Lecture outline

- **Substitutable resources**
- **Linear programming review**

Linear programming review

- Linear programming formulation

$$\min \sum_i \sum_j c_{ij} x_{ij}$$

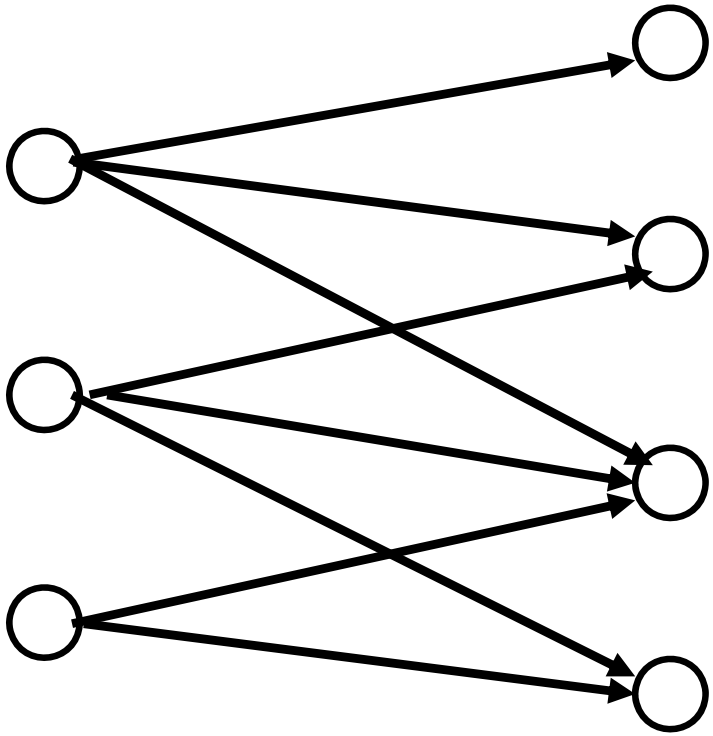
subject to

$$Ax = b$$

$$x \le u$$

$$x \ge 0$$

  - » You can use any of a number of solvers (matlab, Cplex, Gurobi, MP-Express, …) to solve this problem.
  - » Our goal is to derive insights from the solution.



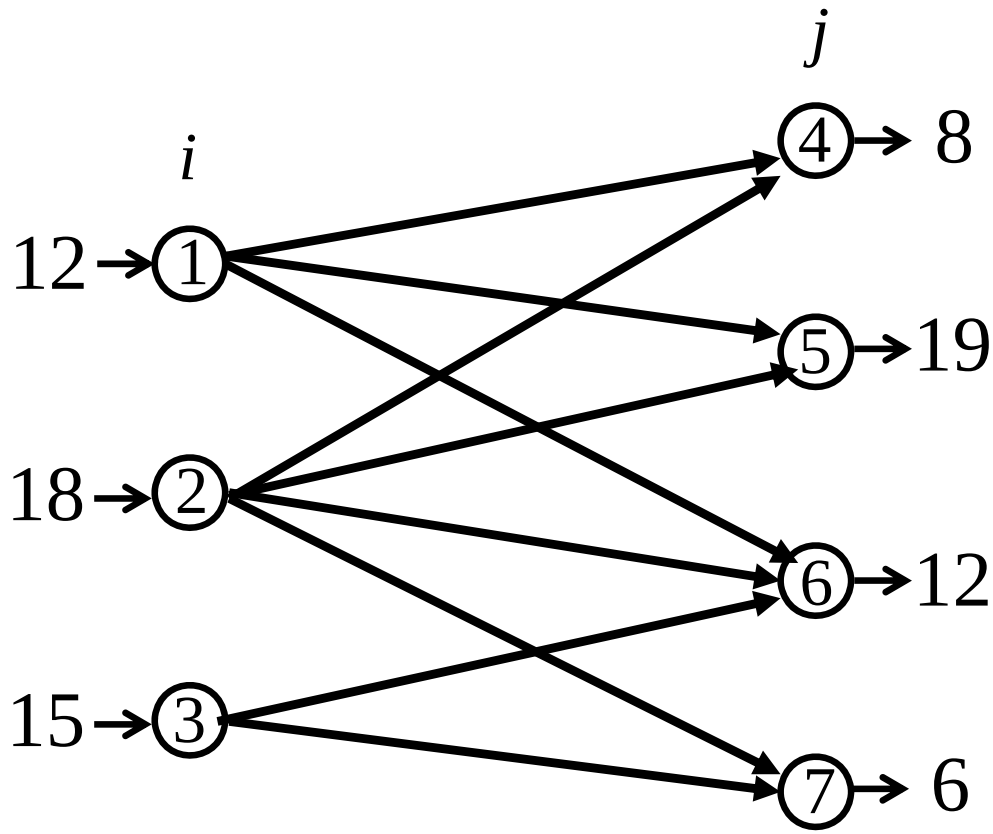
Linear programming review

The linear programming formulation

We begin by writing out the constraint matrix for our network problem. Generic flow conservation constraint is written as

$$\sum_j x_{ij} - \sum_k x_{ki} = S_i = \text{Net supply (>0)}$$

or demand (<0)

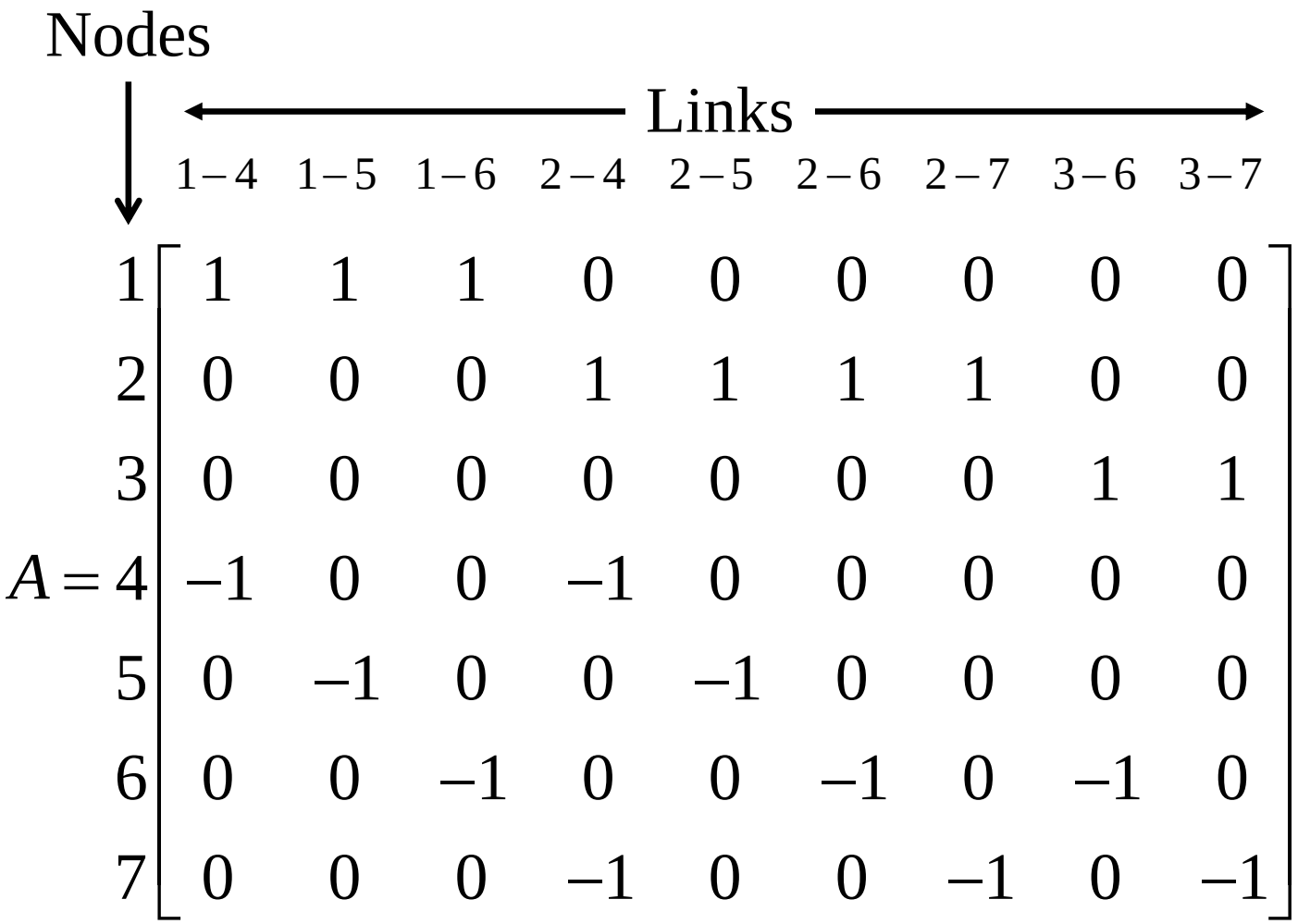


For the graph at the right:

$$x_{14} + x_{15} + x_{16} = 12$$
$$x_{24} + x_{25} + x_{26} + x_{27} = 18$$
$$\ldots$$
$$-x_{14} - x_{24} = -8$$
$$-x_{15} - x_{25} = -19$$

Linear programming review

- The constraint matrix



Linear programming review

- Notes:
  - If we add the rows of the matrix A, they sum to zero. This means that one of the rows is redundant – if we have *n* nodes, we only need *n*-1 constraints.
  - To illustrate: consider a two node network:

  - If we guarantee flow conservation at node 1, then we automatically guarantee flow conservation at node 2.
  - In the larger network, pick one node (any node) and drop it. We call this node the *root node*.
  - Let's pick node 1.

Linear programming review

- Dropping node 1, our constraint matrix now looks like:

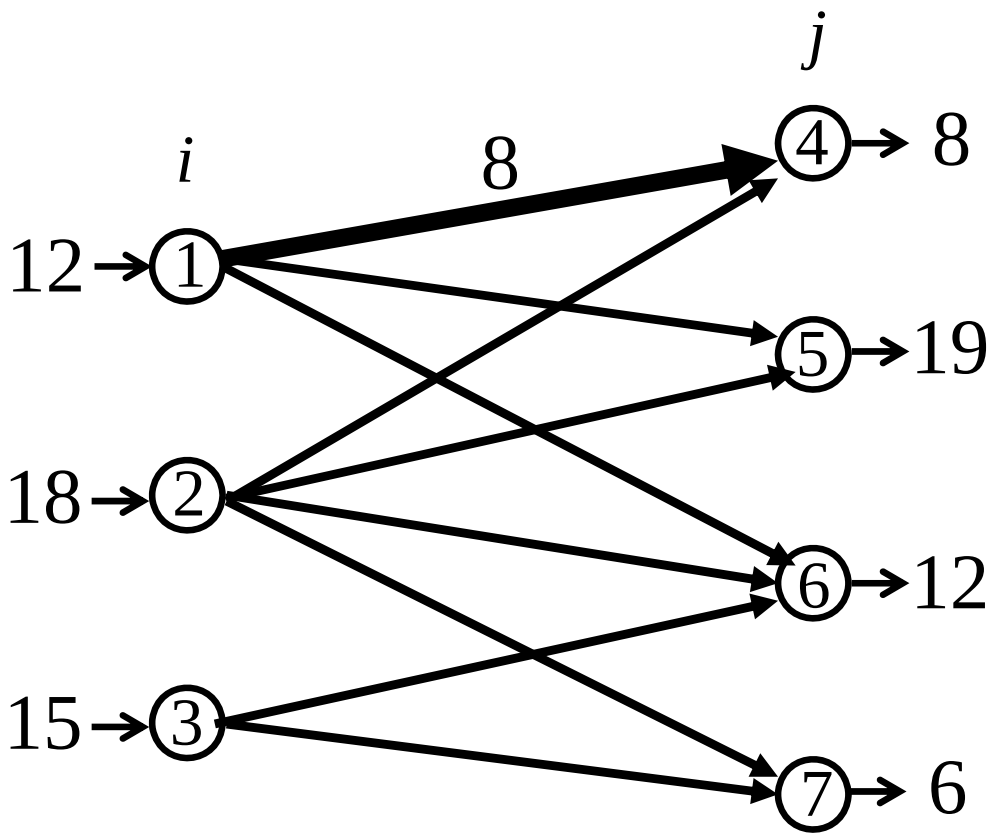
$$A = \begin{array}{c} \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \end{array} \begin{array}{c} \begin{array}{ccccccccc} 1-4 & 1-5 & 1-6 & 2-4 & 2-5 & 2-6 & 2-7 & 3-6 & 3-7 \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ -1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & -1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & -1 \end{bmatrix} \end{array}$$

Linear programming review

- Simplex review:
  - Remember – we pick a set of columns that form a *basis*. We are going to then divide our decision vector into *basic variables* and *nonbasic variables*.
  - How do we do this?
  - We are going to find an initial solution using a simple rule called the “northwest corner rule.”

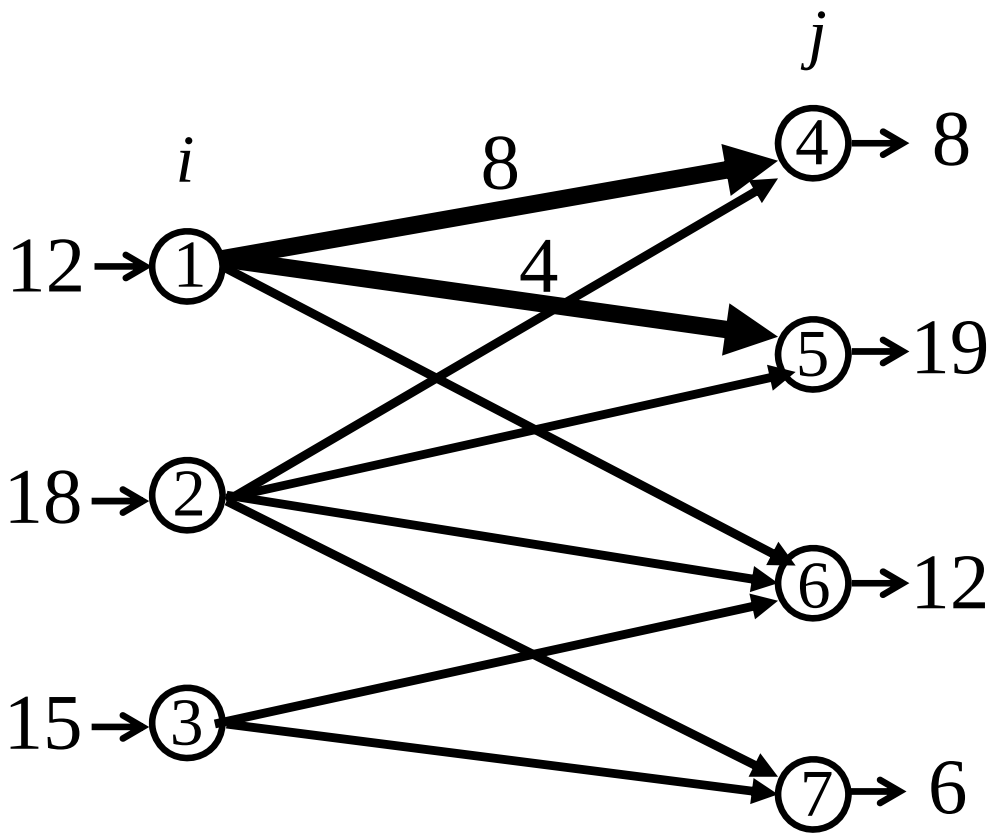
Linear programming review

- Start with node 1 (the "highest" node on the left)
- Assign as much flow to node 4 (the highest node on the right).



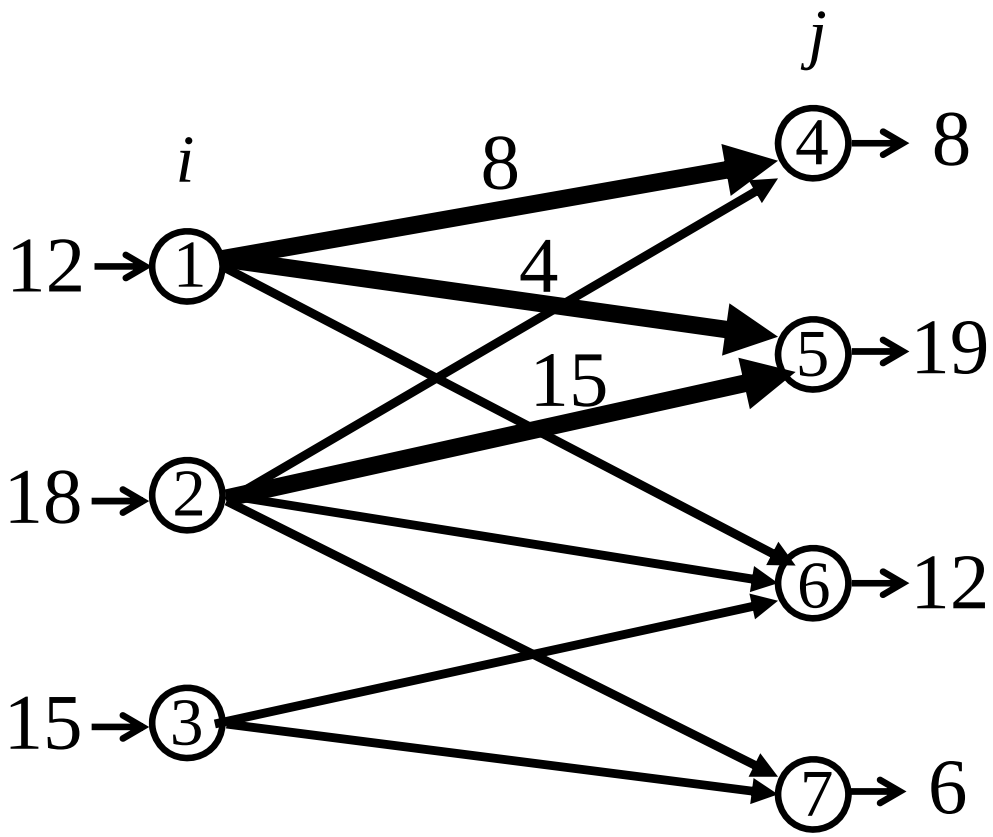
Linear programming review

- Start with node 1 (the "highest" node on the left)
- Assign as much flow to node 4 (the highest node on the right).
- When one node is saturated, "pivot" to the next lower node on the list.



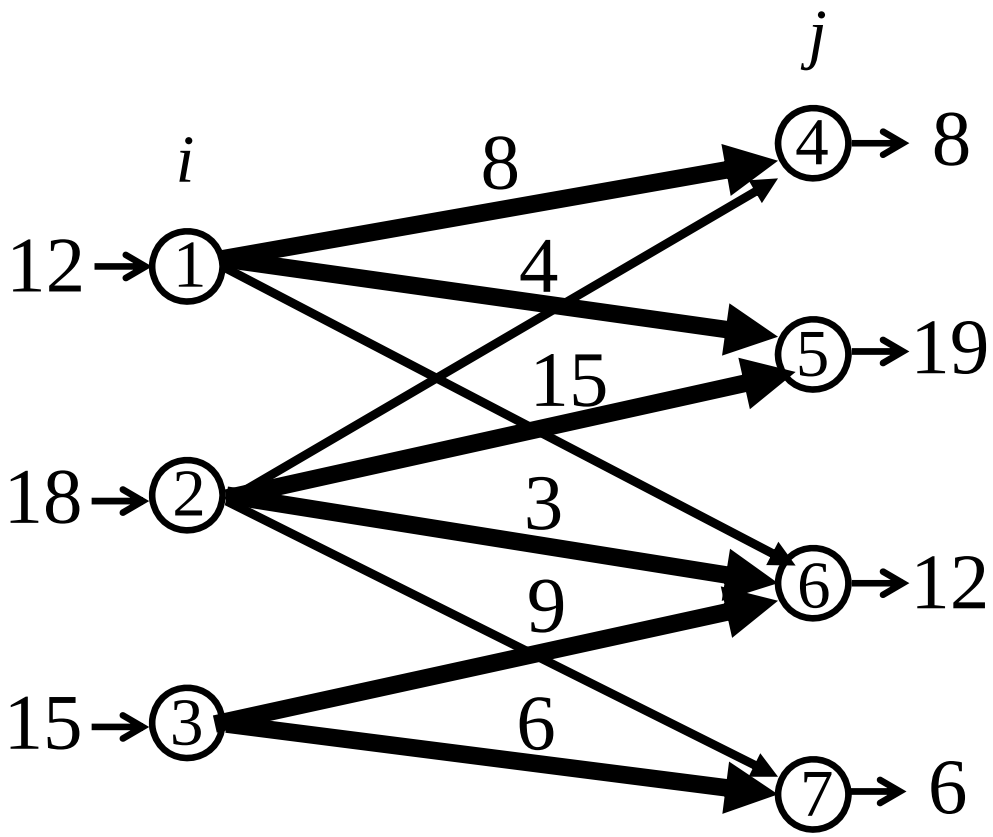
Linear programming review

- Start with node 1 (the "highest" node on the left)
- Assign as much flow to node 4 (the highest node on the right).
- When one node is saturated, "pivot" to the next lower node on the list.



Linear programming review

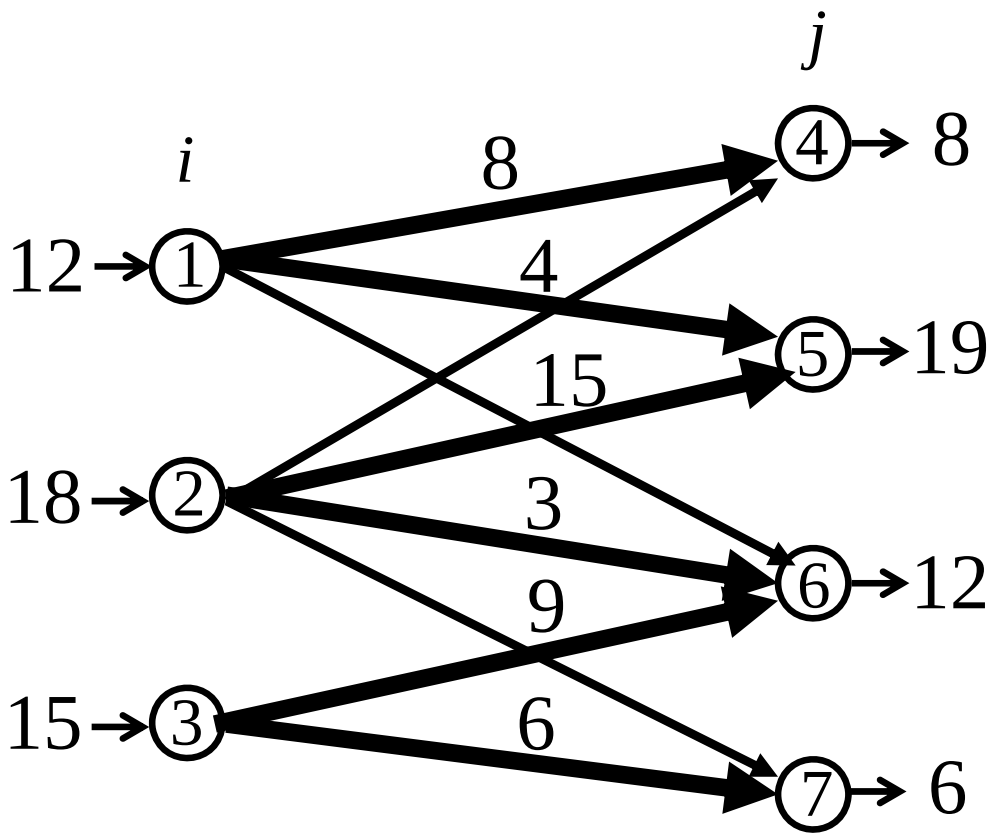
- Start with node 1 (the "highest" node on the left)
- Assign as much flow to node 4 (the highest node on the right).
- When one node is saturated, "pivot" to the next lower node on the list.
- We use this logic to assign the remaining flow.



Linear programming review

- The links with flow (wide solid links) represent the basis.
- The links without flow are *nonbasic*.

$$x^B = \begin{bmatrix} 1-4 \\ 1-5 \\ 2-5 \\ 2-6 \\ 3-6 \\ 3-7 \end{bmatrix} \qquad x^N = \begin{bmatrix} 1-6 \\ 2-4 \\ 2-7 \end{bmatrix}$$



Linear programming review

- This allows us to partition the A matrix into basic and non-basic columns:

$$A^B = \begin{array}{c} \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \end{array} \begin{array}{c} \begin{array}{cccccc} 1-4 & 1-5 & 2-5 & 2-6 & 3-6 & 3-7 \end{array} \\ \begin{bmatrix} 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix} \end{array} \qquad A^N = \begin{array}{c} \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \end{array} \begin{array}{c} \begin{array}{ccc} 1-6 & 2-4 & 2-7 \end{array} \\ \begin{bmatrix} 0 & 1 & 1 \\ 0 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & -1 \end{bmatrix} \end{array}$$

Linear programming review

- Let's restate our constraints:

$$Ax = b$$

  - … becomes

$$\begin{bmatrix} A^B & A^N \end{bmatrix} \begin{bmatrix} x^B \\ x^N \end{bmatrix} = b$$

  - …. or

$$A^B x^B + A^N x^N = b$$

  - We can now solve for x^B as a function of x^N

$$x^B = \left[A^B \right]^{-1} \left[b - A^N x^N \right]$$

Linear programming review

$$x^B = \left[A^B\right]^{-1}\left[b - A^N x^N\right]$$

» By construction, $x^N = 0$, but we can use this equation to understand the effect of increasing a nonbasic variable by adjusting the basic variables so that we maintain conservation of flow. But it means we have to find the inverse of A^B

» We are going to do this by inspection! You can view this as magic, or a cheap parlor trick.

Linear programming review

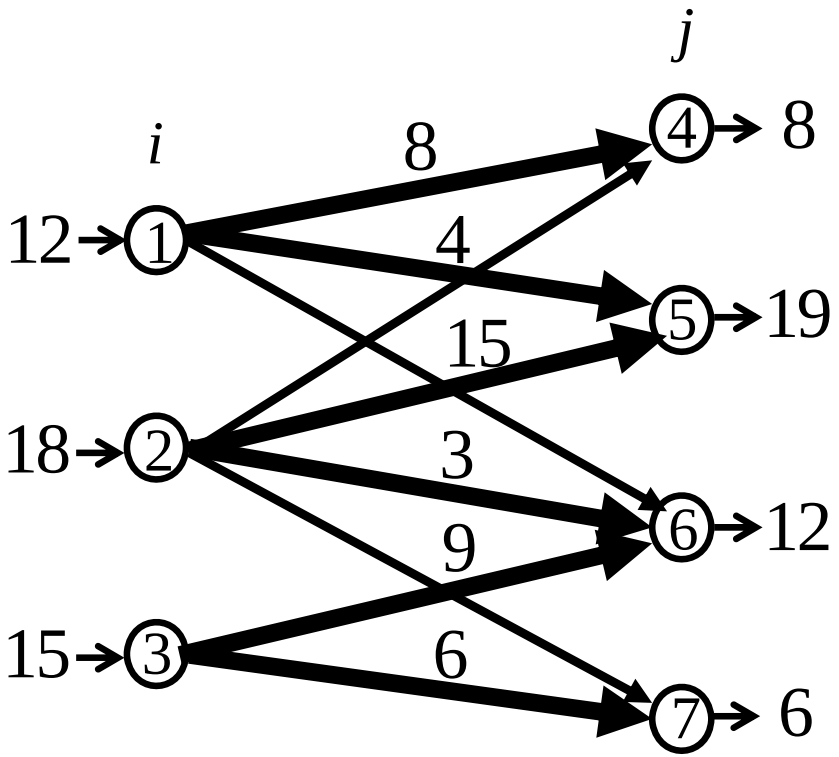
- The inverse

Links

Paths

	2–1	3–1	4–1	5–1	6–1	7–1
1–4	0	0	–1	0	0	0
1–5	–1	–1	0	–1	–1	–1
2–5	1	1	0	0	1	1
2–6	0	–1	0	0	–1	–1
3–6	0	1	0	0	0	1
3–7	0	0	0	0	0	–1

$$\left[A^B\right]^{-1} =$$



» Each column traces a path from a node to the root node.

Linear programming review

- Verify that we have an inverse:

$$A^B \qquad \left[A^B\right]^{-1} \qquad I$$

$$\begin{bmatrix} 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix} \times \begin{bmatrix} 0 & 0 & -1 & 0 & 0 & 0 \\ -1 & -1 & 0 & -1 & -1 & -1 \\ 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & -1 & 0 & 0 & -1 & -1 \\ 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Linear programming review

- Notes:
  - » We refer to A^B as a "node-arc" incide matrix (rows are nodes, columns are arcs (links). The matrix $\left[A^B\right]^{-1}$ is an "arc-chain" incidence matrix. Rows are arcs (links), while columns correspond to paths from each node in the network to the root node (node 1).
  - » We can rewrite our objective function using

$$\begin{aligned}\min c^T x &= \left(c^B\right)^T x^B + \left(c^N\right)^T x^N \\ &= \left(c^B\right)^T \left(\left[A^B\right]^{-1}\left(b - A^N x^N\right)\right) + \left(c^N\right)^T x^N \\ &= \left(c^B\right)^T \left[A^B\right]^{-1} b - \left(c^B\right)^T \left[A^B\right]^{-1} A^N x^N + \left(c^N\right)^T x^N \\ &= \left(c^B\right)^T \left[A^B\right]^{-1} b + \bar{c}^N x^N\end{aligned}$$

where

$$\bar{c}^N = \text{"reduced costs"} = \left(c^N\right)^T - \left(c^B\right)^T \left[A^B\right]^{-1} A^N$$

Linear programming review

- Reduced costs tell us if we should increase flow on a nonbasic link, while adjusting flows on all basic links.

$$\bar{c} = \left(c^N\right)^T - \left(c^B\right)^T \left[A^B\right]^{-1} A^N = \text{The "reduced cost"}$$

$$= \begin{bmatrix} c_{16} & c_{24} & c_{27} \end{bmatrix} - \begin{bmatrix} c_{14} & c_{15} & c_{25} & c_{26} & c_{36} & c_{37} \end{bmatrix}$$

"paths"

$$\begin{bmatrix} 2 & 3 & 4 & 5 & 6 & 7 \end{bmatrix}$$

$$\begin{bmatrix} 0 & 0 & -1 & 0 & 0 & 0 \\ -1 & -1 & 0 & -1 & -1 & -1 \\ 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & -1 & 0 & 0 & -1 & -1 \\ 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix}$$

Nonbasic links

$$\begin{matrix} 1-6 & 2-4 & 2-7 \end{matrix}$$

$$\begin{bmatrix} 0 & 1 & 1 \\ 0 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & -1 \end{bmatrix}$$

Linear programming review

- Reduced costs (cont'd)

$$\bar{c} = \begin{bmatrix} c_{16} & c_{24} & c_{27} \end{bmatrix} - \begin{bmatrix} c_{14} & c_{15} & c_{25} & c_{26} & c_{36} & c_{37} \end{bmatrix} \begin{bmatrix} 0 & 0 & -1 & 0 & 0 & 0 \\ -1 & -1 & 0 & -1 & -1 & -1 \\ 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & -1 & 0 & 0 & -1 & -1 \\ 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} 0 & 1 & 1 \\ 0 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & -1 \end{bmatrix}$$

$$= \begin{bmatrix} c_{16} & c_{24} & c_{27} \end{bmatrix} - \overbrace{\begin{bmatrix} v_2 & v_3 & v_4 & v_5 & v_6 & v_7 \end{bmatrix}}^{\left(c^B\right)^T \left[A^B\right]^{-1}} \begin{bmatrix} 0 & 1 & 1 \\ 0 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & -1 \end{bmatrix}$$

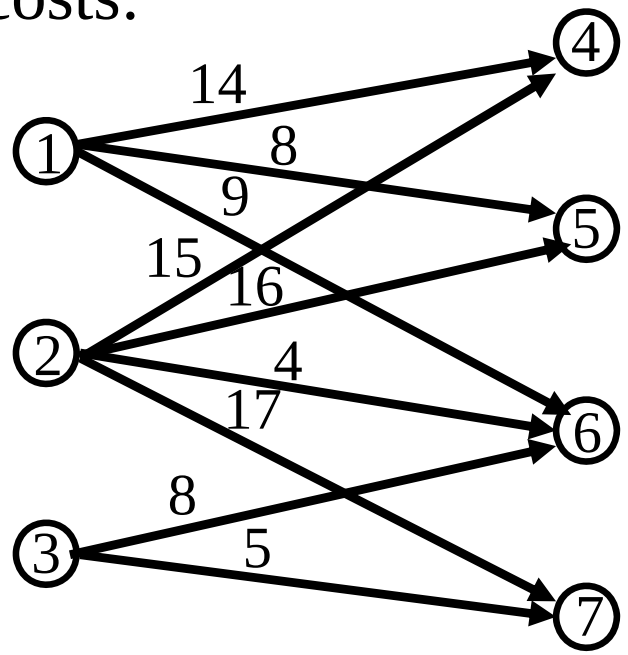
$$= \begin{bmatrix} c_{16} & c_{24} & c_{27} \end{bmatrix} - \begin{bmatrix} -v_6 & v_2 - v_4 & v_2 - v_7 \end{bmatrix}$$

$$= \begin{bmatrix} c_{16} - v_1 + v_6 & c_{24} - v_2 + v_4 & c_{27} - v_2 + v_7 \end{bmatrix}$$

Linear programming review

- Computing "reduced costs"
  - » Assume that we have the following costs:
  - » The path costs for each node are:

$$\left(c^B\right)^T\left[A^B\right]^{-1} = \begin{aligned} v_2 &= 16-8=8 \\ v_3 &= 8-4+16-8=12 \\ v_4 &= -14 \\ v_5 &= -8 \\ v_6 &= -4+16-8=4 \\ v_7 &= -5+8-4+16-8=7 \end{aligned}$$



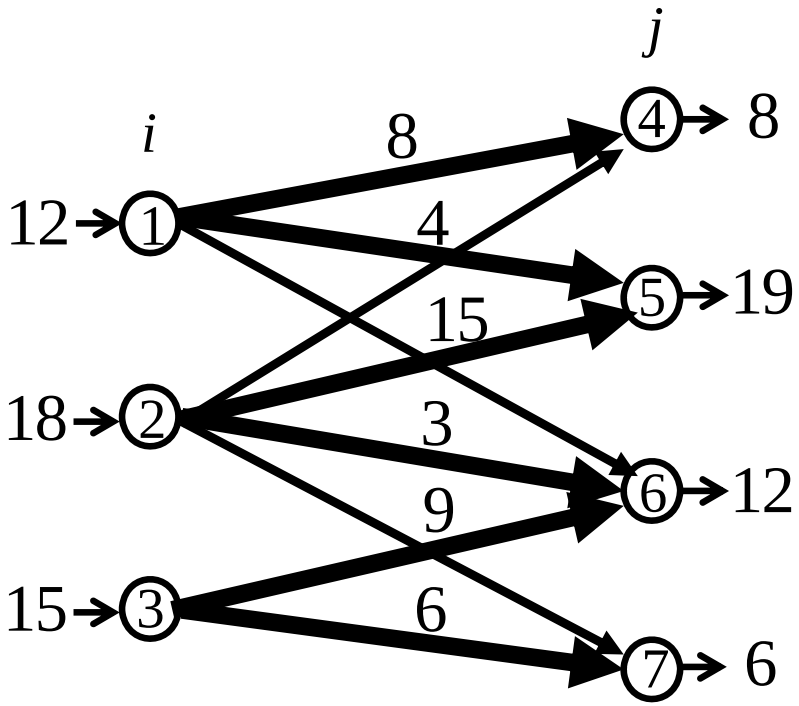
  - » The reduced costs are

$$\begin{aligned} \bar{c} &= \left[\bar{c}_{16} \quad \bar{c}_{24} \quad \bar{c}_{27}\right] \\ &= \left[c_{16}+v_6 \quad c_{24}-v_2+v_4 \quad c_{27}-v_2+v_7\right] \\ &= \left[9+4 \quad 15-8+(-14) \quad 17-8+7\right] \\ &= \left[13 \quad -7 \quad 16\right] \end{aligned}$$

So, should we increase flow on any of the nonbasic links? If so, which one(s)?

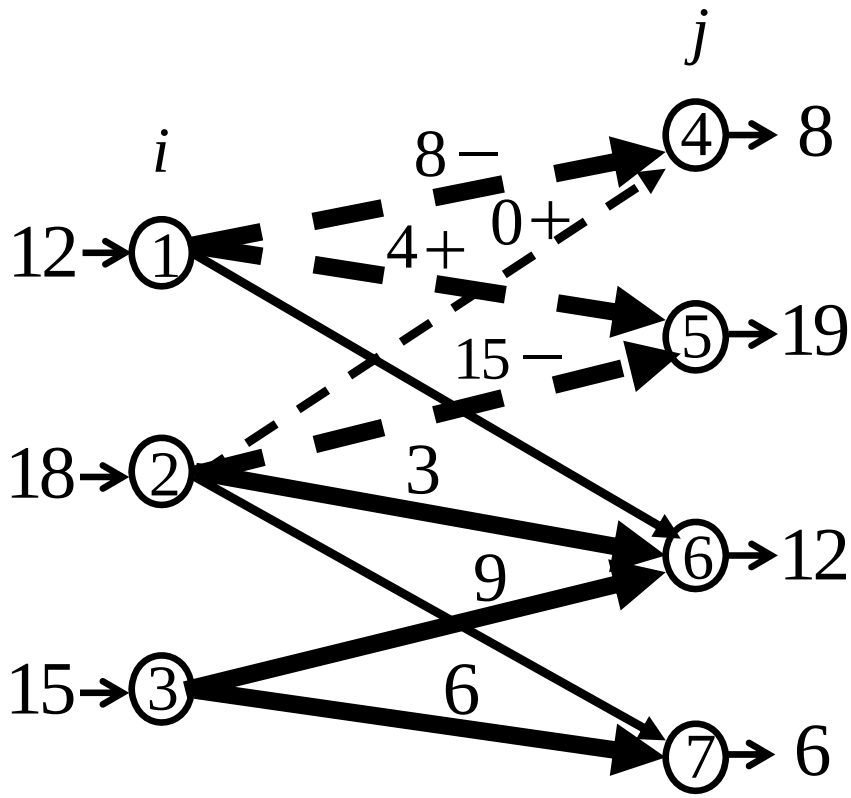
Linear programming review

- Computing “reduced costs”
  - » Consider what happens if we want to increase flow on the nonbasic link from 2 to 4
    - To maintain flow conservation, we need to adjust flows on the basic links.
    - We do this by “pushing” flow from the root node to node 2, and then from 4 back to the root node. This creates a cycle.



Linear programming review

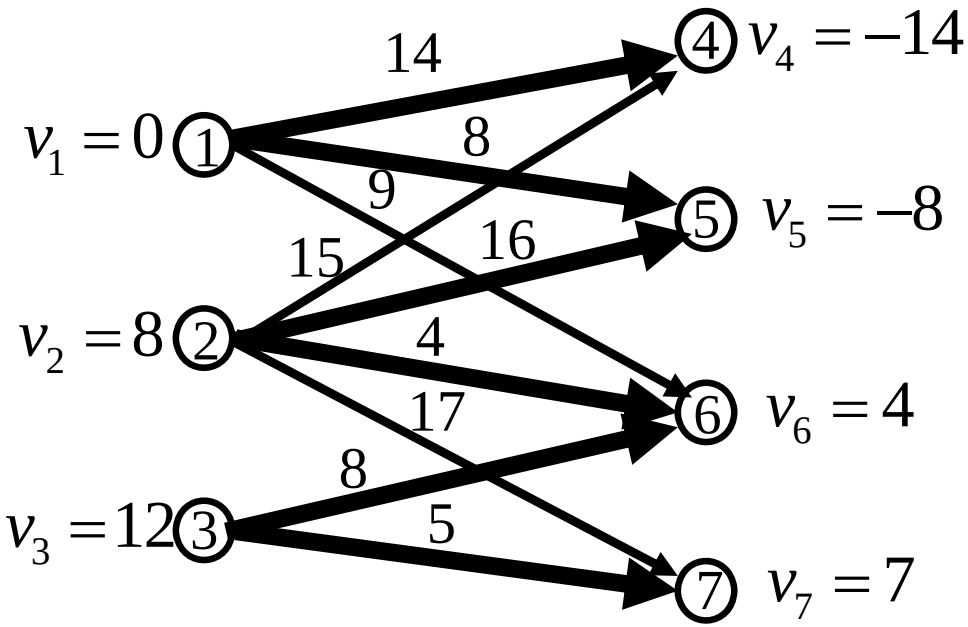
- Pushing flow around the cycle
  - » Imagine that we want to increase the flow from 2 to 4. How does this affect the flows on the rest of the network? This is called the "flow augmenting cycle"
  - » How much flow can we move around the cycle produced by adding the nonbasic link from 2 to 4?
  - » As we move flow around the cycle, can we guarantee that we maintain flow conservation at all times?



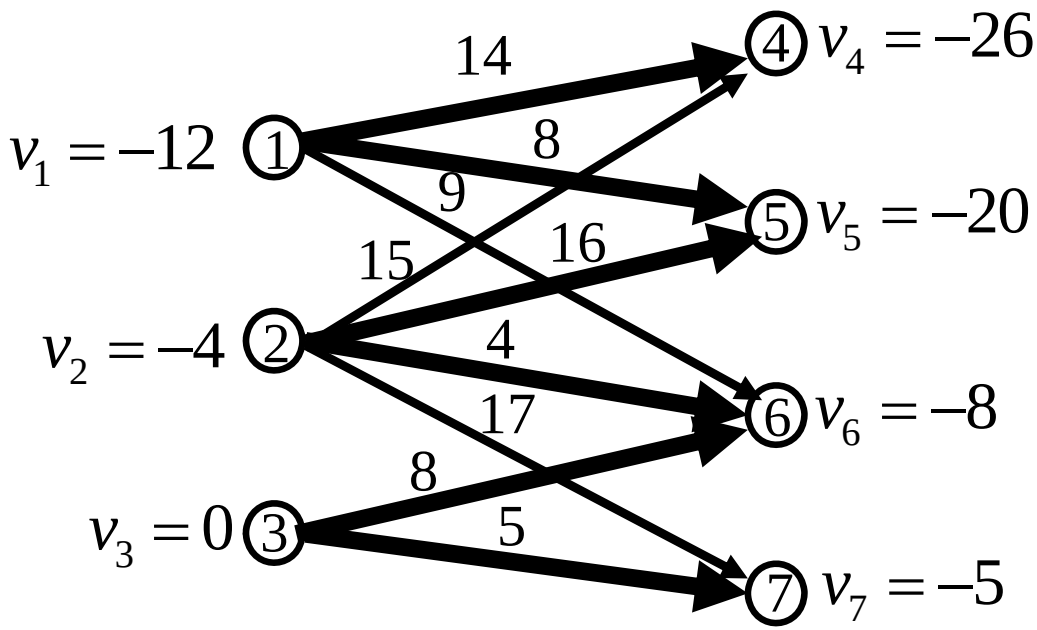
Linear programming review

- All of these calculations depend on our choice of node 1 as the root node.
  - » What if we picked a different node?

Node 1 = root node



Node 3 = root node



  - » Notice that we do not need to do any fancy linear algebra to calculate the path costs.

Linear programming review

- Changing the root node (cont'd)
  - » How did changing the root node affect our reduced costs?
  - » Root node = node 1

$$\begin{aligned}\bar{c} &= \begin{bmatrix}\bar{c}_{16} & \bar{c}_{24} & \bar{c}_{27}\end{bmatrix} \\ &= \begin{bmatrix}c_{16}-v_1+v_6 & c_{24}-v_2+v_4 & c_{27}-v_2+v_7\end{bmatrix} \\ &= \begin{bmatrix}9-0+4 & 15-8+(-14) & 17-8+7\end{bmatrix} \\ &= \begin{bmatrix}13 & -7 & 16\end{bmatrix}\end{aligned}$$

  - » Root node = node 3

$$\begin{aligned}\bar{c} &= \begin{bmatrix}\bar{c}_{16} & \bar{c}_{24} & \bar{c}_{27}\end{bmatrix} \\ &= \begin{bmatrix}c_{16}-v_1+v_6 & c_{24}-v_2+v_4 & c_{27}-v_2+v_7\end{bmatrix} \\ &= \begin{bmatrix}9-(-12)+(-8) & 15-(-4)+(-26) & 17-(-4)+(-5)\end{bmatrix} \\ &= \begin{bmatrix}13 & -7 & 16\end{bmatrix}\end{aligned}$$

  - » They are the same! Choice of root node does not change the reduced costs.

Linear programming review

- Dual variables
  - » Recall that our constraints look like:

$$\begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ -1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & -1 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & -1 & 0 & -1 \end{bmatrix} \begin{bmatrix} x_{14} \\ x_{15} \\ x_{16} \\ x_{24} \\ x_{25} \\ x_{26} \\ x_{27} \\ x_{36} \\ x_{37} \end{bmatrix} = \begin{bmatrix} 18 \\ 15 \\ -8 \\ -19 \\ -12 \\ -6 \end{bmatrix}$$

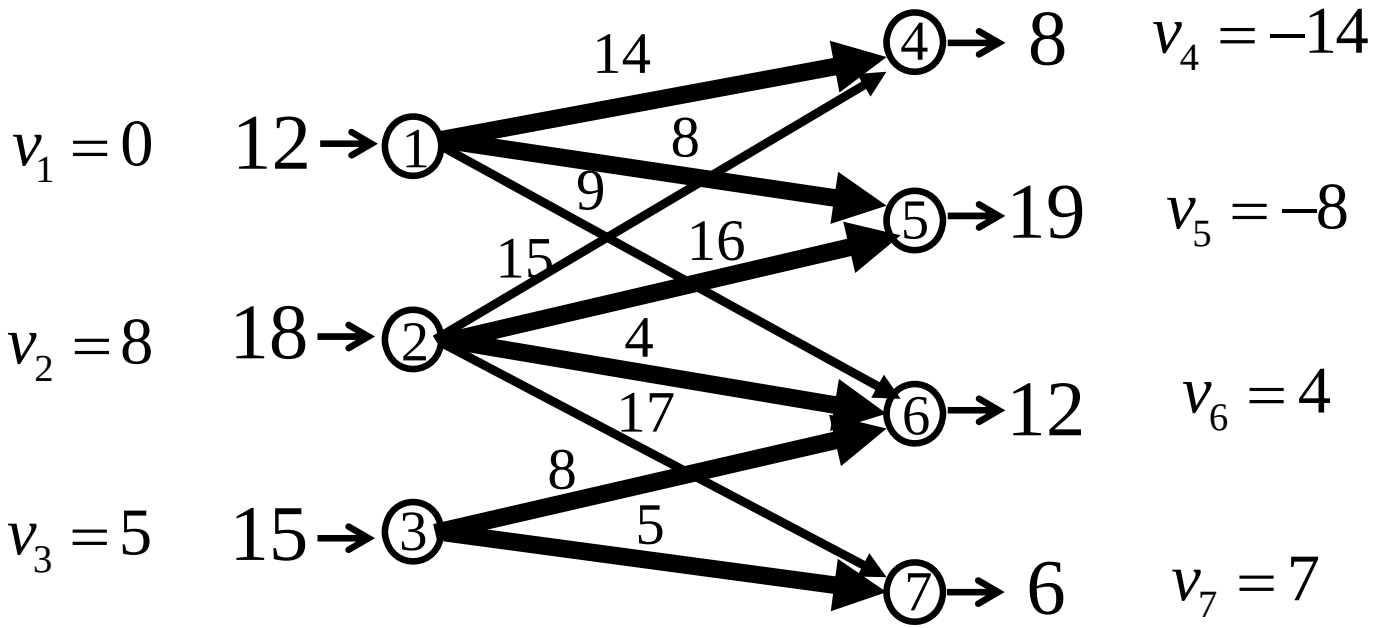
Dual variables

$$\begin{bmatrix} v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \\ v_7 \end{bmatrix}$$

  - » What is the meaning of a dual variable?
  - » How can we calculate it for this problem?

Linear programming review

- The dual variable is often described as the marginal value of a resource (the right hand side constraint).
  - » What happens if we increase the flow into node 2 from 18 to 19?
  - » What happens if we "increase" the right hand side at node 6 from -12 to -11? (Recall how the flow conservation constraint is written).



Linear programming review

- Dual variables (cont'd)
  - The root node serves as a slack node.
  - It allows us to change the supply/demand at any node, and perturb the solution without violating conservation of flow (remembering that we no longer have a conservation of flow constraint at the root node).
  - If we change the root node, do the dual variables change? Do we care?
  - So – dual variables can be arbitrary, but differences between dual variables are meaningful.