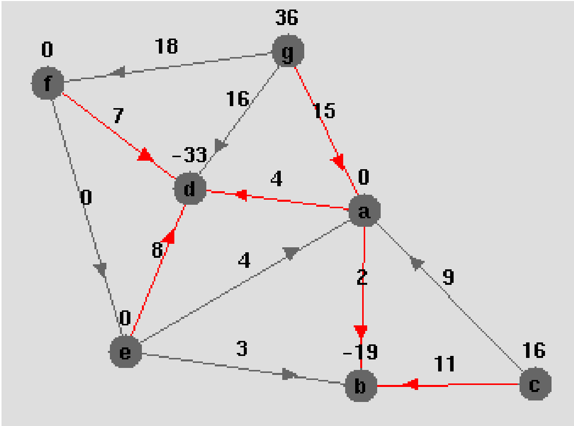
Linear Programming: Chapter 13
Network Flows: Theory

Robert J. Vanderbei

October 17, 2007

Operations Research and Financial Engineering
Princeton University
Princeton, NJ 08544
http://www.princeton.edu/∼rvdb

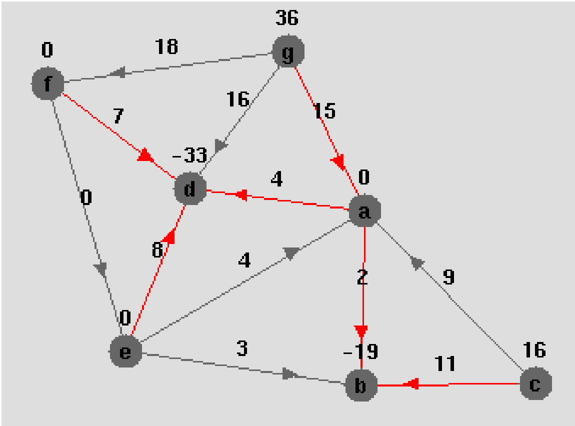
Networks



Basic elements:

- $\mathcal{N}$ *Nodes* (let m denote number of them).
- $\mathcal{A}$ Directed *Arcs*
  - subset of all possible arcs: $\{(i,j): \ i,j \in \mathcal{N}, i \neq j\}$.
  - arcs are *directed*: $(i,j) \neq (j,i)$.

Network Flow Data



- $b_i, \ i \in \mathcal{N}$, *supply* at node i
- $c_{ij}, \ (i,j) \in \mathcal{A}$, *cost* of shipping 1 unit along arc (i,j).

Note: *demands* are recorded as *negative supplies*.

Network Flow Problem

Decision Variables:

- $x_{ij},\ (i,j) \in \mathcal{A}$, *quantity* to ship along arc (i,j).

Objective:

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

Network Flow Problem–Cont.

Constraints:

- Mass conservation (aka flow balance):

$$\text{inflow}(k) - \text{outflow}(k) = \text{demand}(k) = -\text{supply}(k), \qquad k \in \mathcal{N}$$

$$\Updownarrow$$

$$\sum_{\substack{i: \\ (i,k) \in \mathcal{A}}} x_{ik} - \sum_{\substack{j: \\ (k,j) \in \mathcal{A}}} x_{kj} = -b_k, \qquad k \in \mathcal{N}$$

- Nonnegativity:

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

Matrix Notation

$$
\begin{array}{ll}
\text{minimize} & c^T x \\
\text{subject to} & Ax = -b \\
 & \quad x \geq 0
\end{array}
$$

where

$$
c^T = \begin{bmatrix} 2 & 4 & 9 & 11 & 4 & 3 & 8 & 7 & 0 & 15 & 16 & 18 \end{bmatrix}
$$

$$
A = \begin{bmatrix}
-1 & -1 & 1 & & 1 & & & & & 1 & & \\
1 & & & 1 & & 1 & & & & & & \\
 & & -1 & -1 & & & & & & & & \\
 & 1 & & & & & 1 & 1 & & & 1 & \\
 & & & & -1 & -1 & -1 & & 1 & & & \\
 & & & & & & & -1 & -1 & & & 1 \\
 & & & & & & & & & -1 & -1 & -1
\end{bmatrix}, \quad b = \begin{bmatrix} 0 \\ -19 \\ 16 \\ -33 \\ 0 \\ 0 \\ 36 \end{bmatrix}
$$

Notes

- A is called *node-arc incidence matrix*.

- A is *large* and *sparse*.

Dual Problem

$$
\begin{array}{ll}
\text{maximize} & -b^T y \\
\text{subject to} & A^T y + z = c \\
& \qquad\quad z \geq 0
\end{array}
$$

In network notation:

$$
\begin{array}{lll}
\text{maximize} & -\sum_{i \in \mathcal{N}} b_i y_i & \\
\text{subject to} & y_j - y_i + z_{ij} = c_{ij} & (i,j) \in \mathcal{A} \\
& \qquad\qquad z_{ij} \geq 0 & (i,j) \in \mathcal{A}
\end{array}
$$

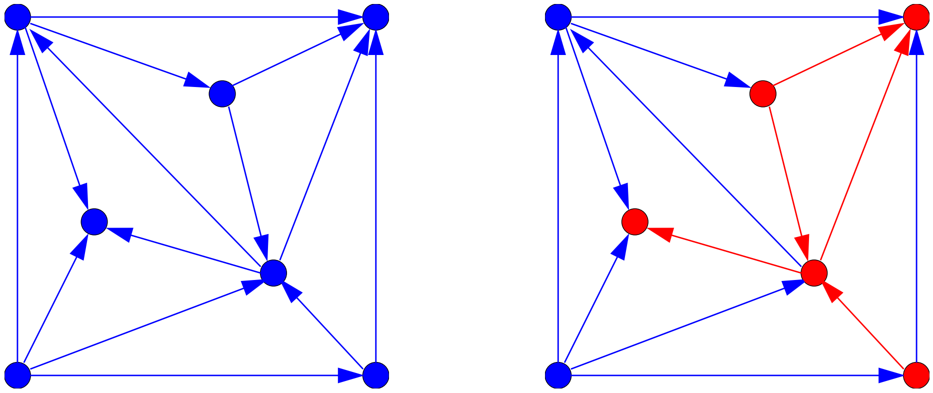
Complementarity Relations

- The primal variables must be nonnegative.
- Therefore the associated dual constraints are inequalities.
- The dual slack variables are complementary to the primal variables:

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

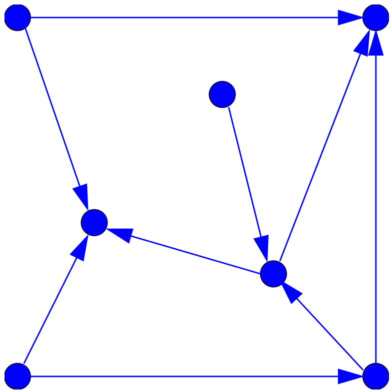
- The primal constraints are equalities.
- Therefore they have no slack variables.
- The corresponding dual variables, the y_i's, are free variables.
- No complementarity conditions apply to them.

Definition: Subnetwork

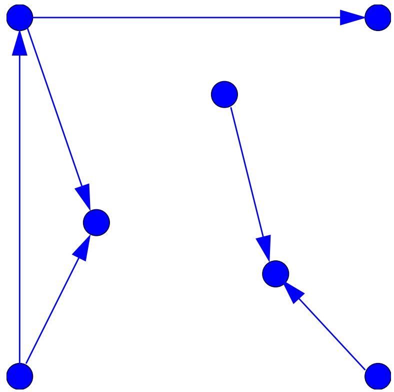


Network Subnetwork

Connected vs. Disconnected

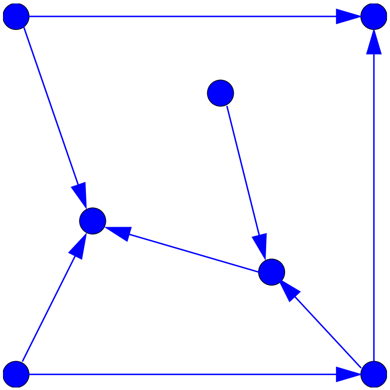


Connected

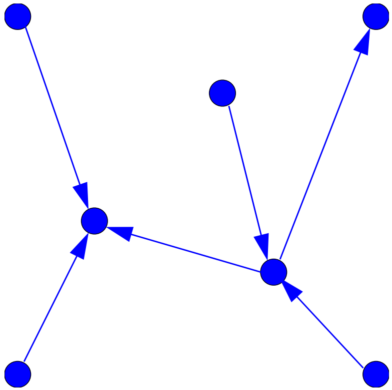


Disconnected

Cyclic vs. Acyclic

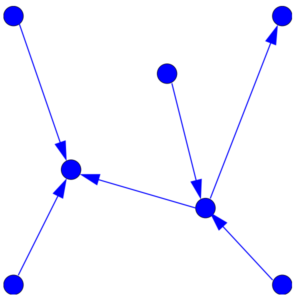


Cyclic

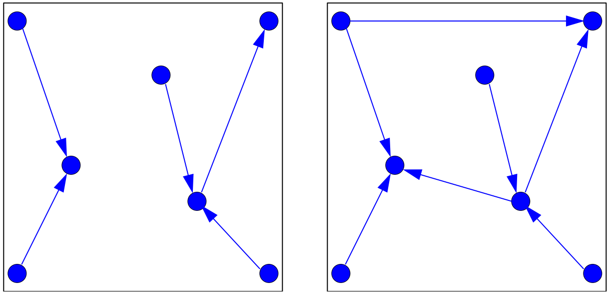


Acyclic

Trees

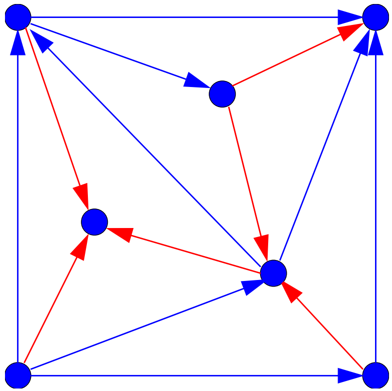


Tree = Connected + Acyclic



Not Trees

Spanning Trees



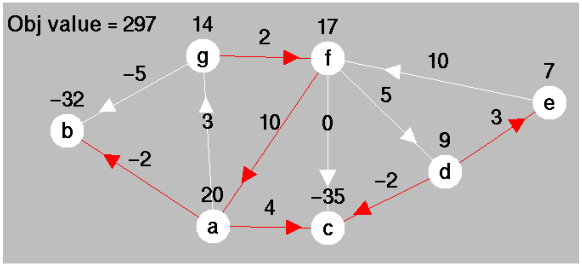
Spanning Tree–A tree touching every node

Tree Solution

$$x_{ij} = 0 \qquad \text{for } (i,j) \notin \text{Tree Arcs}$$

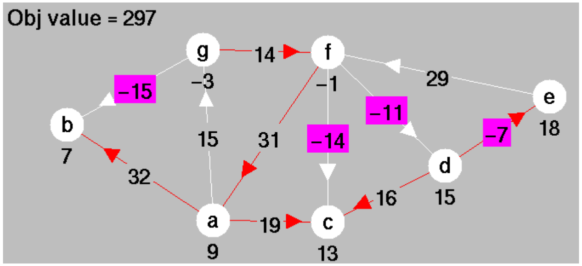
Note: Tree solutions are easy to compute—start at the leaves and work inward...

Online Pivot Tool–Notations



Data:

- *Costs* on arcs shown above arcs.
- *Supplies* at nodes shown above nodes.

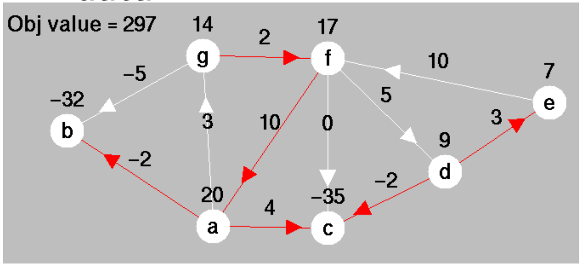


Variables:

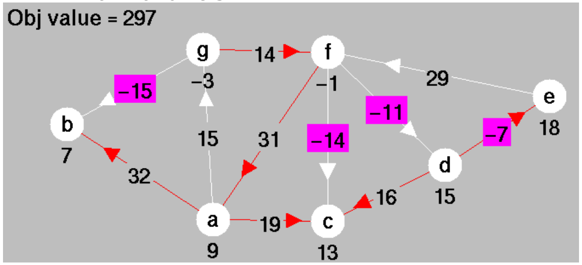
- *Primal flows* shown on tree arcs.
- *Dual slacks* shown on non-tree arcs.
- *Dual variables* shown below nodes.

Tree Solutions–An Example

data



variables



- Fix a root node, say a.
- *Primal flows* on tree arcs calculated recursively from leaves inward.
- *Dual variables* at nodes calculated recursively from root node outward along tree arcs using:
  $$y_j - y_i = c_{ij}$$
- *Dual slacks* on nontree arcs calculated using:
  $$z_{ij} = y_i - y_j + c_{ij}.$$