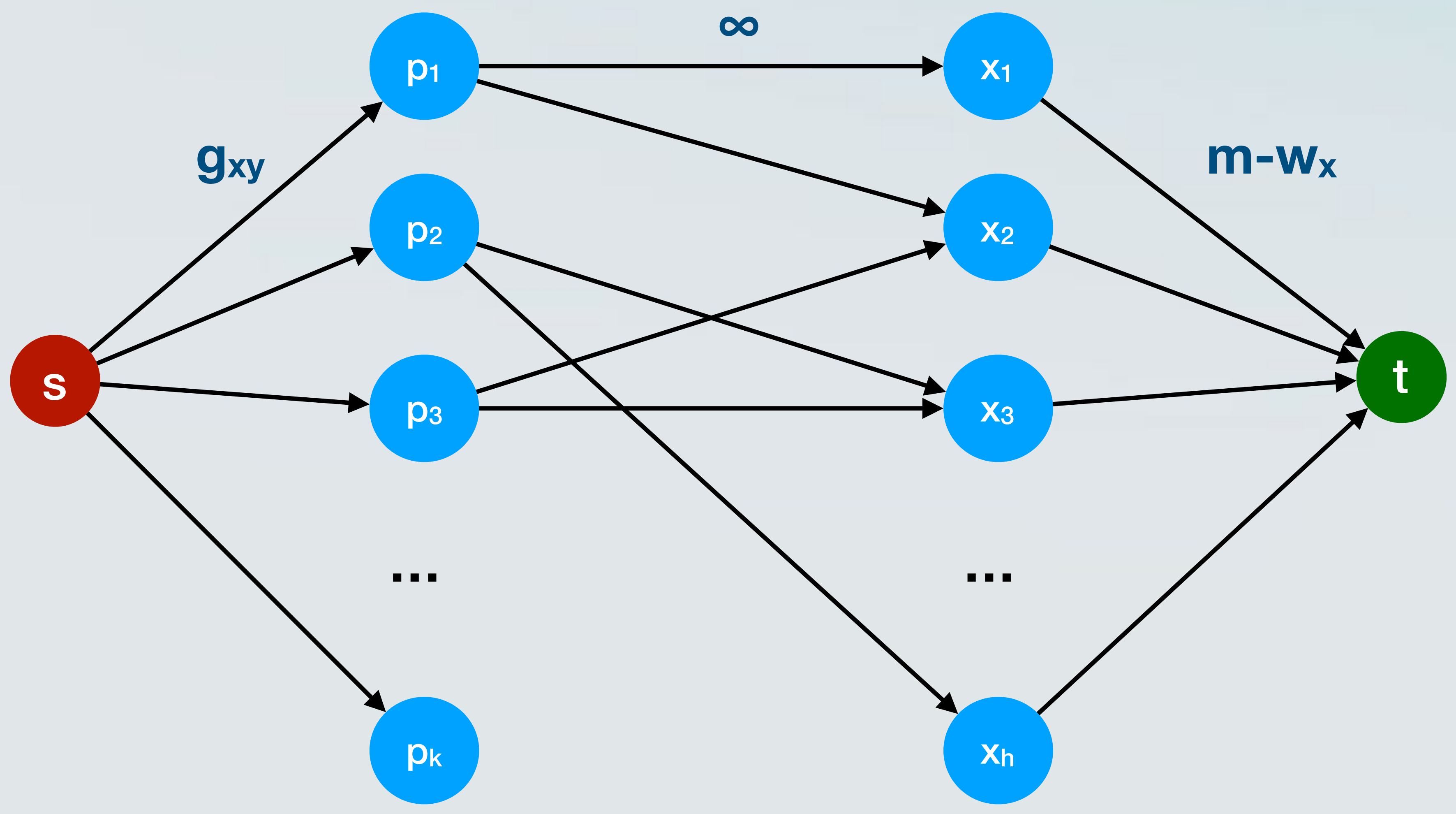
From baseball to flows



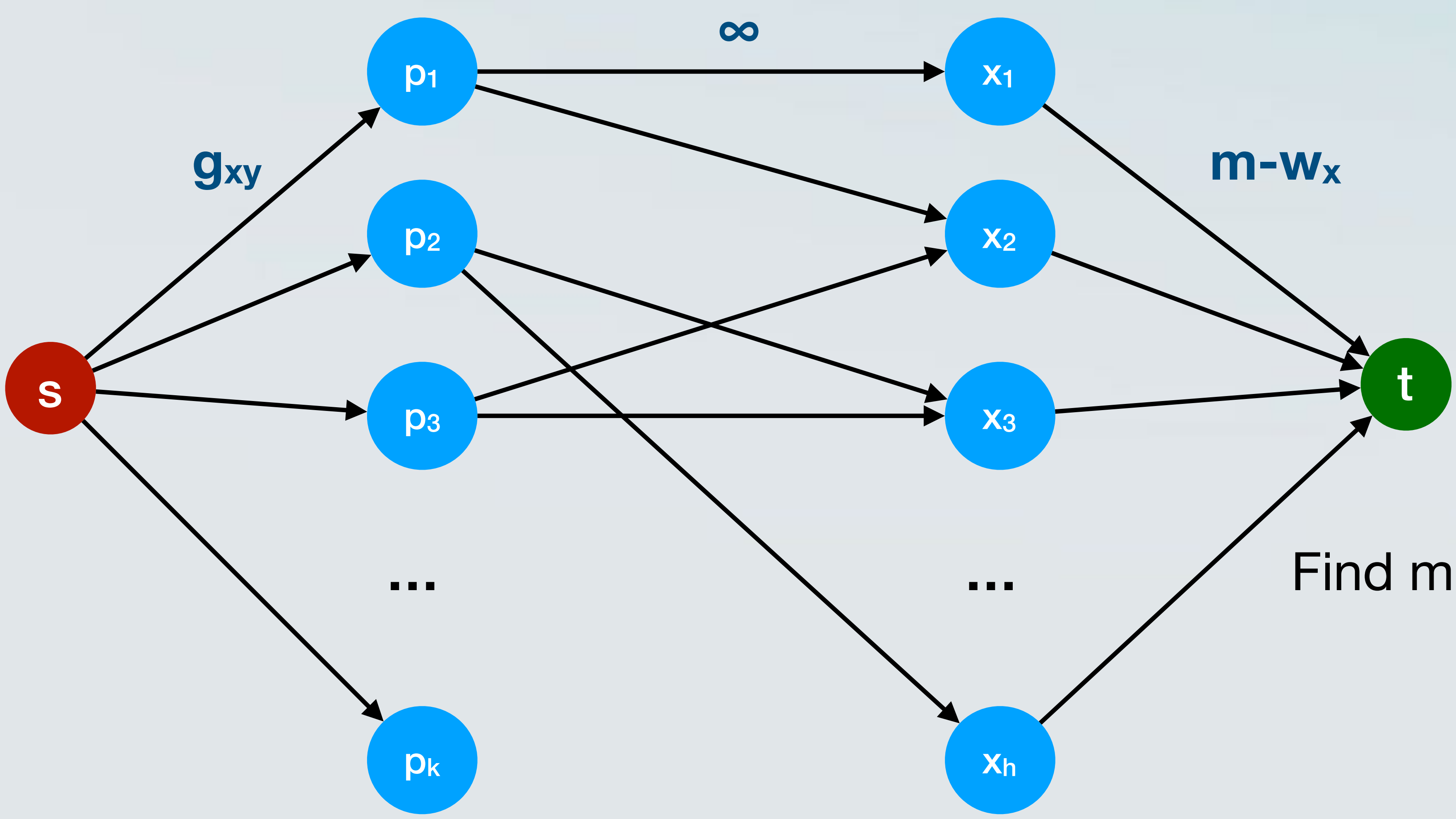
From baseball to flows



From baseball to flows

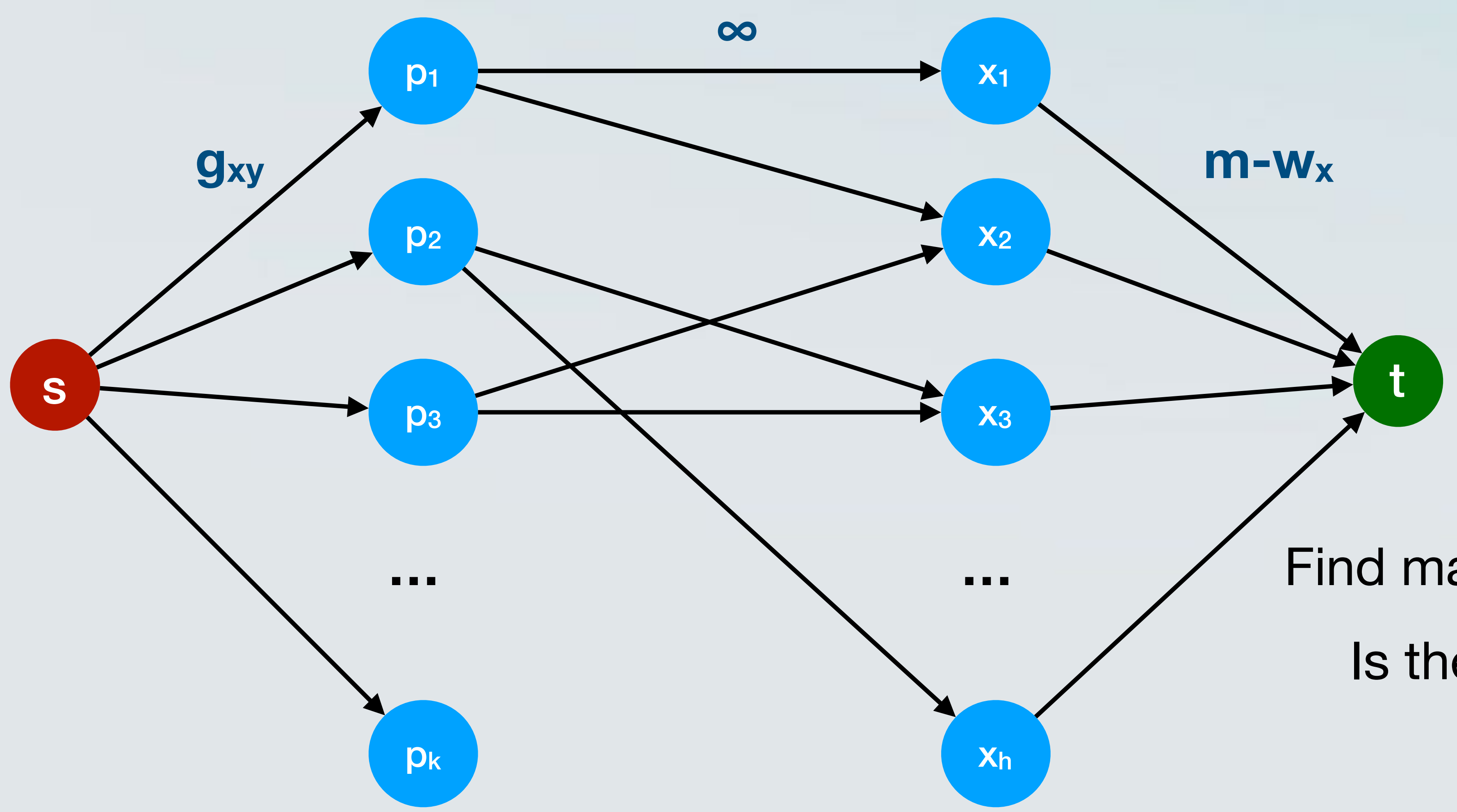


From baseball to flows



Find max flow in the network

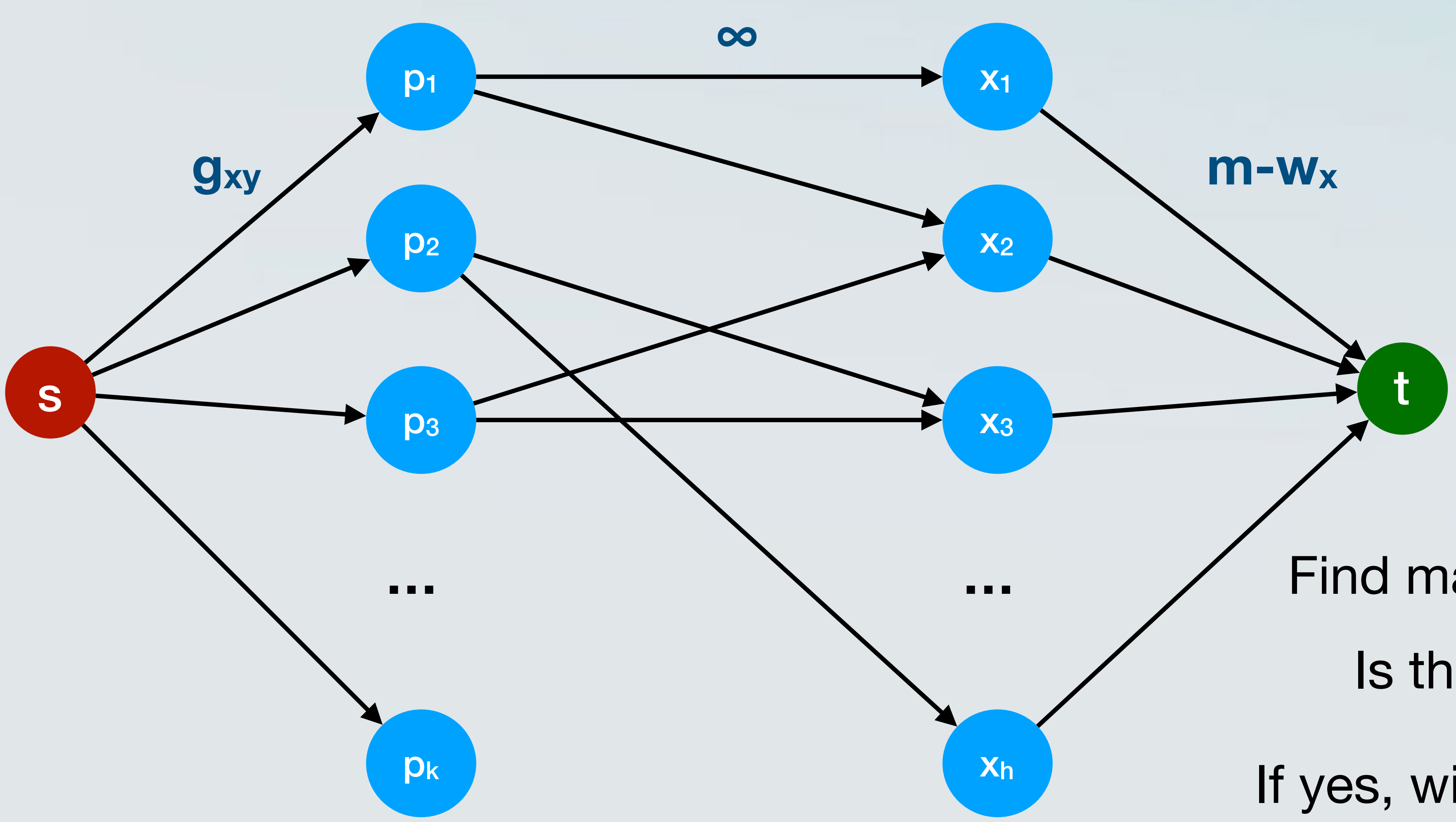
From baseball to flows



Find max flow in the network

Is the value at least g^*?

From baseball to flows

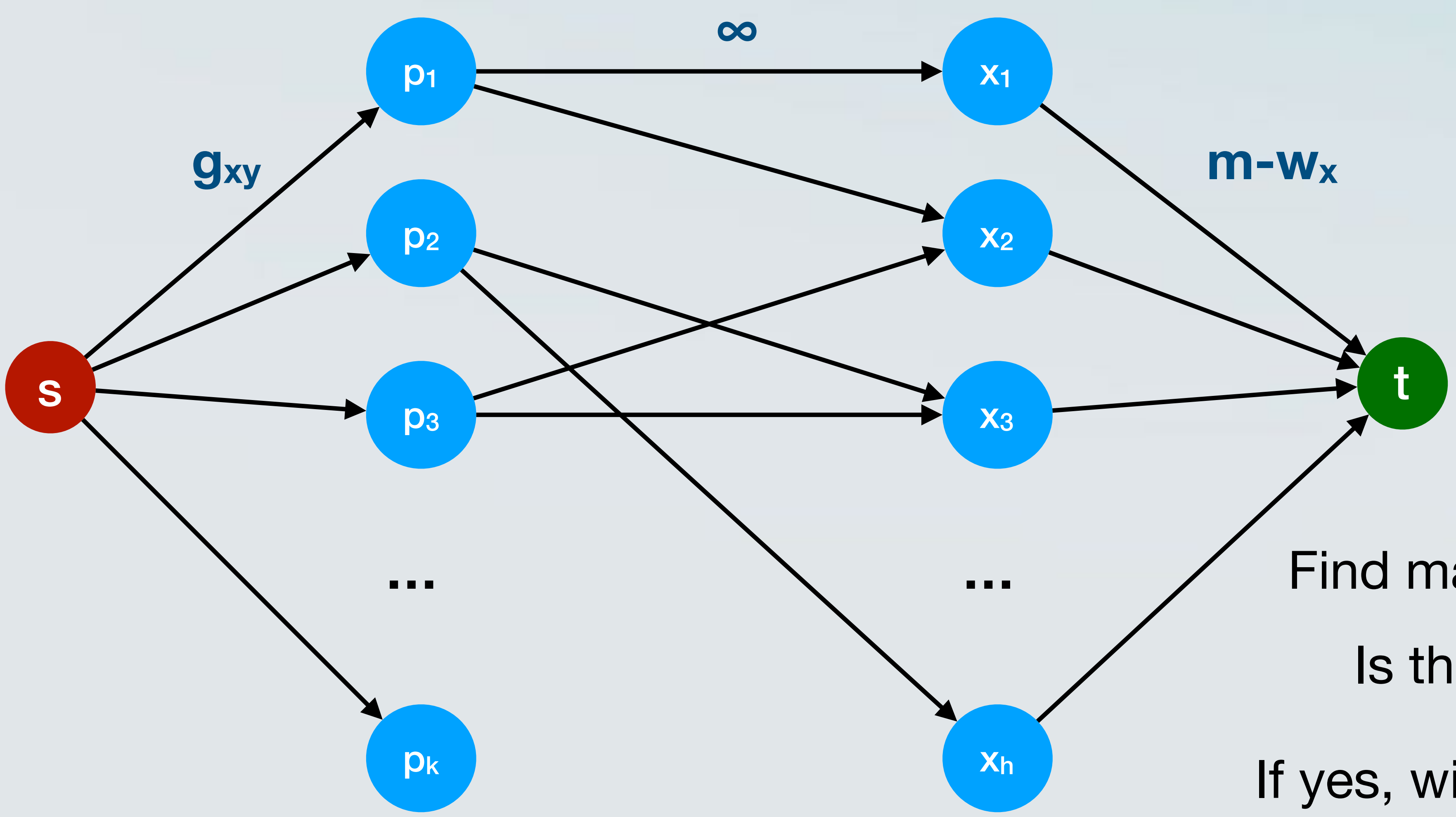


Find max flow in the network

Is the value at least g^*?

If yes, winning is still possible

From baseball to flows



Find max flow in the network

Is the value at least g^*?

If yes, winning is still possible
If no, winning is not possible

Why does this work?

Why does this work?

- Assume that the algorithm says yes.

Why does this work?

- Assume that the algorithm says yes.
  - The value of the flow is equal to the number of remaining games. (why?)

Why does this work?

- Assume that the algorithm says yes.
  - The value of the flow is equal to the number of remaining games. (why?)
  - Flow conservation implies:

Why does this work?

- Assume that the algorithm says yes.
  - The value of the flow is equal to the number of remaining games. (why?)
  - Flow conservation implies:
    - A pair (x, y) will play exactly g_{xy} games.

Why does this work?

- Assume that the algorithm says yes.
  - The value of the flow is equal to the number of remaining games. (why?)
  - Flow conservation implies:
    - A pair (x, y) will play exactly g_{xy} games.
    - A team x will win at most $m-w_x$ games.

Why does this work?

- Assume that the algorithm says yes.
  - The value of the flow is equal to the number of remaining games. (why?)
  - Flow conservation implies:
    - A pair (x, y) will play exactly g_{xy} games.
    - A team x will win at most $m-w_x$ games.
    - Team z can win.

Why does this work?

Why does this work?

- Assume that the algorithm says no.

Why does this work?

- Assume that the algorithm says no.
  - The maximum flow has value ≤ g^*.

Why does this work?

- Assume that the algorithm says no.
  - The maximum flow has value $\leq g^*$.
  - It is not possible to play all the remaining games without giving some team x more than $m - w_x$ points.

Why does this work?

- Assume that the algorithm says no.
  - The maximum flow has value ≤ g^*.
  - It is not possible to play all the remaining games without giving some team x more than $m - w_x$ points.
  - Team z cannot win.

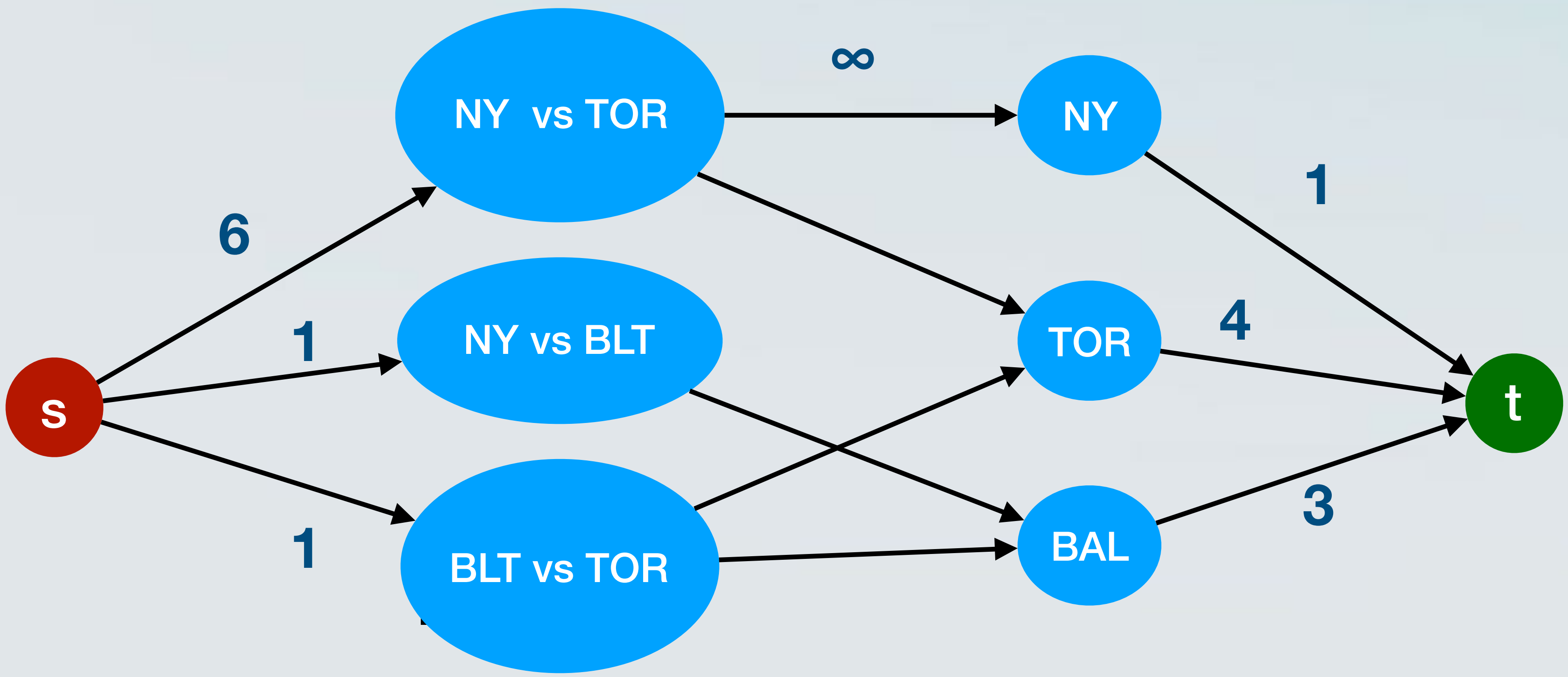
Example

- In the baseball league, there are 4 teams with the following number of wins:`

  New York 90
  Baltimore 88
  Toronto 87
  Boston 79

- There are five games left in the season.
  - NY vs BLT
  - NY vs TOR ***6 games***
  - BLT vs TOR
  - BOS vs ANY ***4 games (12 games total)***
- **Question:** Can Boston finish (possibly tied for) first?

Example

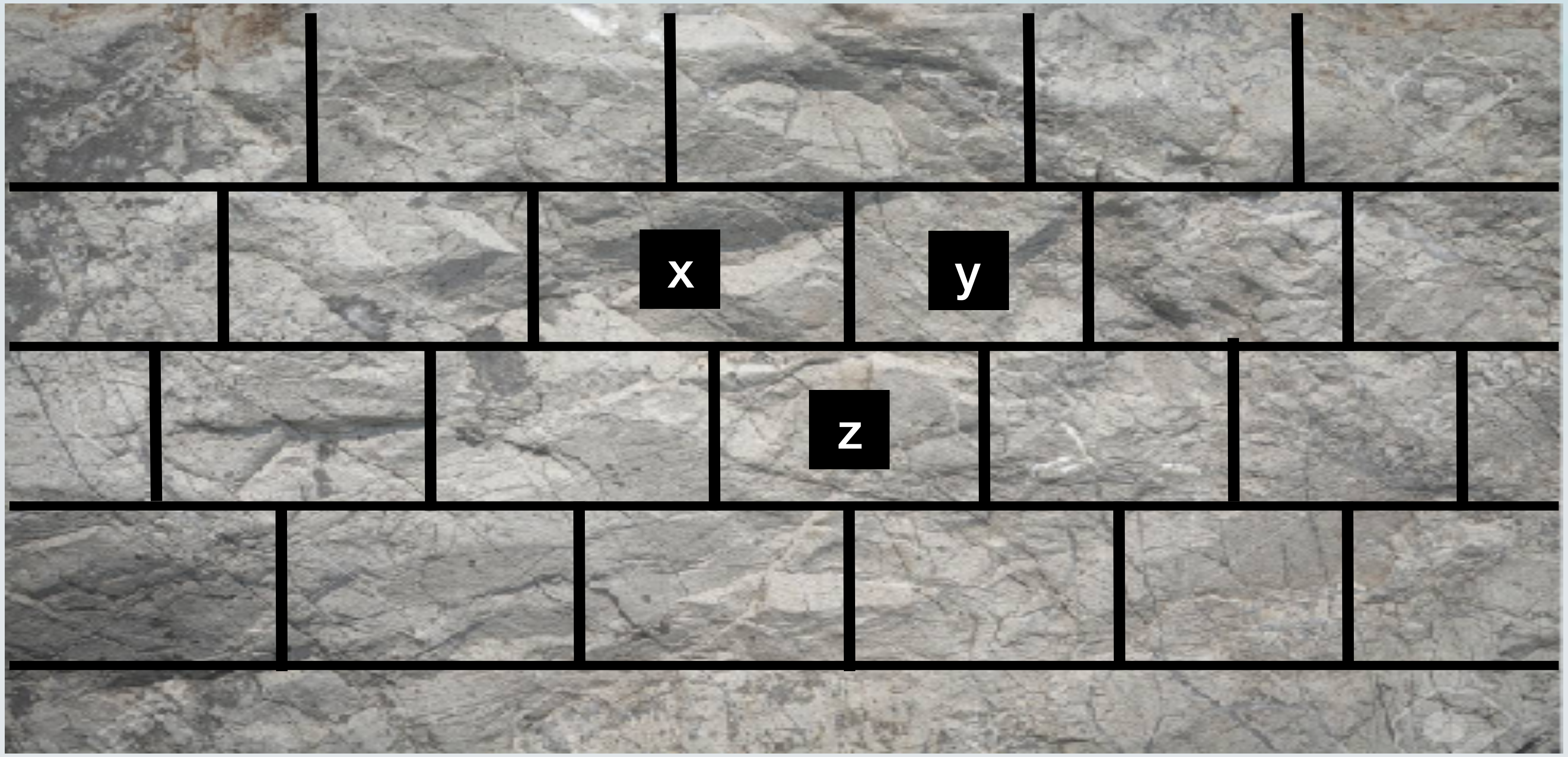


m = 91

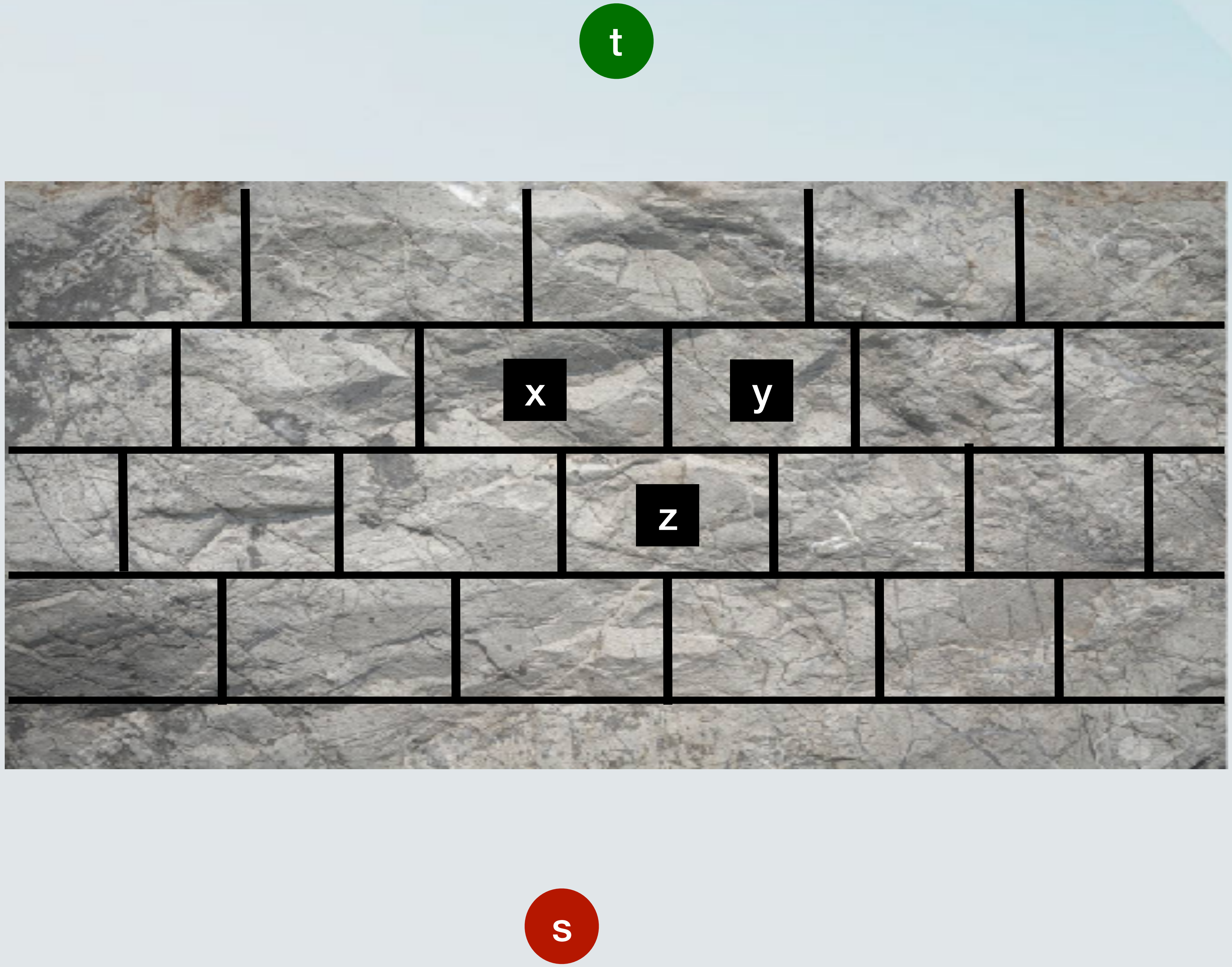
Open pit mining

- We extract blocks of earth from the surface, trying to find gold.
- Each block z that we mine has
  - a value p_z
  - a mining cost c_z
- Constraint: We can not mine a block z unless we mine the two blocks x and y on top of it.
- We want to earn as much money as possible.

Open pit mining

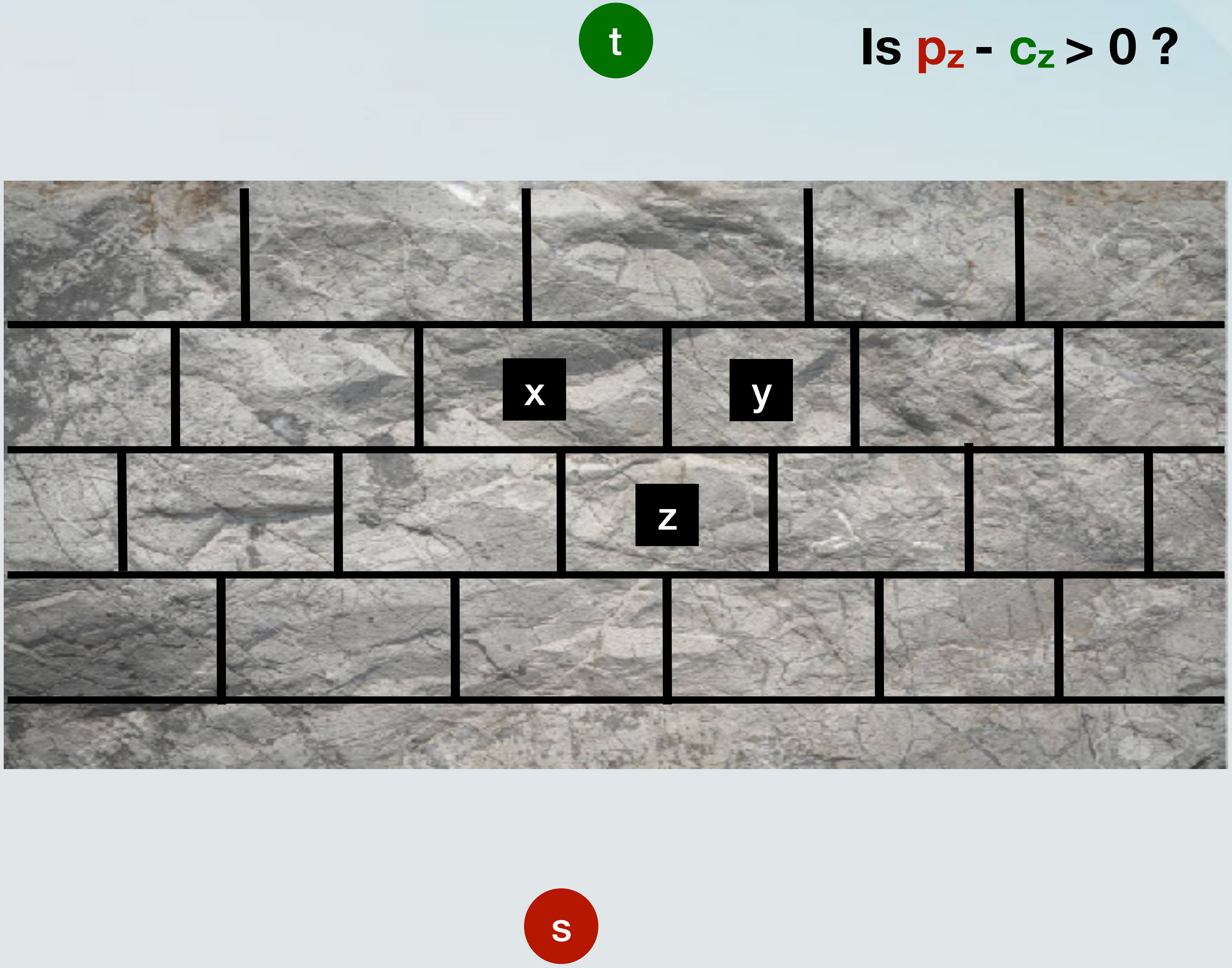


From pits to flows

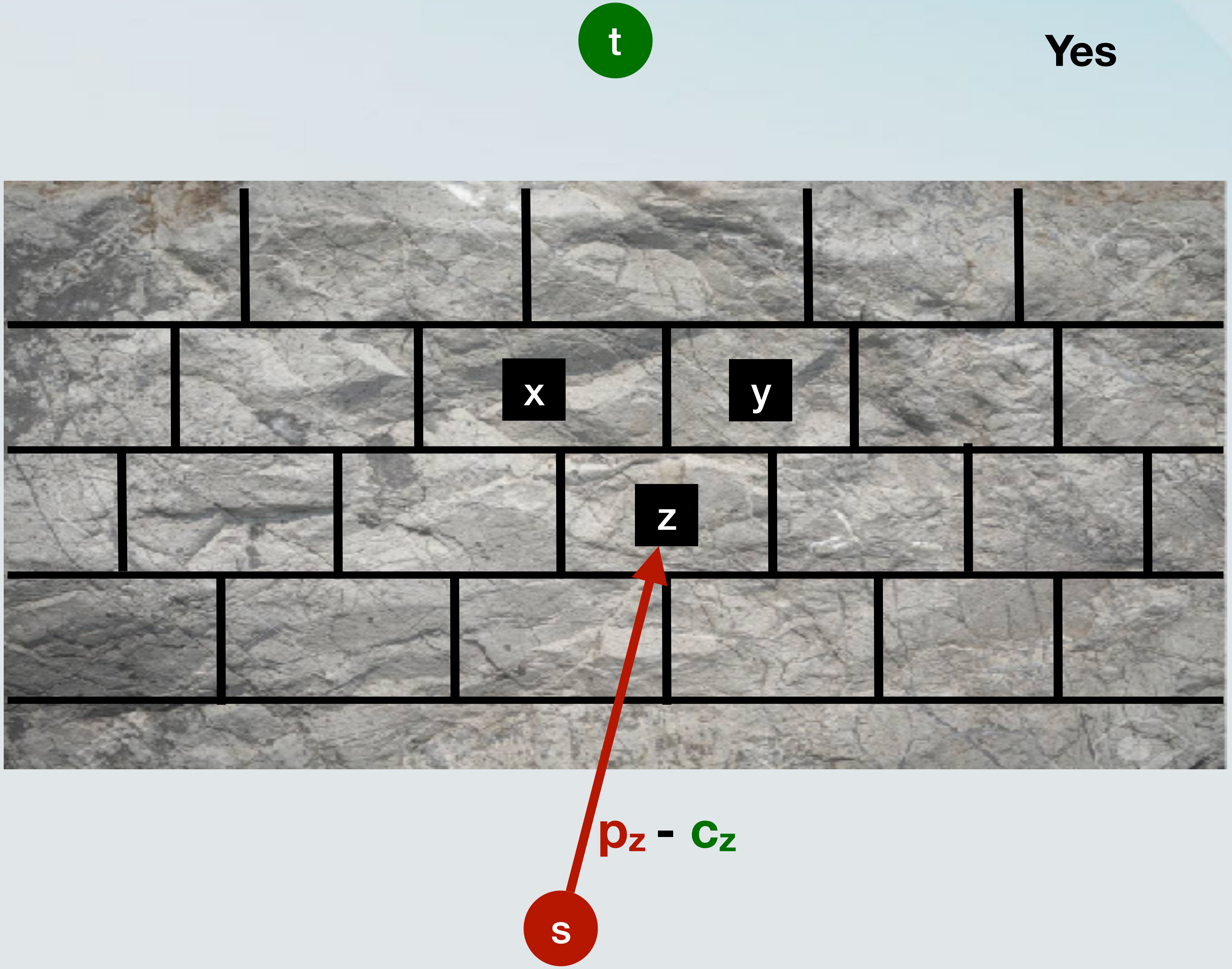


From pits to flows

Is $p_z - c_z > 0$?

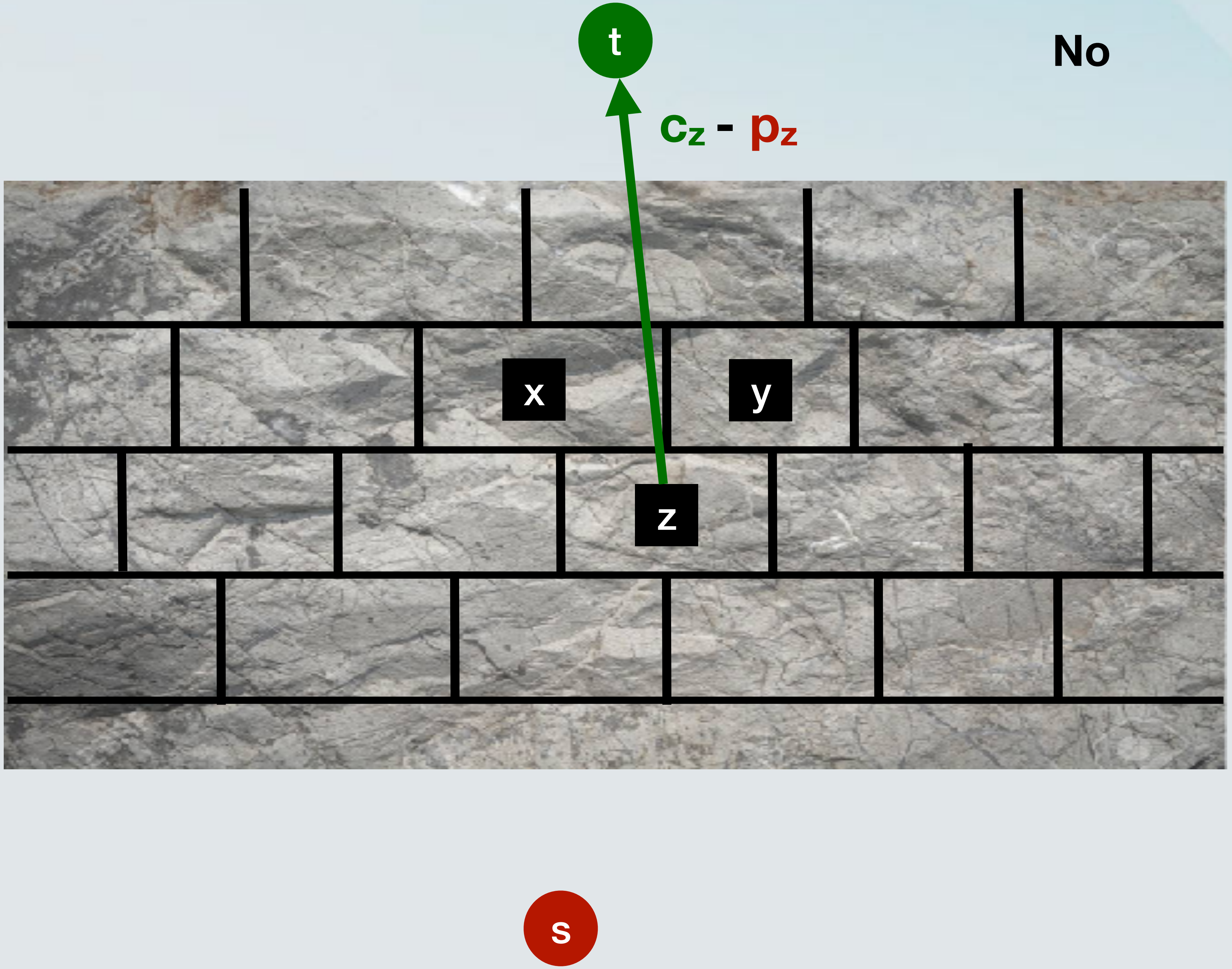


From pits to flows

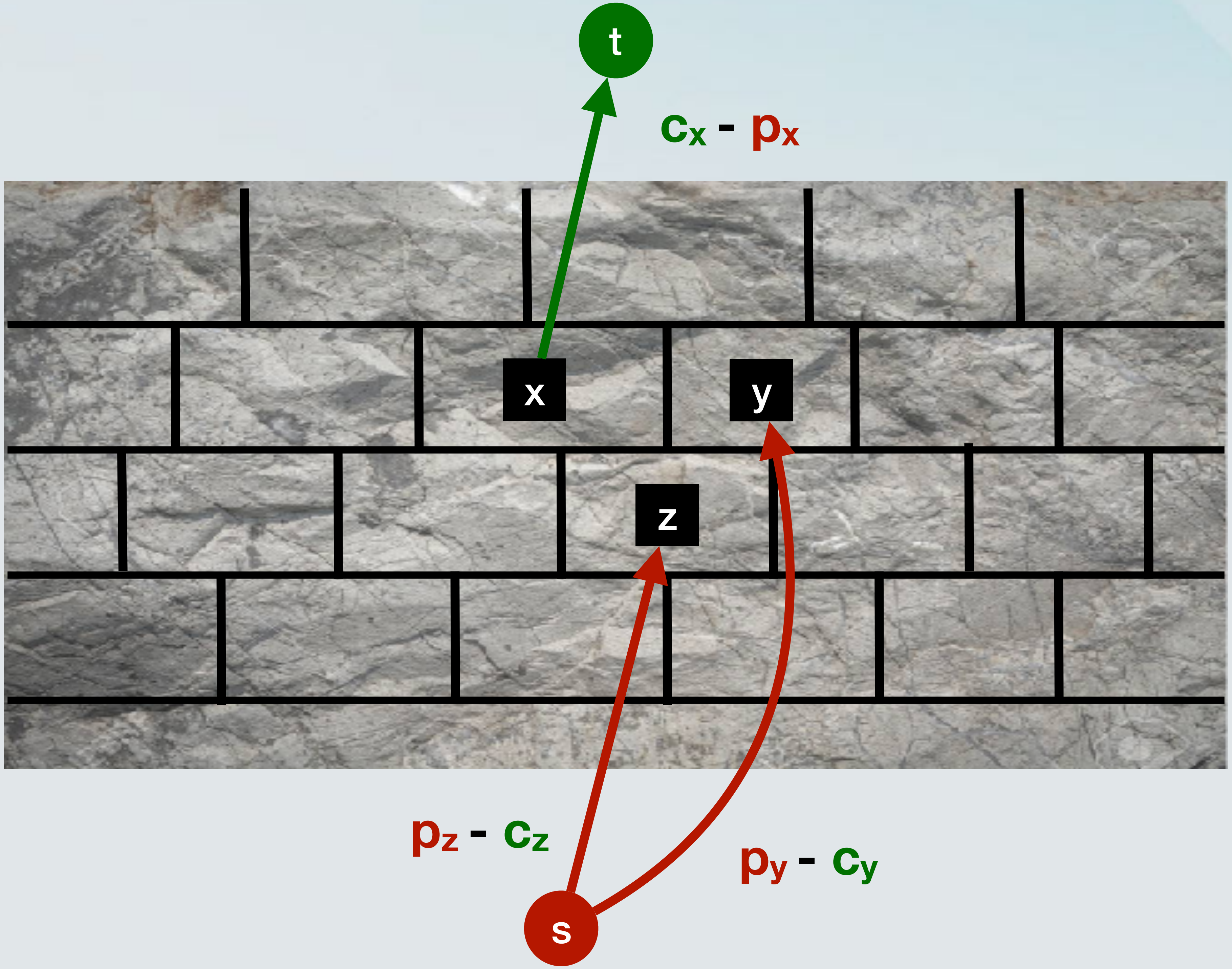


From pits to flows

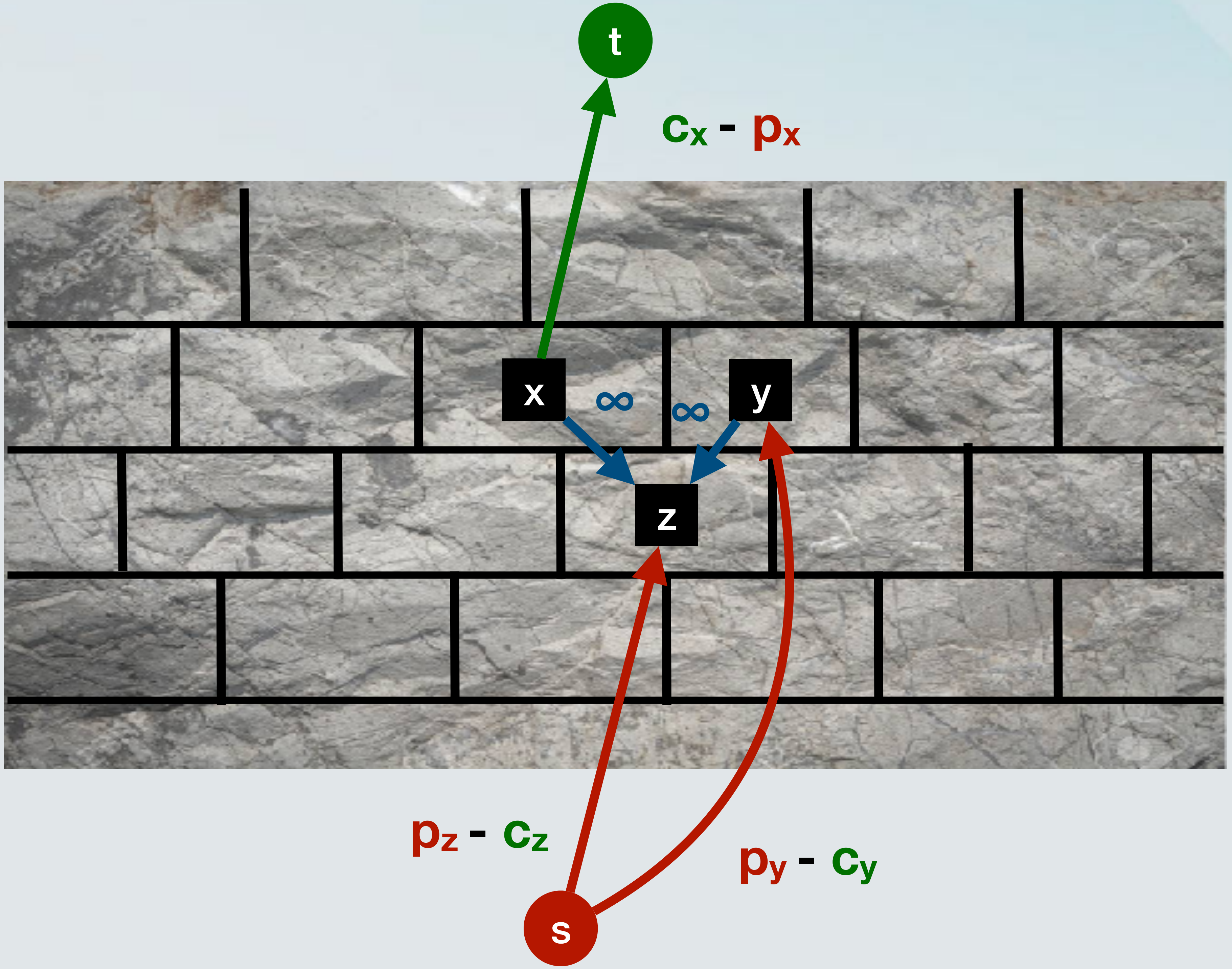
No



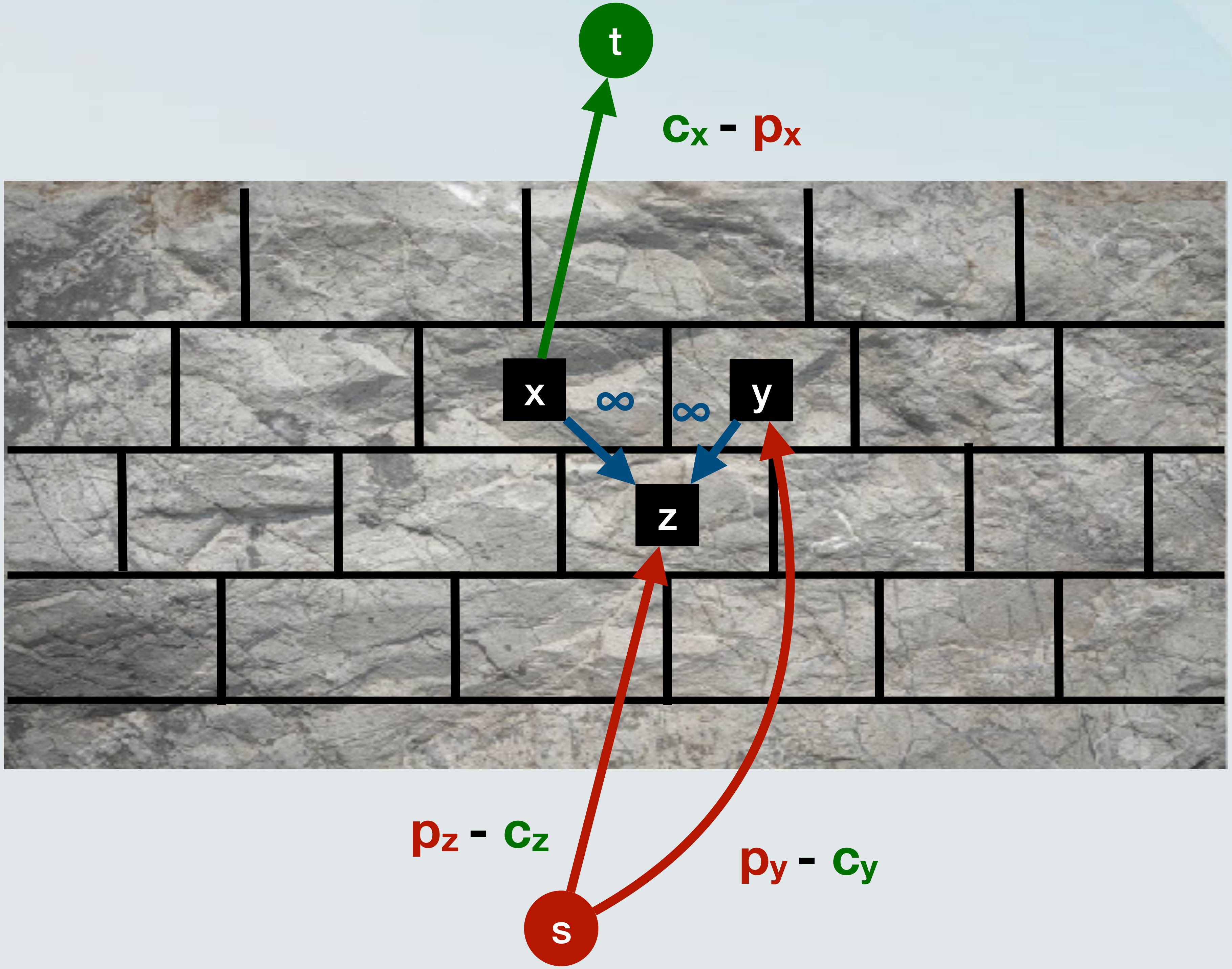
From pits to flows



From pits to flows



From pits to cuts



From pits to cuts

From pits to cuts

- Consider an (S, T) cut C.

From pits to cuts

- Consider an (S, T) cut C.
- If C is minimum, S or T must contain all nodes that are connected with an *infinite capacity* edge.

From pits to cuts

- Consider an (S, T) cut C.
- If C is minimum, S or T must contain all nodes that are connected with an *infinite capacity* edge.
    - If it contains z, it must contain x and y.

From pits to cuts

- Consider an (S, T) cut C.
- If C is minimum, S or T must contain all nodes that are connected with an *infinite capacity* edge.
    - If it contains z, it must contain x and y.
- We will mine S - {s}.

From pits to cuts

- Consider an (S, T) cut C.
- If C is minimum, S or T must contain all nodes that are connected with an *infinite capacity* edge.
  - If it contains z, it must contain x and y.
- We will mine S - {s}.
  - Feasibility guaranteed by the above fact.

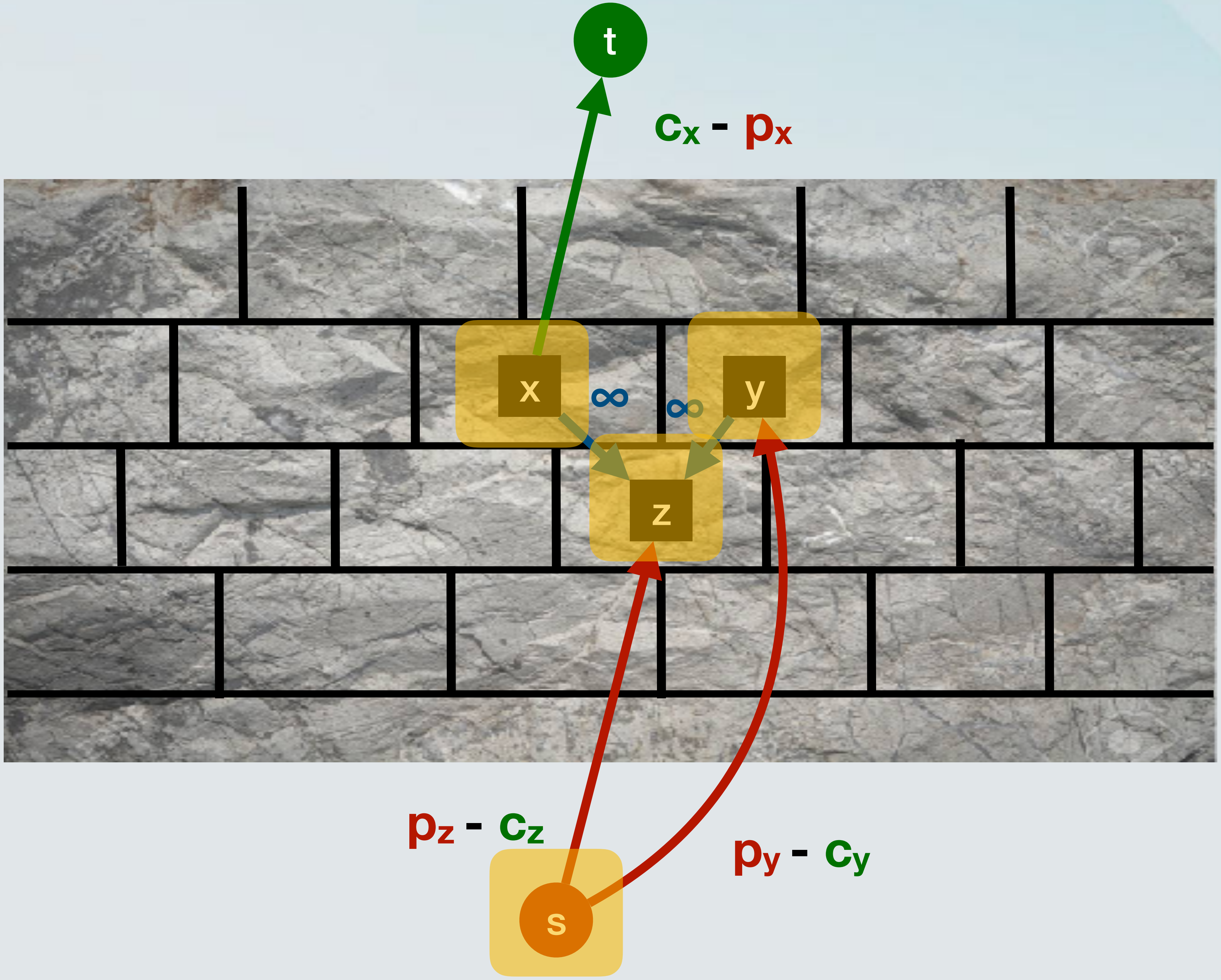
From pits to cuts

- Consider an (S, T) cut C.
- If C is minimum, S or T must contain all nodes that are connected with an *infinite capacity* edge.
    - If it contains z, it must contain x and y.
- We will mine S - {s}.
    - Feasibility guaranteed by the above fact.
    - Optimality?

Optimality of our mining set.

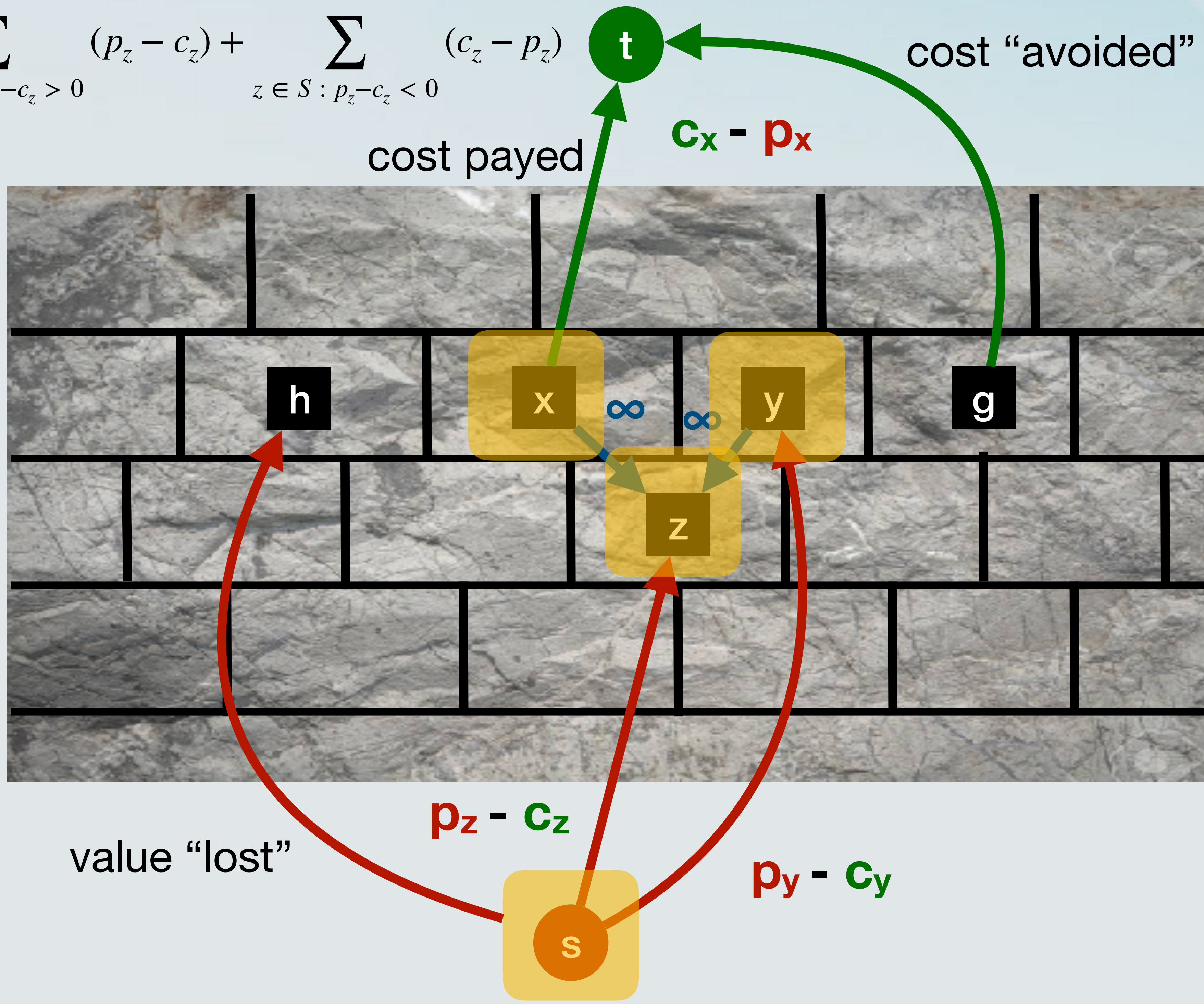
$$c(S,T) = \sum_{z \in T \,:\, p_z - c_z > 0} (p_z - c_z) + \sum_{z \in S \,:\, p_z - c_z < 0} (c_z - p_z)$$

From pits to cuts



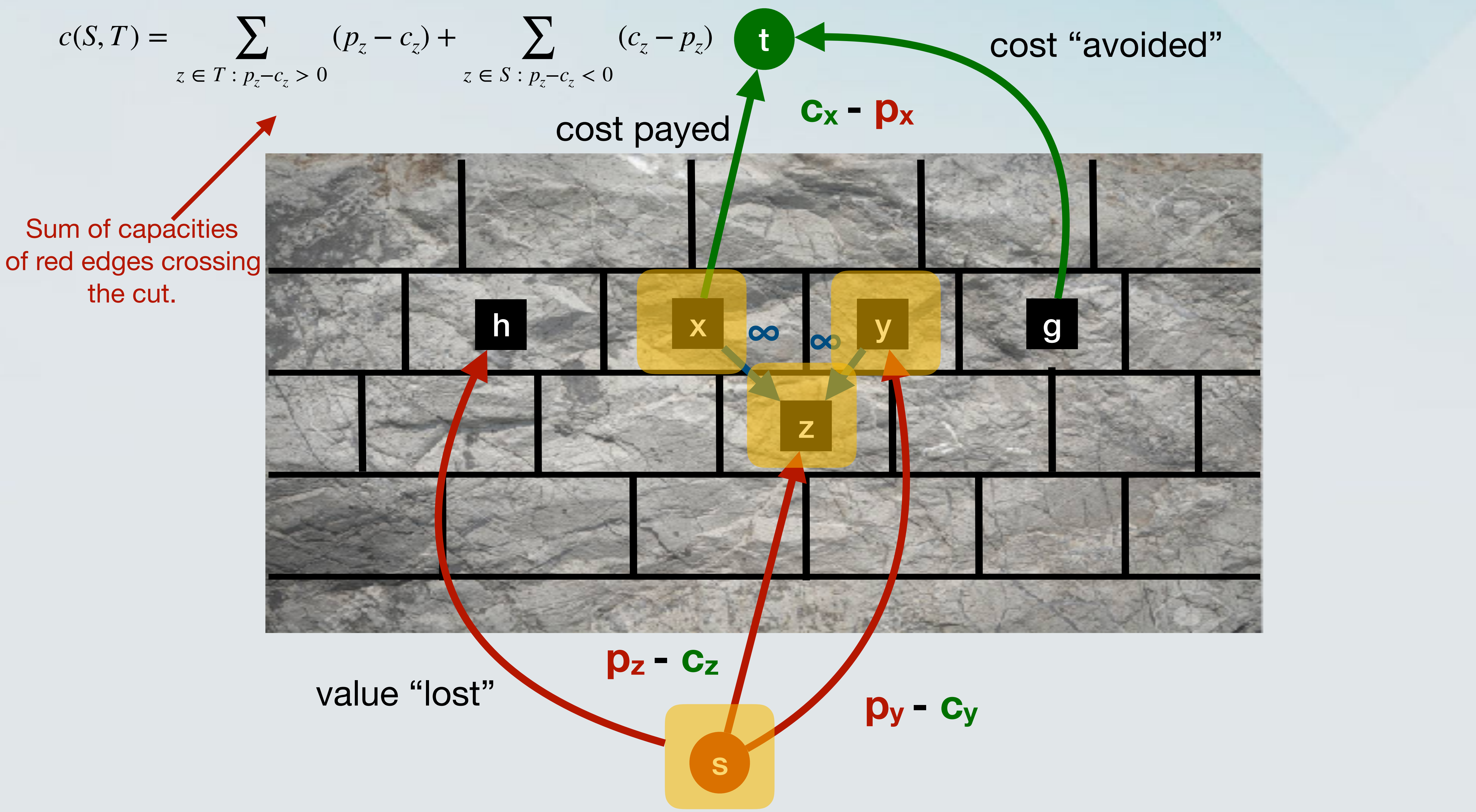
From pits to cuts

$$c(S,T) = \sum_{z \in T \,:\, p_z - c_z > 0} (p_z - c_z) + \sum_{z \in S \,:\, p_z - c_z < 0} (c_z - p_z)$$



From pits to cuts

$$c(S,T) = \sum_{z \in T\,:\,p_z - c_z > 0} (p_z - c_z) + \sum_{z \in S\,:\,p_z - c_z < 0} (c_z - p_z)$$

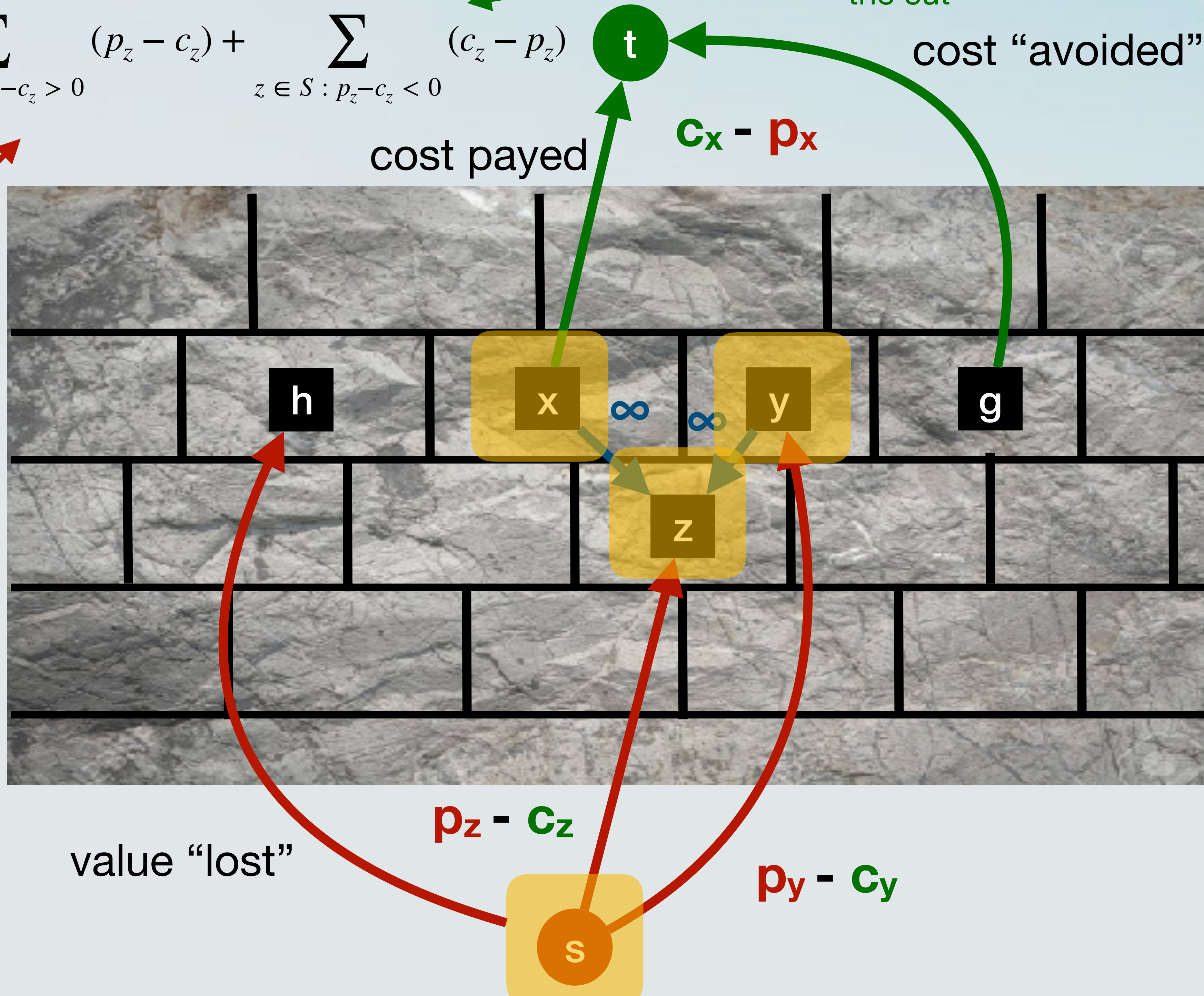


From pits to cuts

$$c(S,T) = \sum_{z \in T \,:\, p_z - c_z > 0} (p_z - c_z) + \sum_{z \in S \,:\, p_z - c_z < 0} (c_z - p_z)$$

Sum of capacities of green edges crossing the cut

Sum of capacities of red edges crossing the cut.



Optimality of our mining set.

$$c(S,T) = \sum_{z \in T \,:\, p_z - c_z > 0} (p_z - c_z) \;+\; \sum_{z \in S \,:\, p_z - c_z < 0} (c_z - p_z)$$

Optimality of our mining set.

$$c(S,T) = \sum_{z \in T \,:\, p_z - c_z > 0} (p_z - c_z) \; + \sum_{z \in S \,:\, p_z - c_z < 0} (c_z - p_z)$$

$$c(S,T) = \sum_{z \in T \,:\, p_z - c_z > 0} (p_z - c_z) \; - \sum_{z \in S \,:\, p_z - c_z < 0} (p_z - c_z)$$

Optimality of our mining set.

$$c(S,T) = \sum_{z \in T \,:\, p_z - c_z > 0} (p_z - c_z) \; + \sum_{z \in S \,:\, p_z - c_z < 0} (c_z - p_z)$$

$$c(S,T) = \sum_{z \in T \,:\, p_z - c_z > 0} (p_z - c_z) \; - \sum_{z \in S \,:\, p_z - c_z < 0} (p_z - c_z)$$

Add and subtract this: $$c(S,T) = \sum_{z \in S \,:\, p_z - c_z > 0} (p_z - c_z)$$

Optimality of our mining set.

$$c(S,T) = \sum_{z \in T\,:\,p_z - c_z > 0} (p_z - c_z) \;+\; \sum_{z \in S\,:\,p_z - c_z < 0} (c_z - p_z)$$

$$c(S,T) = \sum_{z \in T\,:\,p_z - c_z > 0} (p_z - c_z) \;-\; \sum_{z \in S\,:\,p_z - c_z < 0} (p_z - c_z)$$

Add and subtract this:

$$c(S,T) = \sum_{z \in S\,:\,p_z - c_z > 0} (p_z - c_z)$$

$$c(S,T) = \sum_{z \in V\,:\,p_z - c_z > 0} (p_z - c_z) \;-\; \sum_{z \in S} (p_z - c_z)$$

Optimality of our mining set.

$$c(S,T) = \sum_{z \in T \,:\, p_z - c_z > 0} (p_z - c_z) \;+ \sum_{z \in S \,:\, p_z - c_z < 0} (c_z - p_z)$$

$$c(S,T) = \sum_{z \in T \,:\, p_z - c_z > 0} (p_z - c_z) \;- \sum_{z \in S \,:\, p_z - c_z < 0} (p_z - c_z)$$

Add and subtract this: $$c(S,T) = \sum_{z \in S \,:\, p_z - c_z > 0} (p_z - c_z)$$

$$c(S,T) = \sum_{z \in V \,:\, p_z - c_z > 0} (p_z - c_z) - \sum_{z \in S} (p_z - c_z)$$

constant

Optimality of our mining set.

$$c(S,T) = \sum_{z \in T \,:\, p_z - c_z > 0} (p_z - c_z) + \sum_{z \in S \,:\, p_z - c_z < 0} (c_z - p_z)$$

$$c(S,T) = \sum_{z \in T \,:\, p_z - c_z > 0} (p_z - c_z) - \sum_{z \in S \,:\, p_z - c_z < 0} (p_z - c_z)$$

Add and subtract this: $$c(S,T) = \sum_{z \in S \,:\, p_z - c_z > 0} (p_z - c_z)$$

$$c(S,T) = \underbrace{\sum_{z \in V \,:\, p_z - c_z > 0} (p_z - c_z)}_{\text{constant}} - \underbrace{\sum_{z \in S} (p_z - c_z)}_{\text{Mining profit}}$$

Open-pit mining - Summarising

- Construct the flow network.
- Run Ford-Fulkerson to find a maximum flow.
- Find a minimum cut using the final residual graph.
- Mine the blocks in the S part of the cut.