Network flow optimization

15.093: Optimization

Dimitris Bertsimas
Alexandre Jacquillat

Sloan School of Management
Massachusetts Institute of Technology

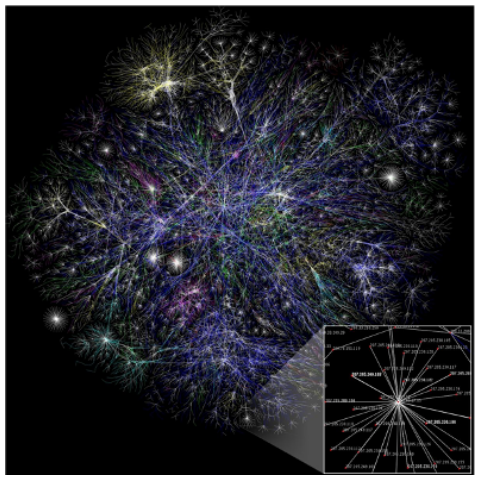
Network industries

Transportation networks

Electricity networks

Telecommunications networks

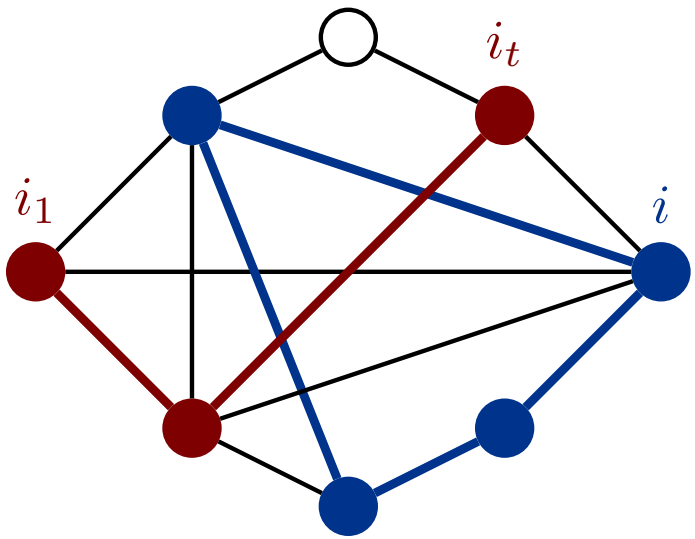
Social networks



Graph terminology

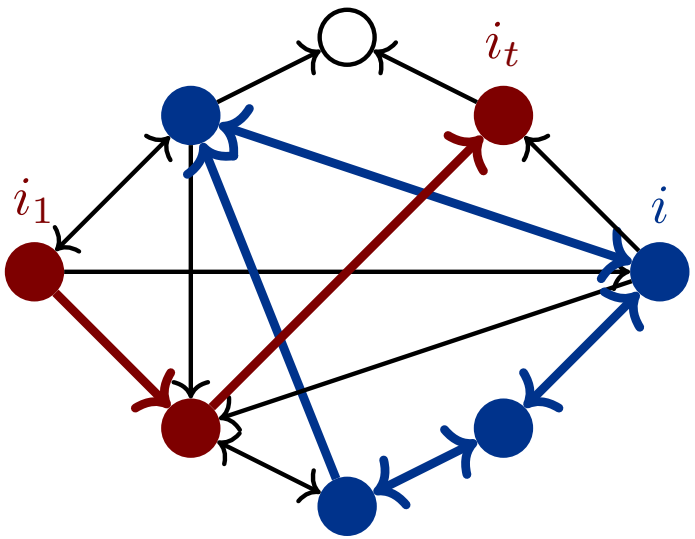
Undirected graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$

- $\mathcal{N}$: set of nodes $i = 1, \cdots, n$
- $\mathcal{E}$: set of undirected edges $\{i, j\}$
- **Path** from i_1 to i_t
- **Cycle**: path from i to i
- Graph is connected if there exists a path between any two nodes



Directed graph $\mathcal{G} = (\mathcal{N}, \mathcal{A})$

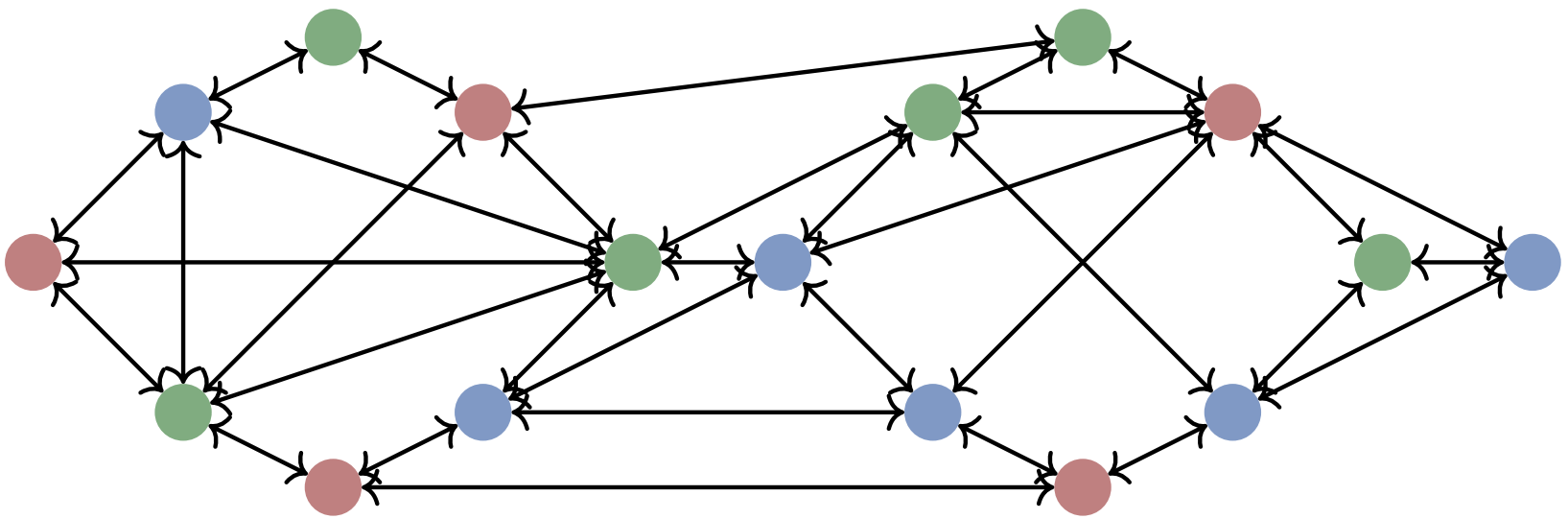
- $\mathcal{N}$: set of nodes $i = 1, \cdots, n$
- $\mathcal{A}$: set of directed arcs (i, j)
- **Path** from i_1 to i_t
- **Cycle**: path from i to i
- Graph is connected if associated undirected graph is connected



The minimum cost flow problem

Problem statement

- Goal: least-cost movements of a commodity through a network
- Directed graph $\mathcal{G} = (\mathcal{N}, \mathcal{A})$: nodes $i, j \in \mathcal{N}$ and arcs $a = (i, j) \in \mathcal{A}$
  - The commodity is available in supply nodes: $b_i > 0$
  - The commodity can be shipped through transshipment nodes: $b_i = 0$
  - The commodity is requested in demand nodes: $b_i < 0$
- Movements on arc $a = (i, j) \in \mathcal{A}$ incur a unit cost c_{ij}
- Movements on arc $a = (i, j) \in \mathcal{A}$ are limited by capacity u_{ij}

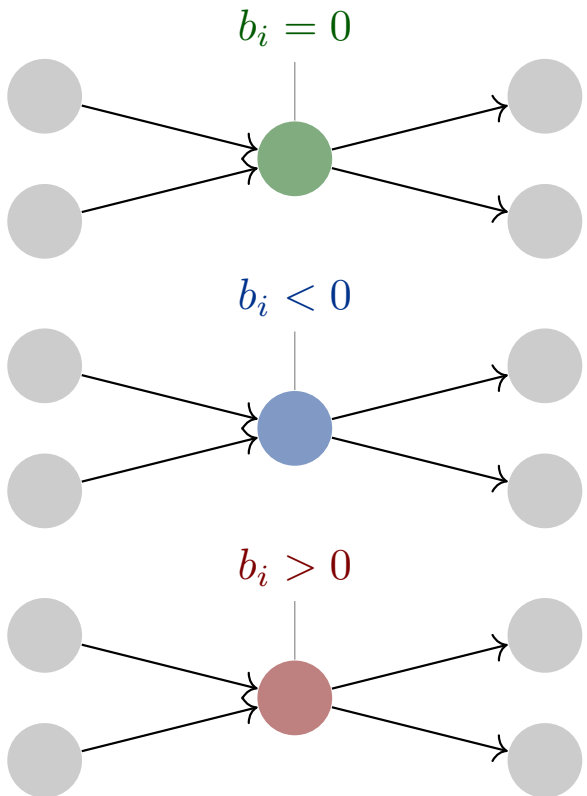


Flow decisions and flow balance

- Assumption of a balanced network: $\sum_{i \in \mathcal{N}} b_i = 0$
- Decision variable:

$$x_{ij} : \text{ amount of flow through arc } (i,j), \text{ for } i,j \in \mathcal{N}$$

- Transshipment node: net supply is zero
  - $\rightarrow$ What comes in comes out
- Demand node: net supply is negative
  - $\rightarrow$ What comes in is equal to what comes out plus the demand at the node $(-b_i)$
- Supply node: net supply is positive
  - $\rightarrow$ What comes in plus the supply at the node (b_i) is equal to what comes out



Problem formulation

- Objective: seeking the minimum-cost flow through the network
- Demand and supply requirements via flow balance constraints
  - Supply node: what comes in + supply = what comes out
  - Demand node: what comes in = demand + what comes out
  - Transshipment node: what comes in = what comes out
- Capacity constraints: flow less than maximum flow allowed

Formulation (Minimum cost network flow)

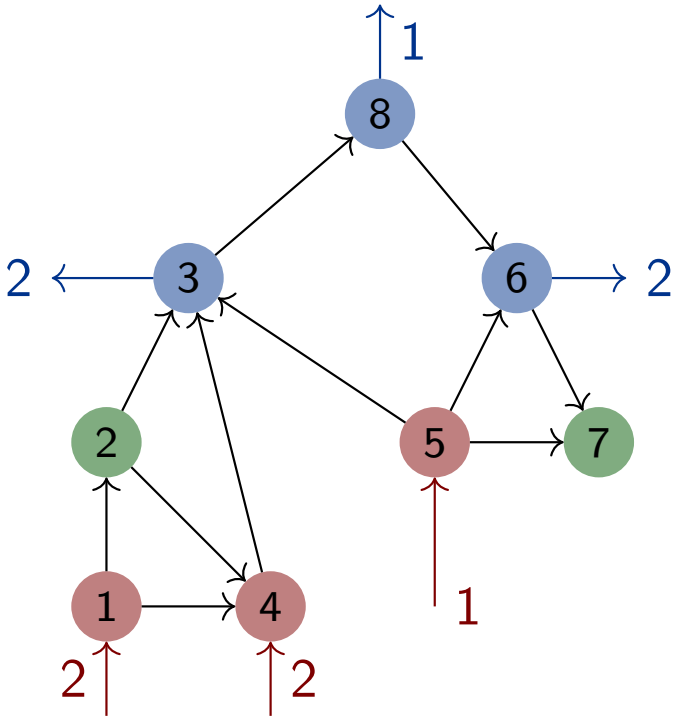
$$\min \quad \sum_{(i,j)\in\mathcal{A}} c_{ij}x_{ij}$$

$$\text{s.t.} \quad \sum_{j:(i,j)\in\mathcal{A}} x_{ij} - \sum_{j:(j,i)\in\mathcal{A}} x_{ji} = b_i, \qquad \forall i \in \mathcal{N}$$

$$x_{ij} \leq u_{ij}, \qquad \forall (i,j) \in \mathcal{A}$$

$$x_{ij} \geq 0, \qquad \forall (i,j) \in \mathcal{A}$$

Example



- Flow balance constraints

$$
\begin{aligned}
x_{12} + x_{14} &= 2 \\
x_{23} + x_{24} - x_{12} &= 0 \\
x_{38} - x_{23} - x_{43} - x_{53} &= -2 \\
x_{43} - x_{14} - x_{24} &= 2 \\
x_{53} + x_{56} + x_{57} &= 1 \\
x_{67} - x_{56} - x_{86} &= -2 \\
- x_{57} - x_{67} &= 0 \\
x_{86} - x_{38} &= -1
\end{aligned}
$$

$$
\boldsymbol{b} = \begin{array}{cr} 1 & +2 \\ 2 & 0 \\ 3 & -2 \\ 4 & +2 \\ 5 & +1 \\ 6 & -2 \\ 7 & 0 \\ 8 & -1 \end{array}
$$

$$
\boldsymbol{A} = \begin{array}{c|cccccccccc}
 & (1,2) & (1,4) & (2,3) & (2,4) & (3,8) & (4,3) & (5,3) & (5,6) & (5,7) & (6,7) & (8,6) \\
1 & +1 & +1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
2 & -1 & 0 & +1 & +1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
3 & 0 & 0 & -1 & 0 & +1 & -1 & -1 & 0 & 0 & 0 & 0 \\
4 & 0 & -1 & 0 & -1 & 0 & +1 & 0 & 0 & 0 & 0 & 0 \\
5 & 0 & 0 & 0 & 0 & 0 & 0 & +1 & +1 & +1 & 0 & 0 \\
6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & +1 & -1 \\
7 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & 0 \\
8 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & +1
\end{array}
$$

Generalized flow balance constraints

- Recall that we have thus far considered a balanced network: $\sum_{i \in \mathcal{N}} b_i = 0$

  $\rightarrow$ Flow balance constraints: outgoing flow $=$ incoming flow $+$ supply

$$\sum_{j:(i,j) \in \mathcal{A}} x_{ij} - \sum_{j:(j,i) \in \mathcal{A}} x_{ji} = b_i, \forall i \in \mathcal{N}$$

- What about excess supply $\sum_{i \in \mathcal{N}} b_i > 0$?

  $\rightarrow$ Flow balance constraints: outgoing flow $\leq$ incoming flow $+$ supply

$$\sum_{j:(i,j) \in \mathcal{A}} x_{ij} - \sum_{j:(j,i) \in \mathcal{A}} x_{ji} \leq b_i, \forall i \in \mathcal{N}$$

- What about excess demand $\sum_{i \in \mathcal{N}} b_i < 0$?

  $\rightarrow$ Flow balance constraints: outgoing flow $+$ demand $\geq$ incoming flow

$$\sum_{j:(i,j) \in \mathcal{A}} x_{ij} - \sum_{j:(j,i) \in \mathcal{A}} x_{ji} \geq b_i, \forall i \in \mathcal{N}$$

Applications

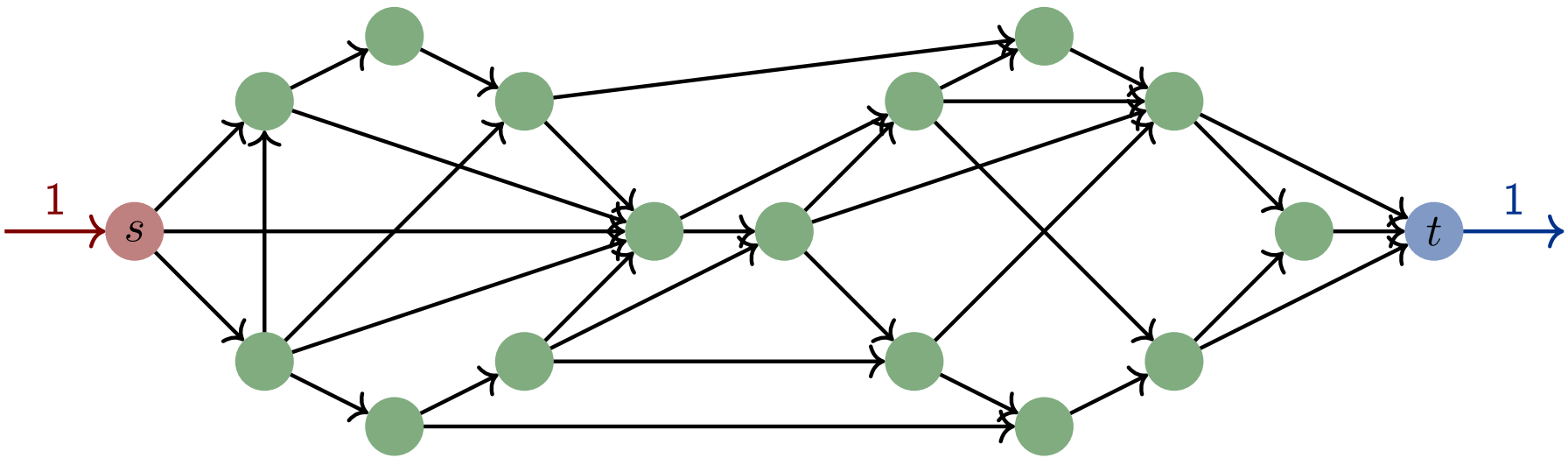
The shortest path problem

- You are traveling from a source (e.g., home) to a sink (e.g., work)
- What is the best route?

→ Core of mapping software: finding a path of minimum length, minimum time, minimum cost, maximum reliability, etc.

→ Network flow problem

- One source node with a supply of 1
- One sink node with a demand of 1
- All intermediate nodes with a supply of 0



Shortest path formulation

- Decision variables:

$$x_{ij} = \begin{cases} 1 & \text{if the path takes arc } (i,j) \\ 0 & \text{otherwise} \end{cases}$$

- Mathematical formulation:
  - Minimizing the total cost of the path
  - Ensuring that each node has exactly one predecessor and one successor

Formulation (Shortest path problem)

$$\min \quad \sum_{(i,j)\in\mathcal{A}} c_{ij}x_{ij}$$

$$\text{s.t.} \quad \sum_{j:(i,j)\in\mathcal{A}} x_{ij} - \sum_{j:(j,i)\in\mathcal{A}} x_{ji} = \begin{cases} 0 & \forall i \neq s,t \\ 1 & \text{for } i = s \\ -1 & \text{for } i = t \end{cases}$$

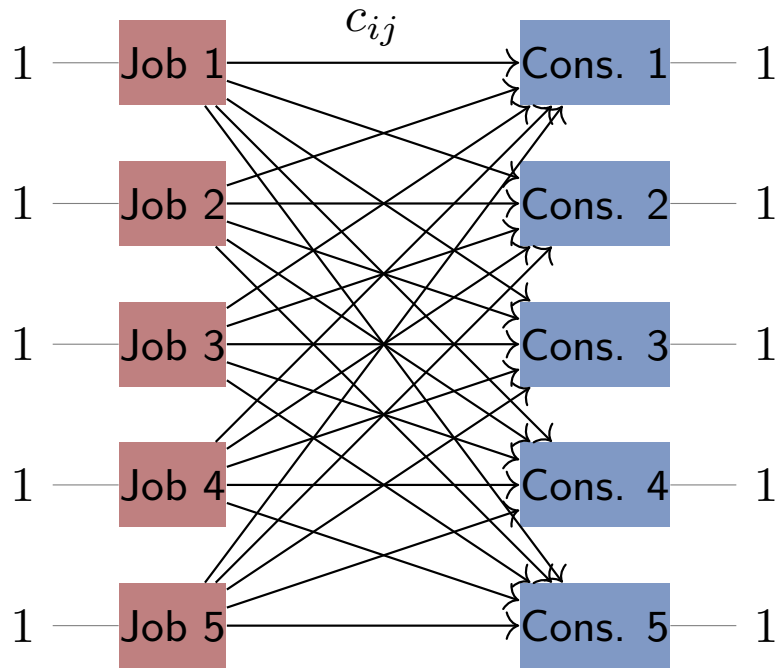
$$x_{ij} \geq 0, \ \forall (i,j) \in \mathcal{A}$$

The assignment problem

- A staffing manager has n consultants available to perform n jobs
- Consultant i incurs cost c_{ij} for job j
- Problem: how to assign jobs to consultants to minimize cost?

$\rightarrow$ Network flow problem

- One node per job with a supply of 1
- One node per consultant with a demand of 1



Assignment formulation

- Decision variables:

$$x_{ij} = \begin{cases} 1 & \text{if consultant } i \text{ is assigned to job } j \\ 0 & \text{otherwise} \end{cases}$$

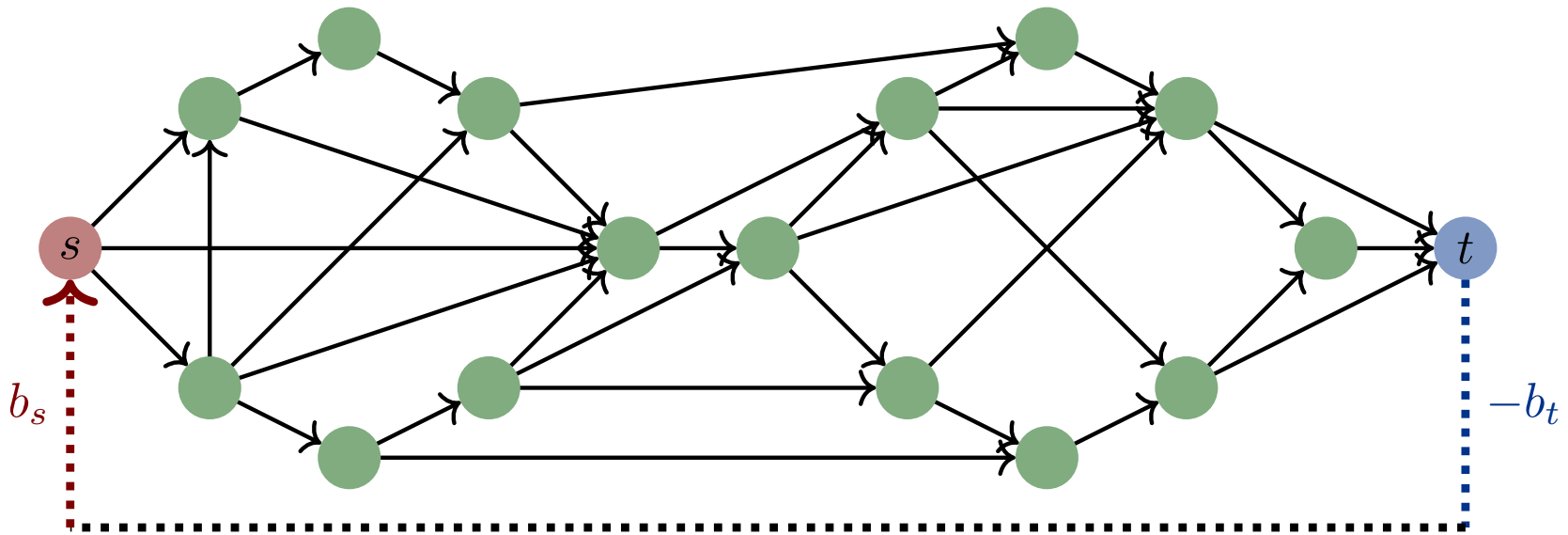
- Mathematical formulation:
  - Minimizing the total cost of the assignment
  - Ensuring that each consultant is assigned exactly one job
  - Ensuring that each consultant is covered by exactly one consultant

Formulation (Assignment problem)

$$\begin{aligned} \min_{\boldsymbol{x} \geq \mathbf{0}} \quad & \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij} \\ \text{s.t.} \quad & \sum_{i=1}^{n} x_{ij} = 1, \quad \forall j = 1, \cdots, n \\ & \sum_{j=1}^{n} x_{ij} = 1, \quad \forall i = 1, \cdots, n \end{aligned}$$

The maximum flow problem

- Determine the maximum flow that can be sent from a source node to a sink node in a capacitated network.
  - steady-state flow of petroleum products in a pipeline network
  - steady-state flow of cars in a road network
  - steady-state flow of information in a telecommunication network
  - steady-state flow of electricity in a power network
- Transformation by adding a virtual link from the sink to the source, with total flow to be maximized on the virtual link



Maximum flow formulation

Formulation

$$
\begin{aligned}
\max \quad & b_s \\
\textit{s.t.} \quad & \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \\
& b_t = -b_s \\
& b_i = 0, \ \forall i \neq s, t \\
& \boldsymbol{0} \leq \boldsymbol{x} \leq \boldsymbol{u}
\end{aligned}
$$

Formulation

$$
\begin{aligned}
\max \quad & b_s \\
\textit{s.t.} \quad & \sum_{j:(i,j)\in\mathcal{A}} x_{ij} - \sum_{j:(j,i)\in\mathcal{A}} x_{ji} = \begin{cases} b_s & \textit{if } i = s \\ -b_s & \textit{if } i = t \\ 0 & \textit{for all } i \neq s, t \end{cases} \\
& 0 \leq x_{ij} \leq u_{ij}, \ \forall (i,j) \in \mathcal{A}
\end{aligned}
$$

- The maximum flow problem optimizes the "supply" at the source node (called **value** of the flow)
- The maximum flow problem can be transformed into a network flow: virtual link from sink to source, with cost -1 and infinite capacity

$$\min \sum_{(i,j)\in\mathcal{A}} c_{ij}x_{ij} \iff \max \ x_{ts} \iff \max \ b_s$$

$\rightarrow$ Maximize flow through a capacitated network, vs. minimizing flow cost

Network simplex algorithm

Tree in network

Definition

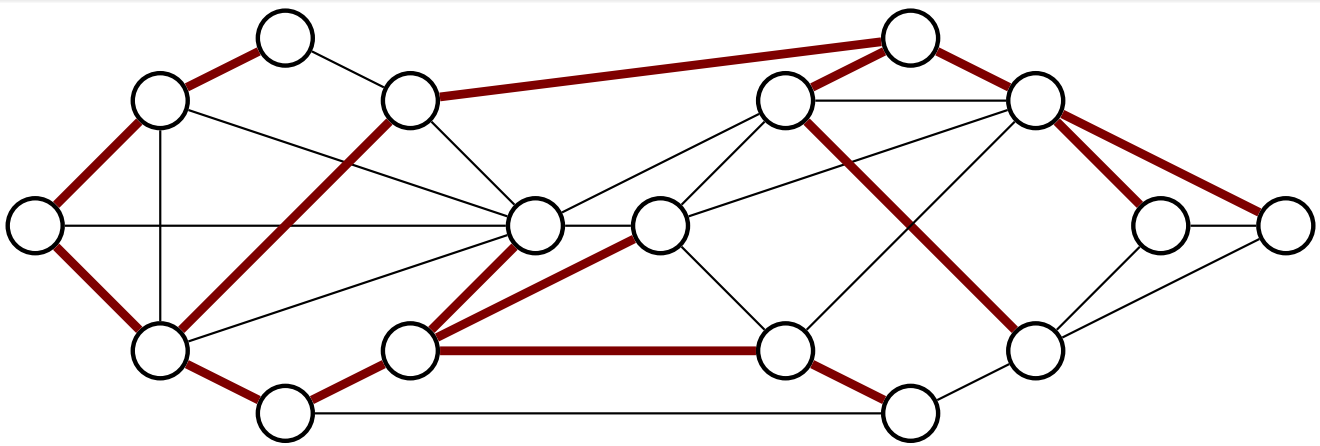
Tree: connected graph with no cycle.

Proposition

$\mathcal{G}$ is a tree if and only if it is connected and # arcs = # nodes -1.

Definition

Spanning tree of $\mathcal{G}$: subgraph that is a tree and contains all nodes of $\mathcal{G}$.

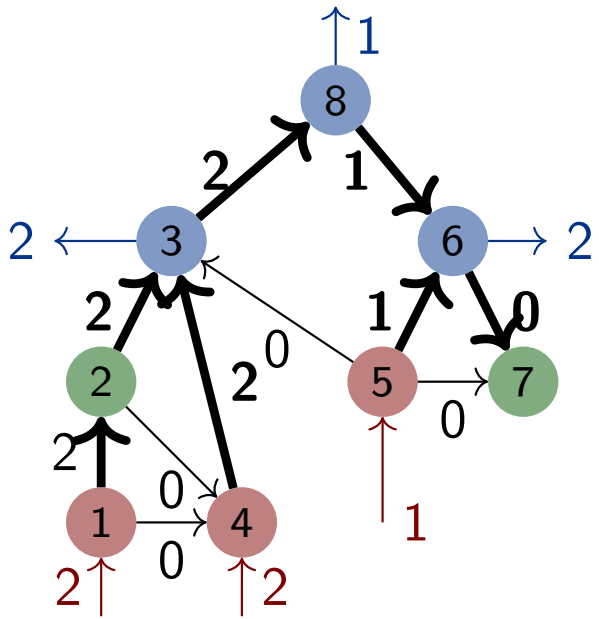


Tree in network

Definition

Tree solution x: there is a spanning tree with a zero flow for non-tree arcs.

$$(\mathcal{N}, \mathcal{T}) \text{ is a spanning tree and } x_{ij} = 0, \ \forall (i,j) \notin \mathcal{T}$$

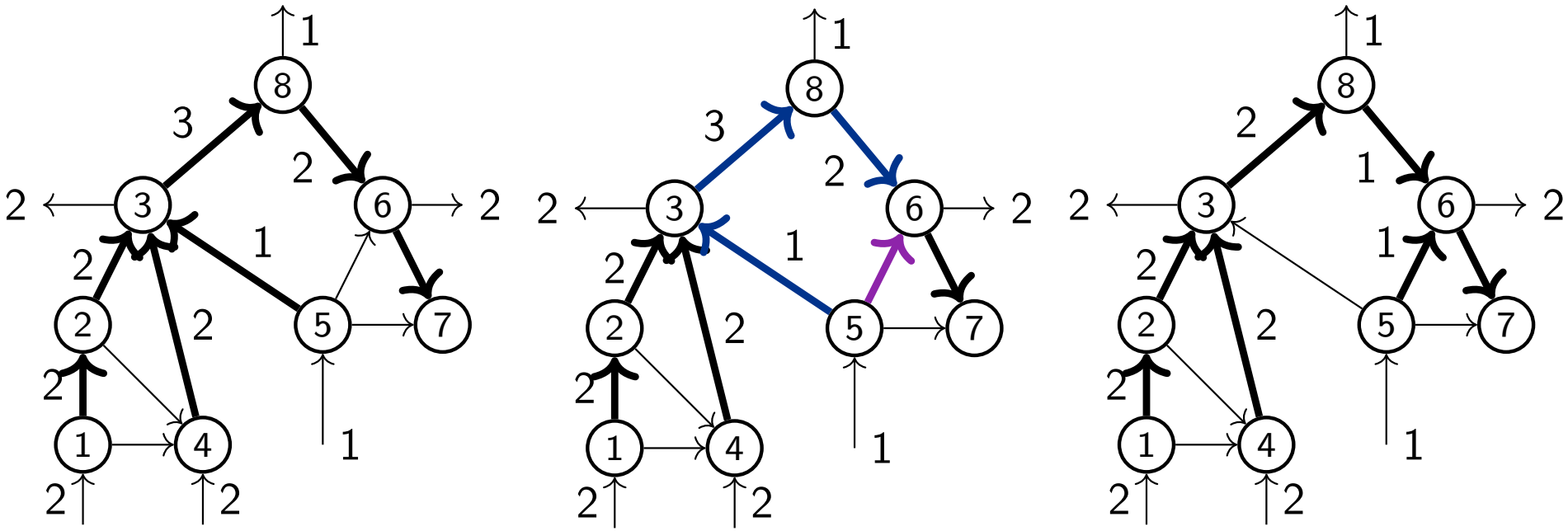


Theorem

A flow vector x is a basic feasible solution if and only if it is a tree solution

Change of basis

Tree solution | **x_{56} enters the basis** | **New tree solution**



- Bringing a new variable in the basis creates a cycle

→ Network simplex: push flow into the cycle until a variable exits

→ The network simplex retrieves a tree solution, hence a BFS

Network simplex operations

- Identify the cycle created by the new basic variable
  - F: set of forward arcs (same direction as new variable)
  - B: set of backward arcs (opposite direction)

$\rightarrow$ Push as much flow into the cycle as possible, until a variable exits

$$x_{kl} \leftarrow \begin{cases} x_{kl} + \theta & \text{if } (k,l) \in F \\ x_{kl} - \theta & \text{if } (k,l) \in B \\ x_{kl} & \text{otherwise} \end{cases} \qquad \Longrightarrow \theta^* = \min_{(k,l) \in B} x_{kl}$$

- The variable (i,j) enters the basis if the cost change is negative:

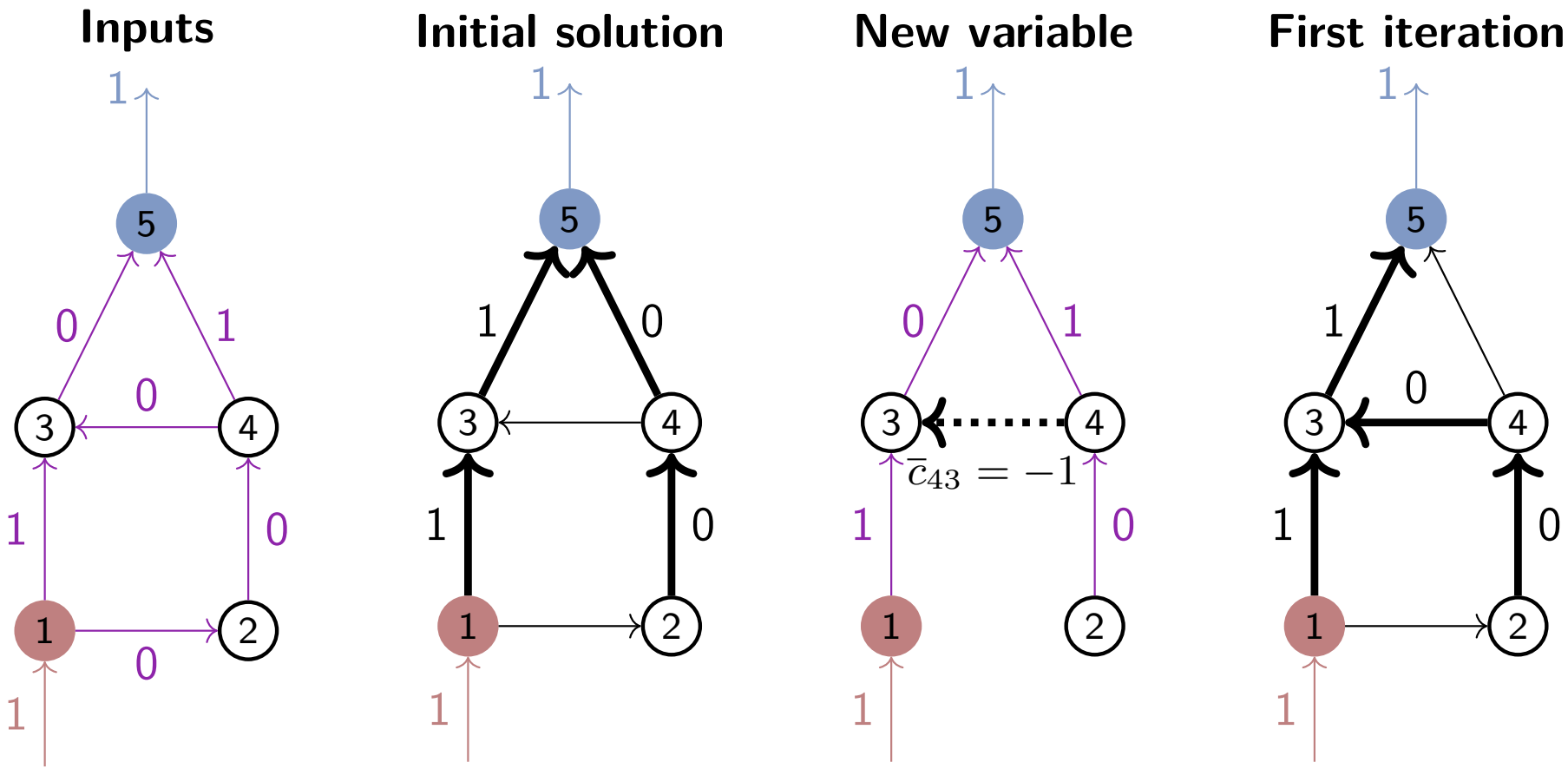
$$\theta^* \cdot \underbrace{\left(\sum_{(k,l) \in F} c_{kl} - \sum_{(k,l) \in B} c_{kl} \right)}_{\overline{c}_{ij}} < 0$$

- To go further: we can compute all reduced costs in $\mathcal{O}(|\mathcal{N}|)$ operations by computing the **potential** of each node

The network simplex algorithm in action: iteration 1

- Identification of variable x_{43} with negative reduced cost
- No flow gets pushed through the cycle because $x_{45} = 0$ in cycle

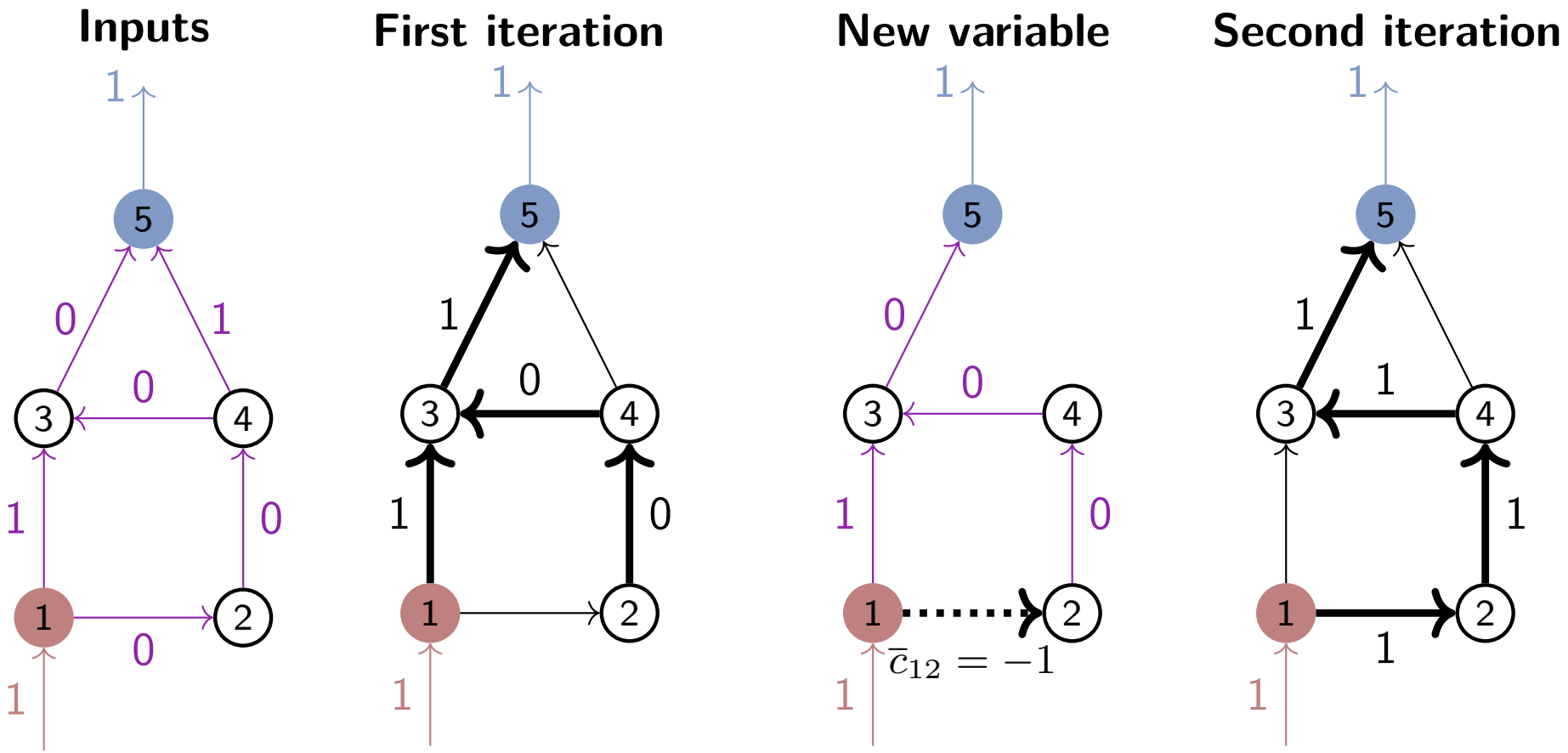
$\rightarrow$ Move from one degenerate solution to another, with no change in cost



The network simplex algorithm in action: iteration 2

- Identification of variable x_{12} with negative reduced cost
- One unit of flow gets pushed through the cycle, until $x_{13} = 0$

$\rightarrow$ New solution, reduction in cost, and optimal solution



Summary of network simplex algorithm

- The network simplex algorithm proceeds as standard simplex
  - Move from one basic feasible solution to an adjacent one
  - `STOP` when all reduced costs become non-negative
- The structure of the network flow problems enables a fast implementation of the network simplex algorithm
  - Direct computation of reduced costs and new basis

  $\rightarrow$ Use network structure rather than algebra (e.g., no need for $\boldsymbol{B}^{-1}$)

  $\rightarrow$ $\mathcal{O}(m)$ operations at each iteration vs. $\mathcal{O}(mn)$ for standard simplex

$\rightarrow$ Significant performance improvements as compared to standard simplex [averaged over 5 random instances with 10,000 nodes and 25,000 arcs]

Algorithm	CPU time (sec)	# iterations
Standard Simplex	334.59	42,759
Network Simplex	7.37	23,306
Ratio	2.2 %	54 %

Integrality property

- What happens if we start with an integral solution?

$$x_{ij} \in \mathbb{Z},\ \forall (i,j) \in \mathcal{A}$$

- At the next iteration, the solution remains integral!

$$\theta^* = \min_{(k,l)\in B} x_{kl} \in \mathbb{Z}$$

$$\Longrightarrow x_{kl} \in \{x_{kl} + \theta^*, x_{kl} - \theta^*, x_{kl}\} \in \mathbb{Z}$$

Theorem (integral solution)

The solution of a network flow problem is integral as long as all supply quantities, demand quantities and arc capacities are integral

- This is a unique property: most linear optimization problem do not yield integer solution even with integral coefficients, e.g.:

$$\begin{cases} x + y = 1 \\ x - y = 0 \end{cases} \Longrightarrow x = y = 0.5$$

Conclusion

Summary

Takeaway

Network flows are a pervasive class of optimization problems in practice.

Takeaway

Minimum cost flow problem: minimizing the transshipment cost from source nodes to demand nodes subject to flow balance and arc capacities.

Takeaway

Sub-classes of problems with particular structure: shortest path problem, assignment problem, maximum flow problem.

Takeaway

Fast and scalable algorithms: network simplex, other tailored algorithms.

Takeaway

Network flow solutions satisfy integrality properties.