ORF 522

Linear Programming and Convex Analysis

Network Flows

Marco Cuturi

Reminder

- Graphs:
  - $\mathcal{N}$ set of nodes
  - directed edges $\mathcal{E}$ or arcs $\mathcal{A}$
- Networks:
  - A graph, usually directed,
  - with contextual information.
- Tree: a connected undirected graph with no cycles.
  - Connected undirected graph with $\#\mathcal{N} - 1$ edges.
  - Trees have at least one leaf, $i.e.$a node of degree 1.
  - Unique path between two nodes i and j.

Network Flows

Mathematical Formulation

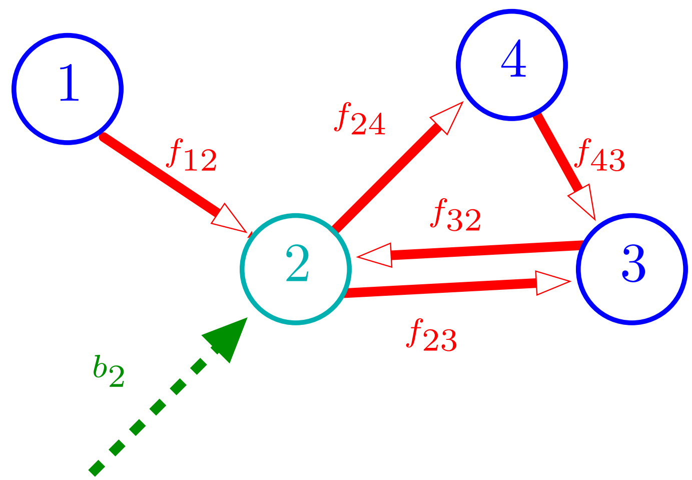
A **network** is a **directed graph** $\mathcal{G} = (\mathcal{N}, \mathcal{A})$ with side information:

- for a in $\mathcal{A}$, or equivalently $(i, j) \in \mathcal{A}$, a nonnegative f_a or $f_{(i,j)}$ and usually written f_{ij} quantifies a **flow** between nodes i and j.
- For each node $i \in \mathcal{N}$ b_i is a **supply** to that node from the **exterior**.
  - if $b_i > 0$ node i is usually called a **source**.
  - if $b_i < 0$ node i is usually called a **sink**.
- Each flow can be **capacitated** that is restricted to be less than $u_{i,j}$.
- When $u_{i,j} = \infty$ the flow is **uncapacitated**.
- Each arc might have a **cost** per unit of flow associated, c_{ij}.

Flow Equations *constraints*

Natural flow equations imply that

$$b_i + \sum_{j \in I(i)} f_{ji} = \sum_{j \in O(i)} f_{ij} \tag{1}$$
$$0 \leq f_{ij} \leq u_{ij}$$



in this case,

$$\boldsymbol{b_2} + f_{1\mathbf{2}} + f_{3\mathbf{2}} = f_{\mathbf{2}4} + f_{\mathbf{2}3}$$
$$0 \leq f_{12}, f_{24}, f_{32}, f_{23} \leq \ldots$$

Flow Equations *constraints*

- More terminology: any vector f with indexed by $\mathcal{E}$ is a flow.
- A flow is **feasible** if it satisfies the **linear** equations (1)
- Note that if we sum up Equation (1)

$$\sum_{i\in\mathcal{N}} b_i = \sum_{i\in\mathcal{N}} \left(\sum_{j\in I(i)} f_{ji} - \sum_{j\in O(i)} f_{ij} \right)$$

$$\sum_{i\in\mathcal{N}} b_i = \sum_{a\in\mathcal{A}} f_a - f_a = 0$$

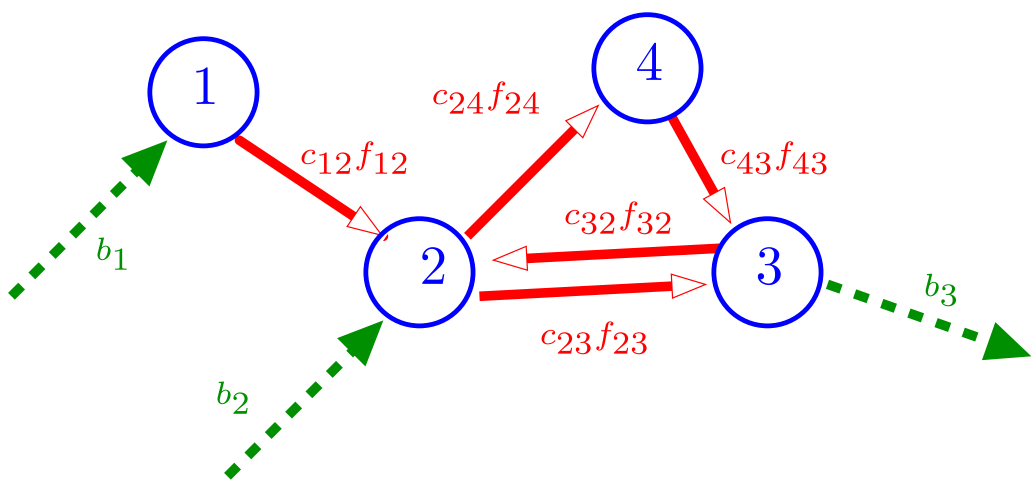
- "*what's taken from the environment goes back to the environment*"
- or: nodes cannot store flows.

Flow Equations *objectives*

- Most network flow problems deal with the minimization of

$$\sum_{(i,j)\in\mathcal{A}} c_{ij} f_{ij}$$

- which is, again, linear in f.



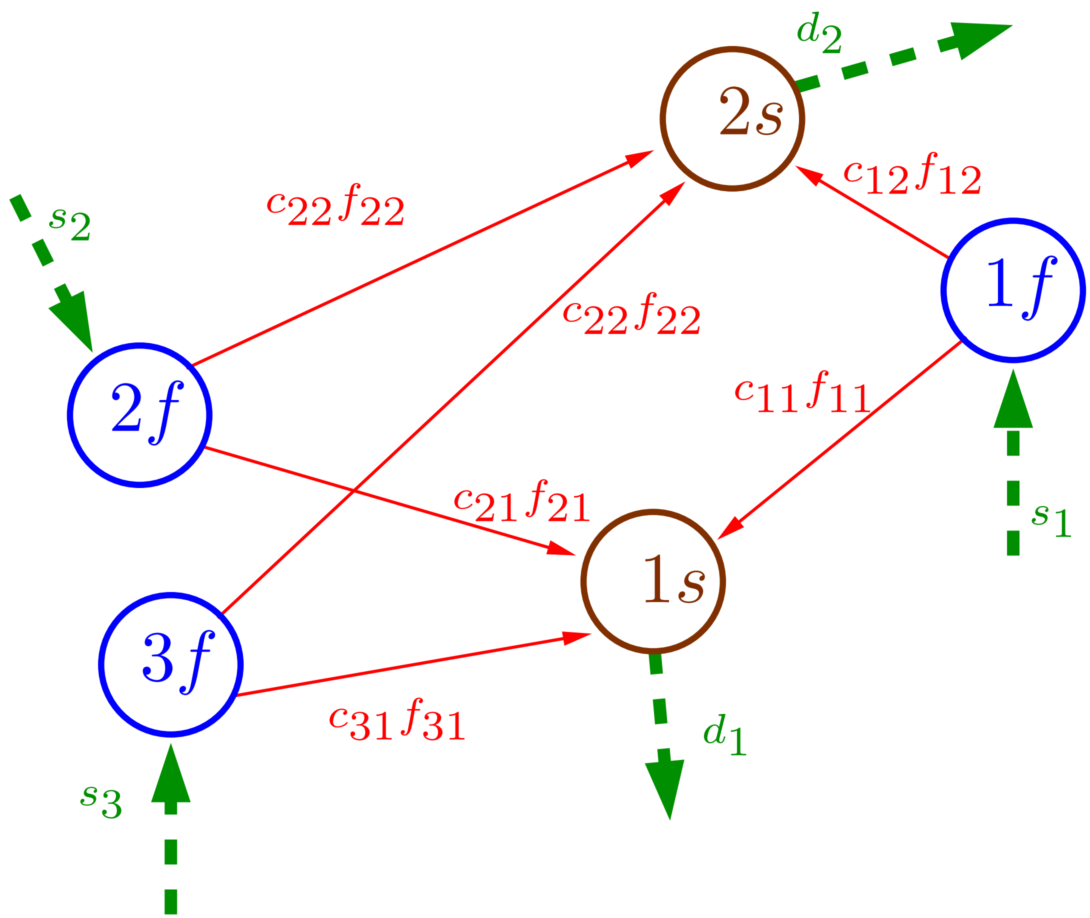
Examples of Network Flow Problems

The Transportation Problem

- Holes and piles of Dirt analogy.
- Old problem, first formulated by Monge in 1781 and Kantorovich in the 30's.
- Suppose there are m **factories** and n **shops** that produce/sell computer units.
- Each factory i produces annually $s_i \geq 0$ computers and a shop j wants $d_j \geq 0$ of them.
- **Each factory i has an arc directed towards each shop j.**
- We suppose the total supply is equal to the demand, $\sum_{i=1}^m s_i = \sum_{j=1}^n d_j$.
- The transport problem is then

$$
\begin{array}{ll}
\text{minimize} & \sum_{i=1}^m \sum_{j=1}^n c_{ij} f_{ij} \\
\text{subject to} & f_{ij} \geq 0 \\
& \forall i = 1, \ldots, m, \quad s_{\boldsymbol{i}} = \sum_{\boldsymbol{j}=1}^n f_{\boldsymbol{ij}}, \\
& \forall j = 1, \ldots, n, \quad d_{\boldsymbol{j}} = \sum_{\boldsymbol{i}=1}^n f_{\boldsymbol{ij}}.
\end{array}
$$

The transportation Problem



- $1s, 2s$ stand for the shops and $1f, 2f, 3f$ is for factories.
- usually c_{ij} are proportional to distances.

The Assignment Problem

- Special case of the TP:
  - $m = n$, same number of suppliers and consumers.
  - supplies are all equal to 1, demands are all equal to 1.
  - problem is to assign one factory to one shop exactly, with minimal cost.
- Another formulation:
  - n employees.
  - n tasks.
  - c_{ij} measures how much time employee i would spend on task j.
  - as a manager, what is the best way to **assign tasks** that **minimizes the total work time**?

Other Problems

- The shortest-path problem
  - Define the length of a path as the sum of weights of the visited edges.
  - Given two nodes i and j find the shortest path.
  - Show later how this is a LP and related algorithms.
- The maximum flow problem
  - sources and sinks, capacitated edges.
  - Find the optimal distribution to maximize flow
- Minimum spanning tree
  - Again, n agents, set of $\mathcal{E}$ I can choose.
  - Setting up each edge has a cost c_{ij}.
  - Target: connect them all for a cheap price.
  - Find a tree $\mathcal{T}$ with minimum total cost $\sum_{a\in\mathcal{T}} c_a$.

Matrix Formulations

- **Objective**: rewrite NF problems (constaints, objectives) with **lighter** notations.
- **Arcs** (variables) numbered from 1 to n,
- **Nodes** (constraints) numbered from 1 to m.
- **Constraints matrix**: the node-arc incidence matrix $A \in \{-1, 0, 1\}^{m \times d}$:

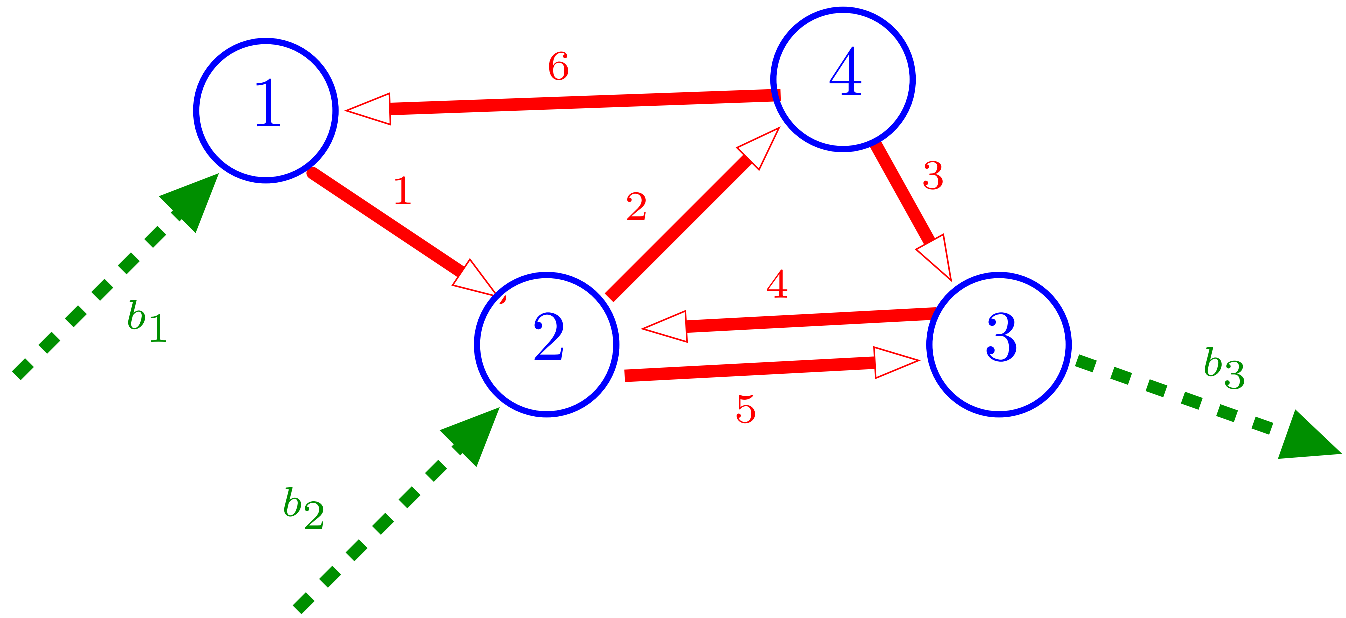
$$A_{ia} = \begin{cases} 1 & \text{if the arc number } a \text{ starts at node } i \\ -1 & \text{if the arc number } a \text{ ends at node } i \\ 0 & \text{otherwise} \end{cases}$$

- Flow conservation:

$$b_i + \sum_{j \in I(i)} f_{ji} = \sum_{j \in O(i)} f_{ij} \quad \Leftrightarrow \quad \sum_{a \text{ start at } i} f_a - \sum_{a \text{ ends at } i} f_a = b_i$$

- Equivalently $A\mathbf{f} = \mathbf{b}$

Matrix Formulations



$$A = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & -1 \\ -1 & 1 & 0 & -1 & 1 & 0 \\ 0 & 0 & -1 & 1 & -1 & 0 \\ 0 & -1 & 1 & 0 & 0 & 1 \end{pmatrix}$$

- Note how all columns have exactly one 1 and -1 and lots of zeros (sparse)
- In particular, rows sum to zero... linear dependence if nothing is changed.

Cost Function

- Directly related to the cost per unit of flow through arc a
- The formulation

$$\sum_{(i,j)\in\mathcal{A}} c_{ij} f_{ij}$$

can be written more simply as

$$\sum_{a\in\mathcal{A}} c_a f_a = \mathbf{c}^T\mathbf{f}$$

where both vectors are n-dimensional.

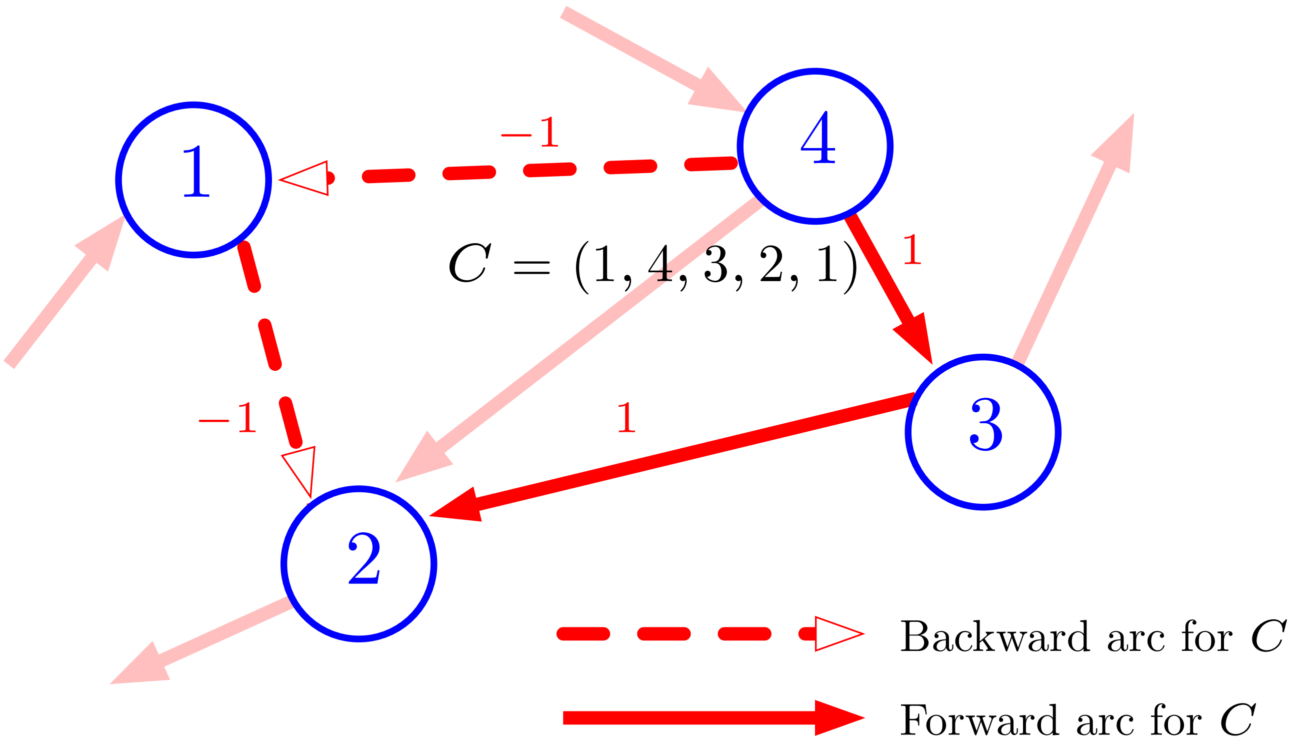
Circulations

- A bit of context: when dealing with $A\mathbf{x} = \mathbf{b}$ and IPM, we introduced directions $\mathbf{d}$ such that $A\mathbf{d} = \mathbf{0}$.
- In the NF context, a flow $\mathbf{f}$ such that $A\mathbf{f} = \mathbf{0}$ is called a **circulation**.
- Suppose there exists a **cycle** $C = i_1, a_1, i_2, \cdots, a_{k-1}, i_k$ in the graph.
- The arcs are either **forward** or **backward** arcs, grouped in subsets F and B.
- The flow vector associated to a cycle C, $\mathbf{h}^C$ defined as

$$h_a^C = \begin{cases} 1 & \text{if arc } a \text{ belongs to } F \\ -1 & \text{if arc } a \text{ belongs to } B \\ 0 & \text{otherwise} \end{cases}$$

is called the **simple circulation associated to C**.

Circulations



- The cycle satisfies $A\mathbf{h}^C = \mathbf{0}$.
- We also define its cost as $\mathbf{c}^T\mathbf{h}^C$.
- Note that for any feasible flow $\mathbf{f}$, scalar θ and cycle C, we obtain a new flow $\mathbf{f} + \theta\mathbf{h}^C$ by **pushing** θ units of flow around the cycle C in $\mathbf{f}$.
- Cost changes by $\theta \cdot \mathbf{c}^T\mathbf{h}^C$.

Tree Solutions

Standard Network Flow

- In the following $\sum b_i = 0$ and $\mathcal{G}(\mathcal{N}, \mathcal{A})$ is a directed **connected** graph.
- We study **uncapacitated** problems in this lecture:

$$\begin{array}{ll} \text{minimize} & \mathbf{c}^T\mathbf{f} \\ \text{subject to} & A\mathbf{f} = \mathbf{b} \\ & \mathbf{f} \geq 0 \end{array} \tag{1}$$

  with

  - $\mathbf{f} \in \mathbf{R}^n$ such that $\sum_{i=1^m} b_i = 0$,
  - A the corresponding arc-node incidence matrix in $\mathbf{R}^{m\times n}$.

- We have a problem: $\text{rank}(A) = m - 1$. All the algebra: basis etc.. wont work.
- We start by making sure we work with a constraint matrix A that has l.i. rows.

Dropping out the constraint on the last node

- We assume A and $\mathbf{b}$ have been updated in the following way:

$$A \leftarrow \begin{bmatrix} \cdots & A_1 & \cdots \\ \vdots & \vdots & \vdots \\ \cdots & A_{m-1} & \cdots \end{bmatrix}, \mathbf{b} \leftarrow \begin{bmatrix} b_1 \\ \vdots \\ b_{m-1} \end{bmatrix}.$$

- we have removed the last line of A and last coordinate of $\mathbf{b}$.
- From now on, $A \in \mathbf{R}^{m-1 \times n}$ and $\mathbf{b} \in \mathbf{R}^{m-1}$.
- In practice: we have **dropped** the **flow constraint** of the last node m.

Tree Solutions

Definition 1. *A flow vector* $\mathbf{f}$ *is called a* ***tree solution*** *if it can be constructed by the following procedure.*

(a) Pick a set $\mathbf{T} \subset \mathcal{A}$ *of* $m-1$ *arcs that form a* ***tree*** *when ignoring their directions.*

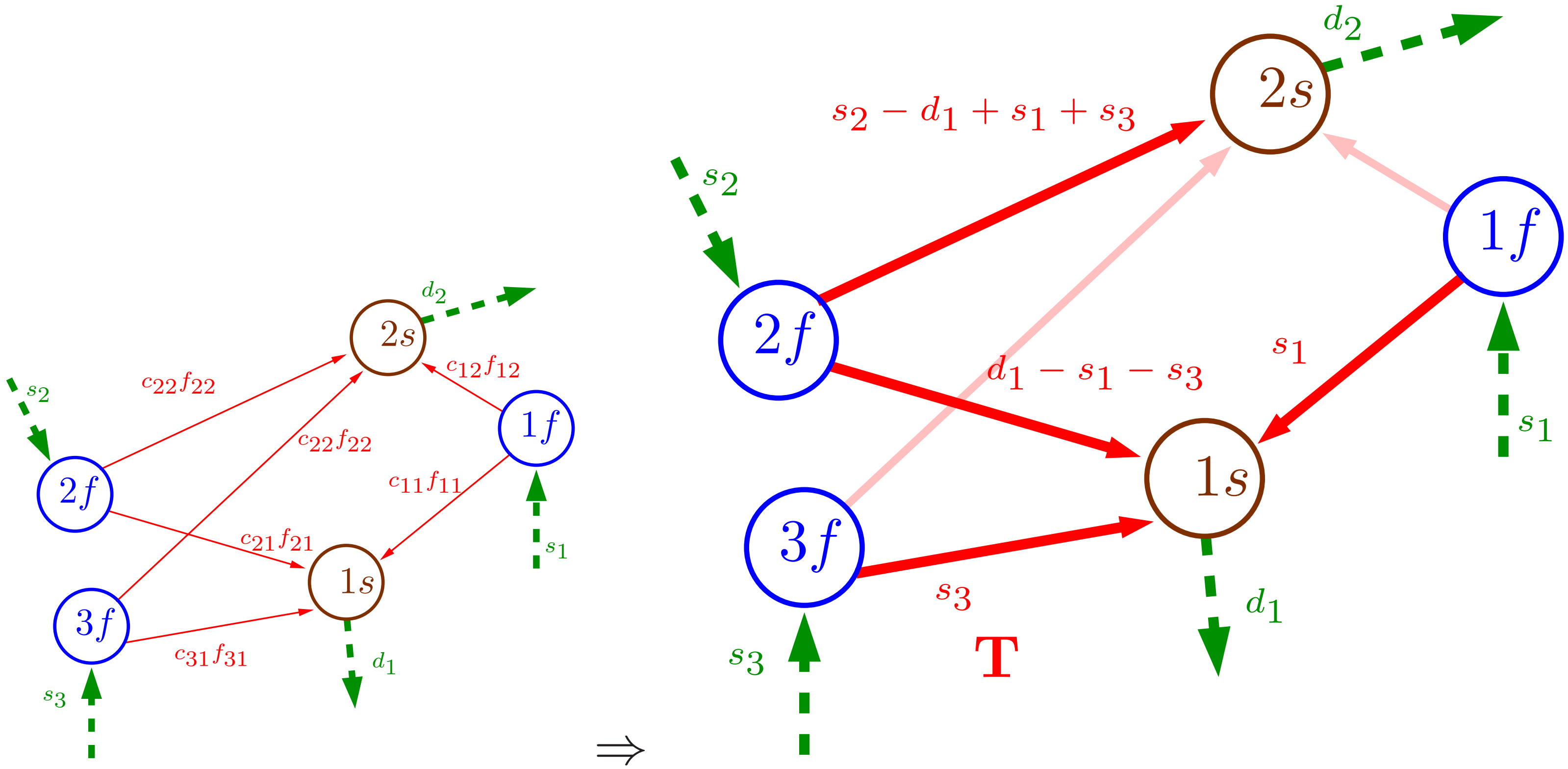
(b) Set $f_a = 0$ *for all other arcs* $a \notin \mathbf{T}$*, exactly* $n - (m-1)$*.*

(c) Use the flow constraints $A\mathbf{f} = \mathbf{b}$ *to set the values of* f *on arcs of* $\mathbf{T}$*.*

- Obvious parallel: choose a basis $\mathbf{I}$ standard LP's / choose a tree $\mathbf{T}$ for NF.

Definition 2. *A tree solution* $\mathbf{f}$ *that satisfies* $\mathbf{f} \geq 0$ *is called a* ***feasible tree solution****.*

Example



- Choose a tree T and proceed by assigning values on leaves, and go up the tree.

Tree solutions exist for all trees T

Theorem 1. *Let* $\mathbf{T} \subset \mathcal{A}$ *be a subset of size* $m-1$ *which forms a tree when ignoring their direction. Let* $B_{\mathbf{T}}$ *be the* $m-1 \times m-1$ *matrix obtained by taking the columns of* A *labelled by* $\mathbf{T}$*. Then* $B_{\mathbf{T}}\mathbf{f}_T = \mathbf{b}$ *has an unique solution.*

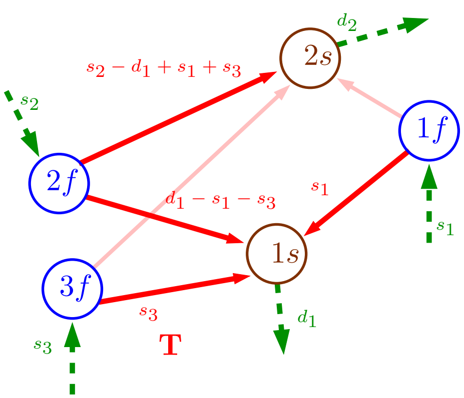
- **Proof** We just need to prove that for all $\mathbf{T}$, $B_{\mathbf{T}}$ is invertible. In order to do that, let's use the following lemma

Lemma: Renumbering Labels and Arcs

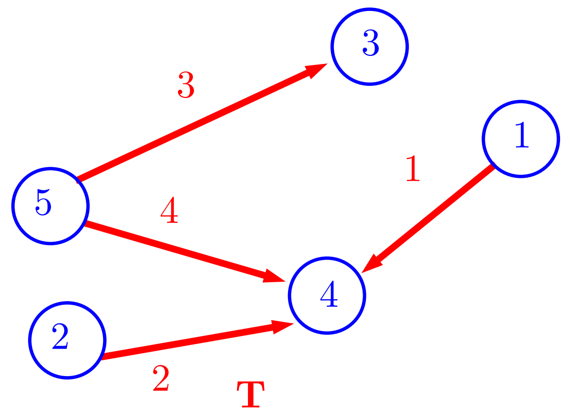
Lemma 1. *Let $\mathbf{T} \subset \mathcal{A}$ be a subset of size $m-1$ which forms a tree when ignoring their direction. Let $B_{\mathbf{T}}$ be the $m-1 \times m-1$ matrix obtained by taking the columns of A labelled by $\mathbf{T}$. Then the rows and columns of $B_{\mathbf{T}}$ can be permuted to make it triangular with ± 1 coefficients on the diagonal.*

- **Proof**: We proceed by renaming labels and arcs described in $\mathcal{T}$, the tree made of arcs $\mathbf{T}$.
- Start with $i = 1$ and $\mathcal{T}_1 \leftarrow \mathcal{T}$
  - if $\mathcal{T}_1 \neq \emptyset$ consider one leaf of $\mathcal{T}_1$.
  - Number it i and remove it from $\mathcal{T}_1$. $i \leftarrow i + 1$.
- The loop finishes after exactly $m-1$ iterations and all nodes of $\mathcal{T}$ have been renumbered.
- Renumber the arcs: If $(i,j) \in \mathbf{T}$, (i,j) receives the number $\min\{i,j\}$.
- Given this renumbering, any path from a leaf to the node labeled m (5 in the example) is unique and increasing with labels.

Lemma: Renumbering Labels and Arcs



$\Rightarrow$



$$\Rightarrow \quad B_{\mathbf{T}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ -1 & -1 & 0 & -1 \end{bmatrix}$$

- With this numbering, the ith column of B corresponds to the ith arc.
- the ith arc joins two nodes i and j with $j > i$, i.e. it is either equal to (i, j) or (j, i).
- Thus any nonzero element in column i will be in row i (on the diagonal) and row j (below diagonal).
- Hence $B_{\mathbf{T}}$ is lower triangular with nonzero diagonal $\Rightarrow$ invertible.

Fundamental parallel between tree solutions and basic solutions

Theorem 2. *A flow vector is a basic solution if and only if it is a tree solution*

Proof

- $(\Rightarrow)$: n variables, $m-1$ constraints. A tree solution has up to $m-1$ nonzero variables.
- $(\Leftarrow)$ By contradiction: suppose $\mathbf{f}$ is **not** a tree solution. We show it cannot be basic.
  - Let $\mathcal{S} = \{a \in \mathcal{A} | f_a \neq 0\}$.
  - If $\#(\mathcal{S}) > m-1$, then $\mathbf{f}$ cannot be basic.
  - If $\mathcal{S}$ does not have a cycle it can be completed by labels to a tree with $m-1$ arcs and is hence a tree solution.
  - Hence $\#(\mathcal{S}) \leq m-1$ and $\mathcal{S}$ has a cycle C.
    - ▷ Let $\mathbf{h}^C$ be its circulation.
    - ▷ Note that $\mathbf{h}_a^C = 0$ for arcs $a \notin \mathcal{S}$.
    - ▷ Then $A(\mathbf{f} + \mathbf{h}^C) = \mathbf{b}$ and the set of constraints is the same.
    - ▷ Hence $\mathbf{f}$ is not basic otherwise it would have been uniquely determined.

Next time

- Continue with Network simplex
- Some comments on complexity
- Some specialized algorithms : Ford-Fulkerson