Integer Programming

Nonlinear Objective Function [V] p.394

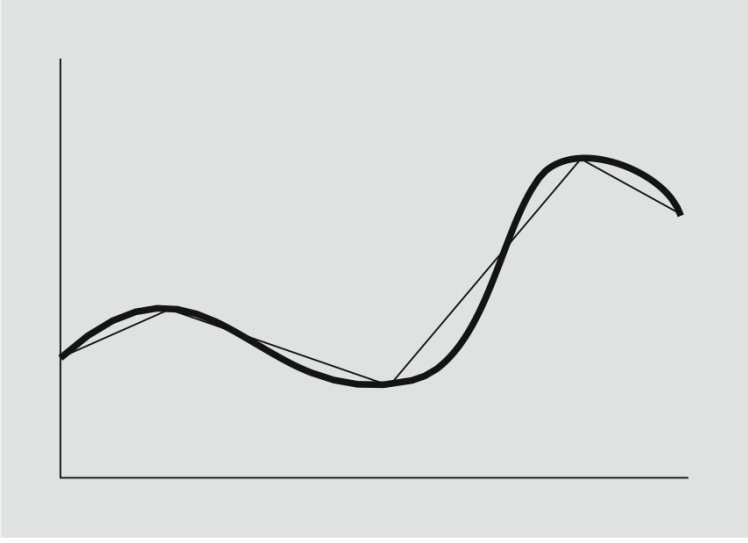


FIGURE 23.3. A nonlinear function and a piecewise linear approximation to it.

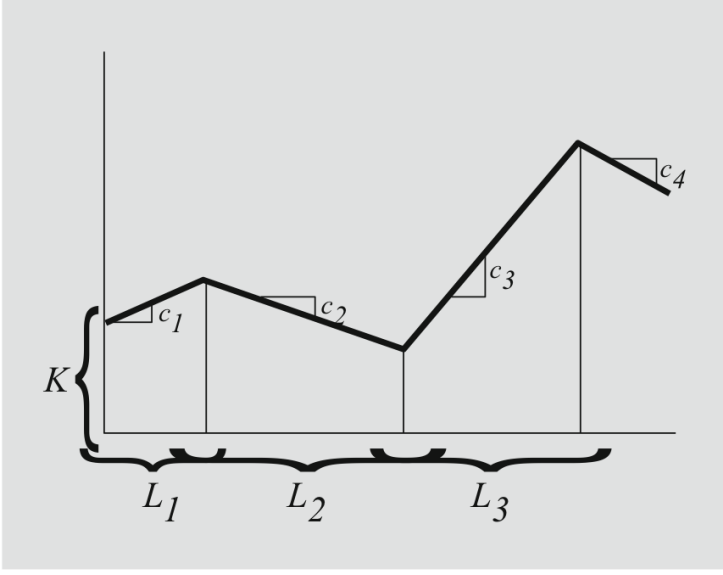
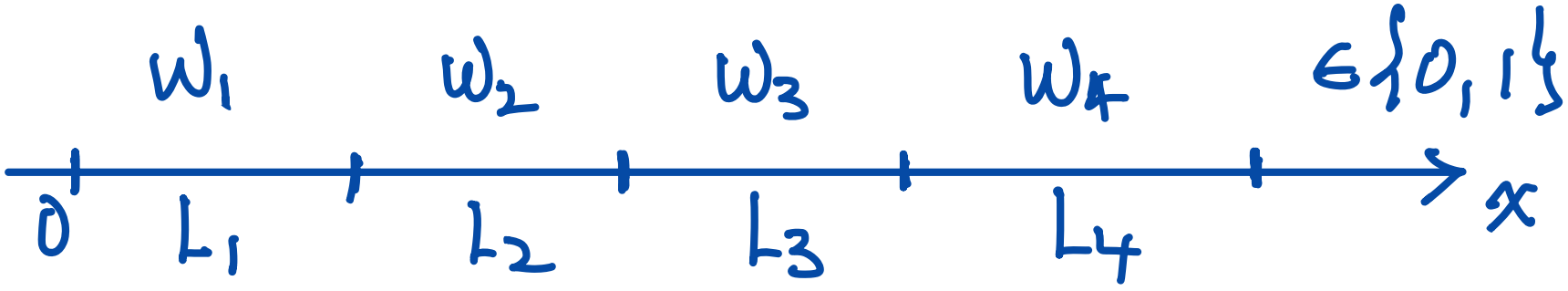


FIGURE 23.4. A piecewise linear function.

Nonlinear Objective Function [V] p.394



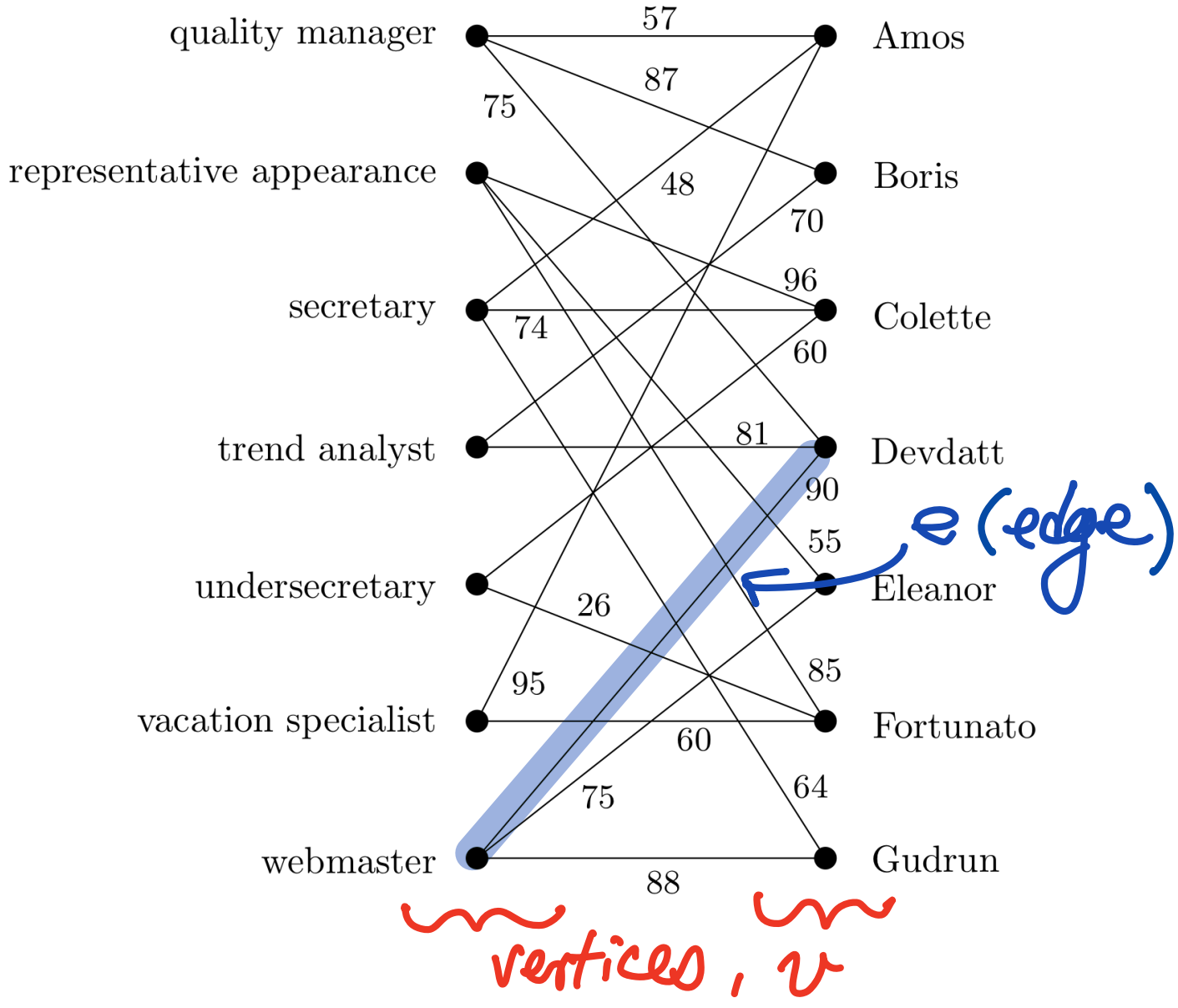
$$
\begin{aligned}
L_j w_j \le x_j \le L_j w_{j-1} \qquad & j = 1, 2, \ldots, k \\
w_0 = 1 & \\
w_j \in \{0, 1\} \qquad & j = 1, 2, \ldots, k \\
x_j \ge 0 \qquad & j = 1, 2, \ldots, k.
\end{aligned}
$$

Indeed, it follows from these constraints that $w_j \le w_{j-1}$ for $j = 1, 2, \ldots, k$. This inequality implies that once one of the w_j's is zero, then all the subsequent ones must be zero. If $w_j = w_{j-1} = 1$, the two-sided inequality on x_j reduces to $L_j \le x_j \le L_j$. That is, $x_j = L_j$. Similarly, if $w_j = w_{j-1} = 0$, then the two-sided inequality reduces to $x_j = 0$. The only other case is when $w_j = 0$ but $w_{j-1} = 1$. In this case, the two-sided inequality becomes $0 \le x_j \le L_j$. Therefore, in all cases, we get what we want. Now with this decomposition we can write the piecewise linear function as

$$K + c_1 x_1 + c_2 x_2 + \cdots + c_k x_k.$$

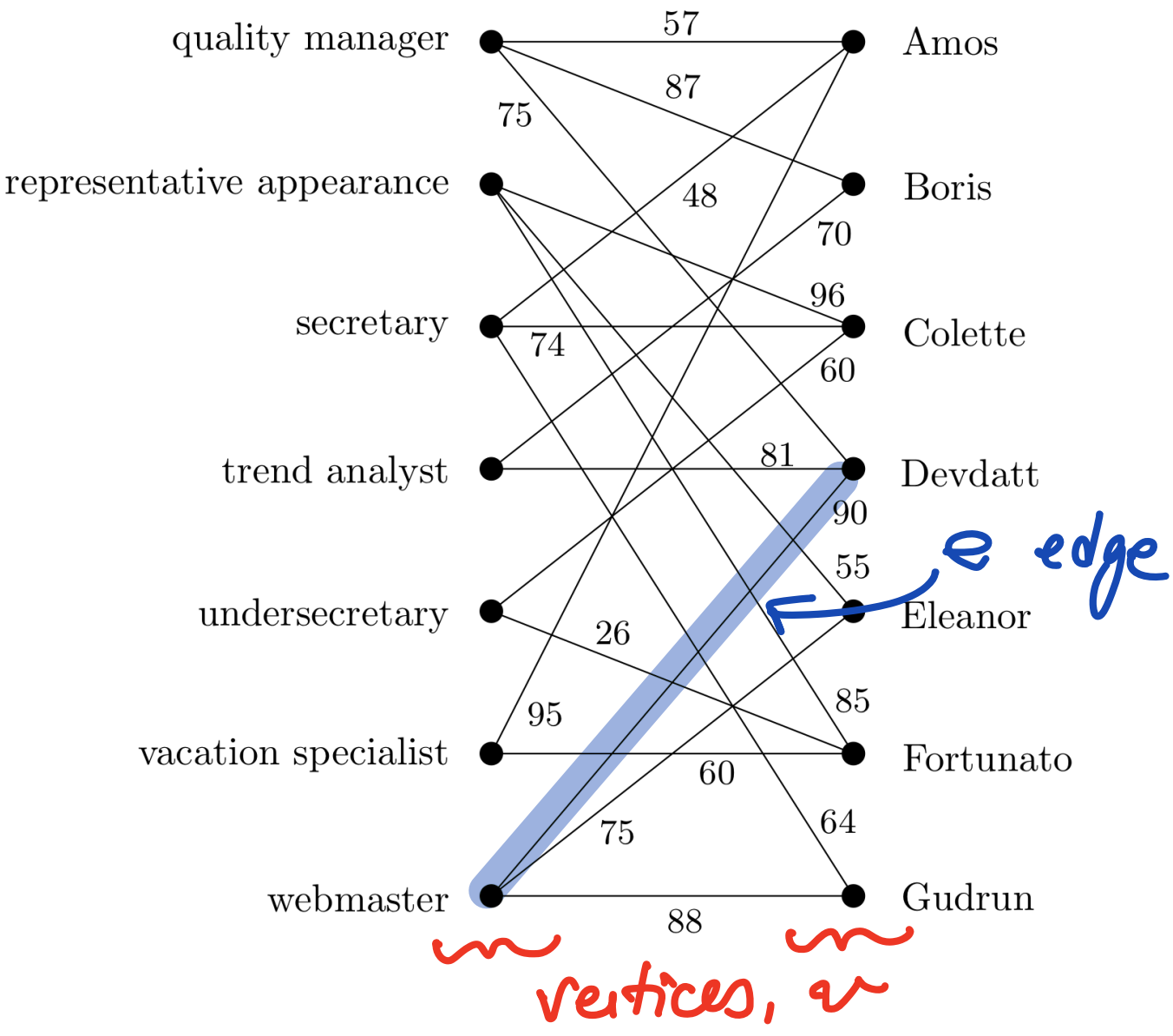
Integer Programming

[MG] p. 31 Maximum Weight Matching



Integer Programming

[MG] p. 31 Maximum Weight Matching



$$\max \sum_e w_e x_e$$

s.t. for any v,

$$\sum_{e,\, v \in e} x_e = 1$$

$$x_e \in \{0, 1\}$$

[v] p. 259

2. The Assignment Problem

Given a set $\mathcal{S}$ of m people, a set $\mathcal{D}$ of m tasks, and for each $i \in \mathcal{S}$, $j \in \mathcal{D}$ a cost c_{ij} associated with assigning person i to task j, the *assignment problem* is to assign each person to one and only one task in such a manner that each task gets covered by someone and the total cost of the assignments is minimized. If we let

$$x_{ij} = \begin{cases} 1 & \text{if person } i \text{ is assigned task } j, \\ 0 & \text{otherwise,} \end{cases}$$

then the objective function can be written as

$$\text{minimize} \sum_{i \in \mathcal{S}} \sum_{j \in \mathcal{D}} c_{ij} x_{ij}.$$

The constraint that each person is assigned exactly one task can be expressed simply as

$$\sum_{j \in \mathcal{D}} x_{ij} = 1, \qquad \text{for all } i \in \mathcal{S}.$$

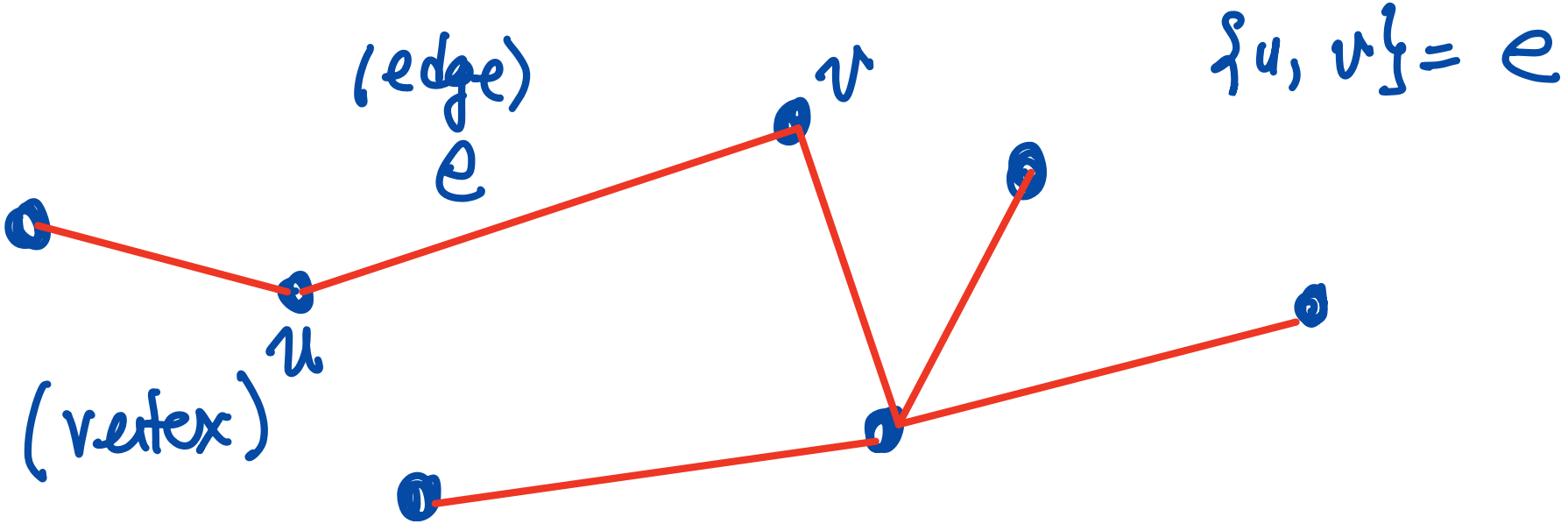
Also, the constraint that every task gets covered by someone is just

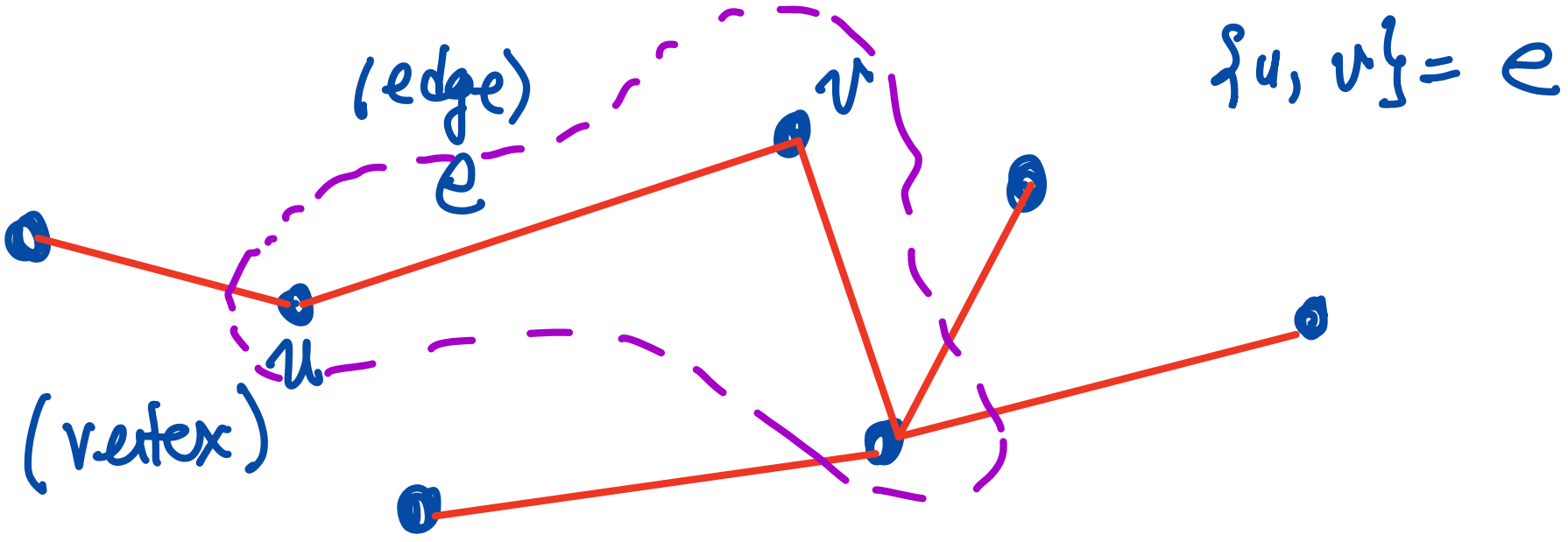
$$\sum_{i \in \mathcal{S}} x_{ij} = 1, \qquad \text{for all } j \in \mathcal{D}.$$

3.3 Minimum Vertex Cover

[MG, p. 37]

The Internet had been expanding rapidly in the Free Republic of West Mordor, and the government issued a regulation, purely in the interest of improved security of the citizens, that every data link connecting two computers must be equipped with a special device for gathering statistical data about the traffic. An operator of a part of the network has to attach the government's monitoring boxes to some of his computers so that each link has a monitored computer on at least one end. Which computers should get boxes so that the total price is minimum? Let us assume that there is a flat rate per box.





This problem can be written as an integer program:

$$\begin{array}{lll} \text{Minimize} & \sum_{v \in V} x_v & \\ \text{subject to} & x_u + x_v \geq 1 & \text{for every edge } \{u, v\} \in E \\ & x_v \in \{0, 1\} & \text{for all } v \in V. \end{array} \tag{3.2}$$

For every vertex v we have a variable x_v, which can attain values 0 or 1. The meaning of $x_v = 1$ is $v \in S$, and $x_v = 0$ means $v \notin S$. The constraint $x_u + x_v \geq 1$ guarantees that the edge $\{u, v\}$ has at least one vertex in S. The objective function is the size of S.

Knapsack Problem

[V] p.413

23.1 *Knapsack Problem.* Consider a picnicker who will be carrying a knapsack that holds a maximum amount b of "stuff." Suppose that our picnicker must decide what to take and what to leave behind. The jth thing that might be taken occupies a_j units of space in the knapsack and will bring c_j amount of "enjoyment." The knapsack problem then is to maximize enjoyment subject to the constraint that the stuff brought must fit into the knapsack:

$$\text{maximize} \quad \sum_{j=1}^{n} c_j x_j$$

$$\text{subject to} \quad \sum_{j=1}^{n} a_j x_j \leq b$$

$$x_j \in \{0, 1\} \qquad j = 1, 2, \ldots, n.$$

[C] p.201

In the usual notation, the knapsack problem is recorded as

$$\text{maximize} \quad \sum_{i=1}^{m} c_i x_i$$

$$\text{subject to} \quad \sum_{i=1}^{m} a_i x_i \leq b \tag{13.5}$$

$$x_i = \text{nonnegative integer} \qquad (i = 1, 2, \ldots, m).$$

Integer Programming

[MG] p. 148 Machine Scheduling

	Single B&W	Duplex B&W	Duplex Color
Master's thesis, 90 pages two-sided, 10 B&W copies	—	45 min	60 min
All the Best Deals flyer, 1 page one-sided, 10,000 B&W copies	2h 45 min	4h 10 min	5h 30 min
Buyer's Paradise flyer, 1 page one-sided, 10,000 B&W copies	2h 45 min	4h 10 min	5h 30 min
Obituary, 2 pages two-sided, 100 B&W copies	—	2 min	3 min
Party platform, 10 pages two-sided, 5,000 color copies	—	—	3h 30 min

Integer Programming

[MG] p. 148 Machine Scheduling

$M = \{1, 2, \ldots, m\}$ m - Machines

$J = \{1, 2, \ldots, n\}$ n - Jobs

d_{ij} = running time of Job j in Machine i

$$\min \quad t \quad \text{(total run time)}$$

$$\text{s.t.} \quad \sum_{i \in M} x_{ij} = 1 \qquad j \in J$$

$$\sum_{j \in J} d_{ij} x_{ij} \leq t$$

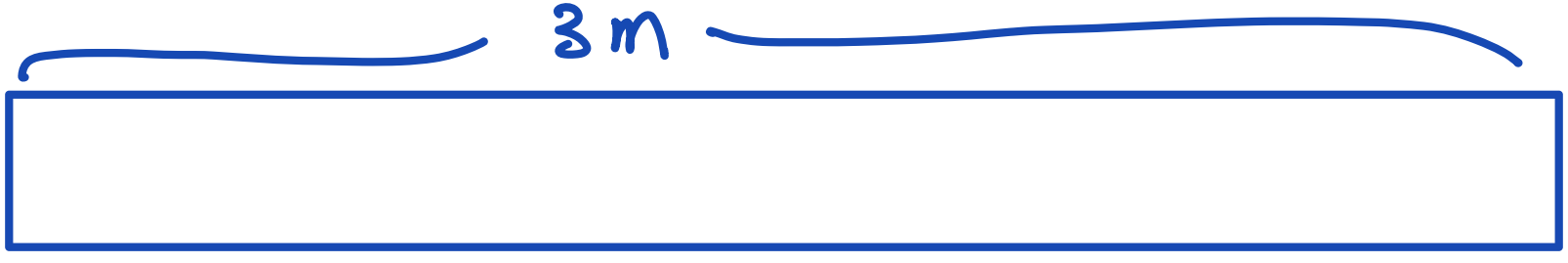
$$x_{ij} \in \{0, 1\}$$

Cutting Stock Problem [MG p. 26]

A paper mill manufactures rolls of paper of a standard width 3 meters. But customers want to buy paper rolls of shorter width, and the mill has to cut such rolls from the 3 m rolls. One 3 m roll can be cut, for instance, into two rolls 93 cm wide, one roll of width 108 cm, and a rest of 6 cm (which goes to waste).

Let us consider an order of

- 97 rolls of width 135 cm,
- 610 rolls of width 108 cm,
- 395 rolls of width 93 cm, and
- 211 rolls of width 42 cm.



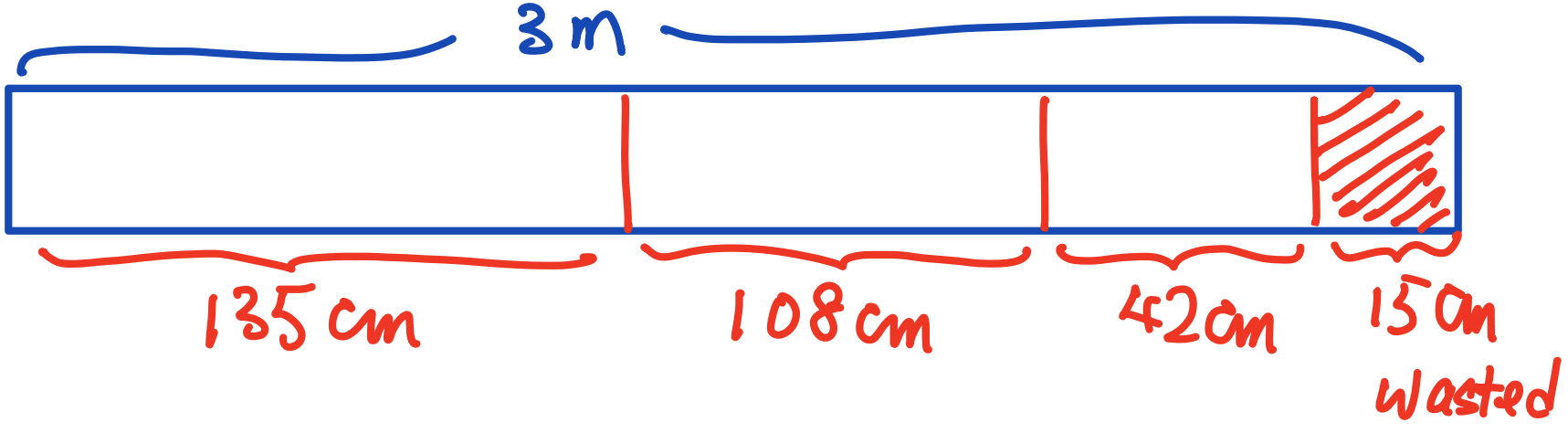
Cutting Stock Problem [MG p. 26]

A paper mill manufactures rolls of paper of a standard width 3 meters. But customers want to buy paper rolls of shorter width, and the mill has to cut such rolls from the 3 m rolls. One 3 m roll can be cut, for instance, into two rolls 93 cm wide, one roll of width 108 cm, and a rest of 6 cm (which goes to waste).

Let us consider an order of

- 97 rolls of width 135 cm,
- 610 rolls of width 108 cm,
- 395 rolls of width 93 cm, and
- 211 rolls of width 42 cm.

e.g.



Cutting Stock Problem [MG p. 26]

In order to engage linear programming one has to be generous in introducing variables. We write down all of the requested widths: 135 cm, 108 cm, 93 cm, and 42 cm. Then we list all possibilities of cutting a 3 m paper roll into rolls of some of these widths (we need to consider only possibilities for which the wasted piece is shorter than 42 cm):

P1:	2×135	P7:	$108 + 93 + 2 \times 42$
P2:	$135 + 108 + 42$	P8:	$108 + 4 \times 42$
P3:	$135 + 93 + 42$	P9:	3×93
P4:	$135 + 3 \times 42$	P10:	$2 \times 93 + 2 \times 42$
P5:	$2 \times 108 + 2 \times 42$	P11:	$93 + 4 \times 42$
P6:	$108 + 2 \times 93$	P12:	7×42

$$258 \leq 135 n_1 + 108 n_2 + 93 n_3 + 42 n_4 \leq 300$$

$$0 \leq n_i, \text{ integers}$$

Cutting Stock Problem [MG p. 26]

In order to engage linear programming one has to be generous in introducing variables. We write down all of the requested widths: 135 cm, 108 cm, 93 cm, and 42 cm. Then we list all possibilities of cutting a 3 m paper roll into rolls of some of these widths (we need to consider only possibilities for which the wasted piece is shorter than 42 cm):

P1:	2×135	P7:	$108 + 93 + 2 \times 42$
P2:	$135 + 108 + 42$	P8:	$108 + 4 \times 42$
P3:	$135 + 93 + 42$	P9:	3×93
P4:	$135 + 3 \times 42$	P10:	$2 \times 93 + 2 \times 42$
P5:	$2 \times 108 + 2 \times 42$	P11:	$93 + 4 \times 42$
P6:	$108 + 2 \times 93$	P12:	7×42

$x_i = \#\text{ of } P_i, \quad i = 1, \ldots, 12$

$x_i \geq 0$, integers

$\min \quad x_1 + x_2 + \cdots + x_{12}$

Cutting Stock Problem [MG p. 26]

In order to engage linear programming one has to be generous in introducing variables. We write down all of the requested widths: 135 cm, 108 cm, 93 cm, and 42 cm. Then we list all possibilities of cutting a 3 m paper roll into rolls of some of these widths (we need to consider only possibilities for which the wasted piece is shorter than 42 cm):

P1:	2×135	P7:	$108 + 93 + 2 \times 42$
P2:	$135 + 108 + 42$	P8:	$108 + 4 \times 42$
P3:	$135 + 93 + 42$	P9:	3×93
P4:	$135 + 3 \times 42$	P10:	$2 \times 93 + 2 \times 42$
P5:	$2 \times 108 + 2 \times 42$	P11:	$93 + 4 \times 42$
P6:	$108 + 2 \times 93$	P12:	7×42

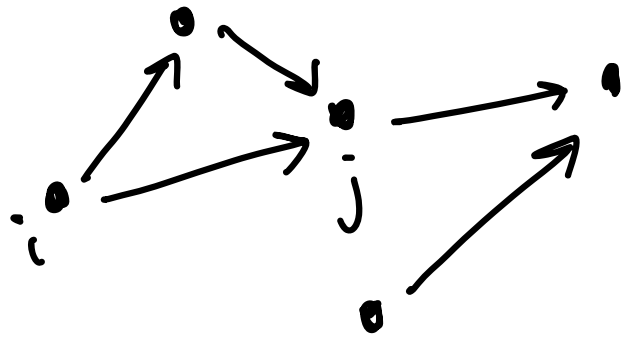
$$2x_1 + x_2 + x_3 + x_4 \geq 97$$
$$x_2 + 2x_5 + x_6 + x_7 + x_8 \geq 610$$
$$x_3 + 2x_6 + x_7 + 2x_9 + x_{11} \geq 395$$
$$x_2 + x_3 + x_4 + 2x_5 + 2x_7 + 4x_8 + 2x_{10} + 4x_{11} + 7x_{12} \geq 211$$

Network Flow ([V] Chapter 14)

N = set of nodes $\{i\}$

A = set of (directed) arcs

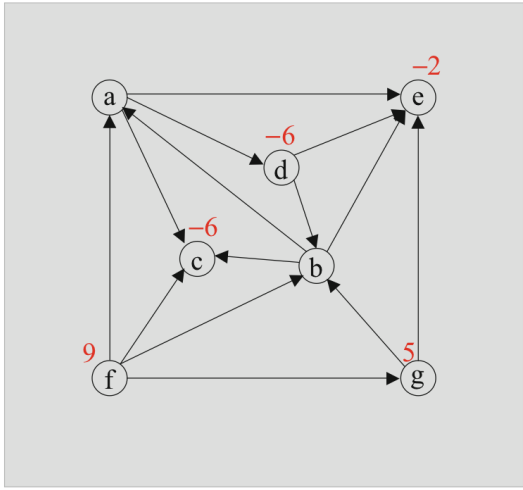
$\subseteq \{(i,j) : i,j \in N\}$



Network = (N, A)

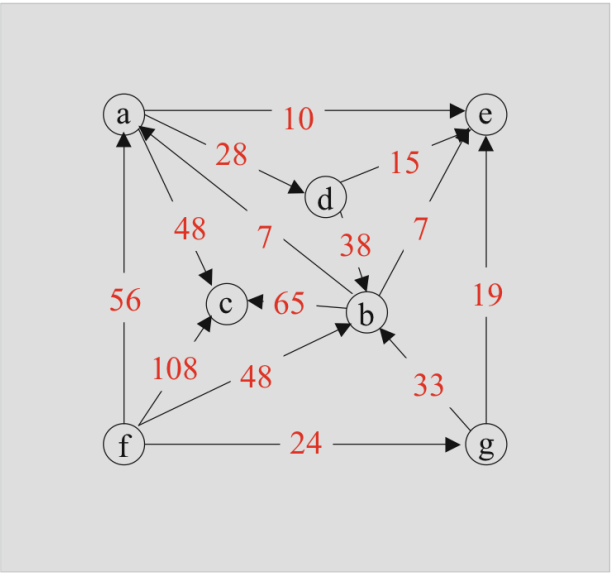
graph (or digraph)

Network Flow ([V] Chapter 14)



b_i

FIGURE 14.1. A network having 7 nodes and 14 arcs. The numbers written next to the nodes denote the supply at the node (negative values indicate demands; missing values indicate no supply or demand).



c_{ij}

FIGURE 14.2. The costs on the arcs for the network in Figure 14.1.

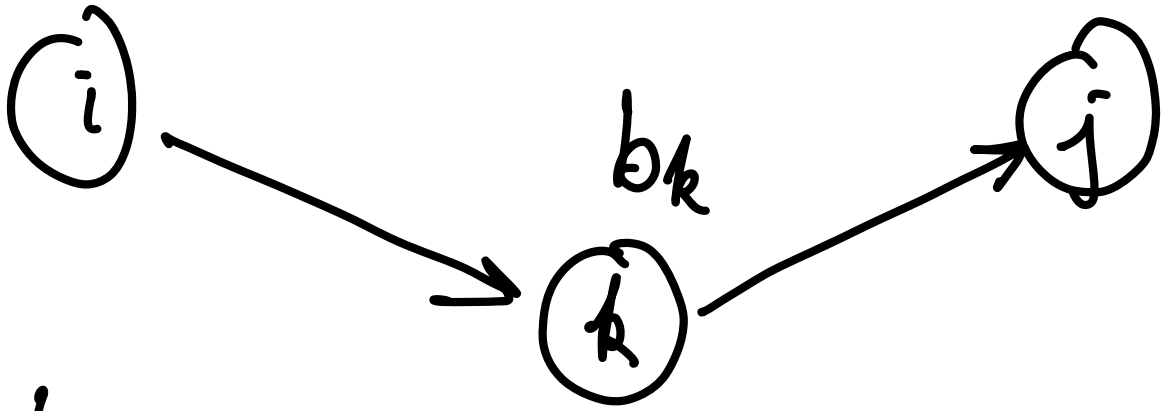
(1) $b_i > 0$ (supply) ; $b_i < 0$ (demand) $\quad \sum_{i \in N} b_i = 0$

(2) x_{ij} = amount transported from i to j

(3) c_{ij} = cost of transportation from i to j

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

Network Flow ([V] Chapter 14)



Balanced eqn at node k:

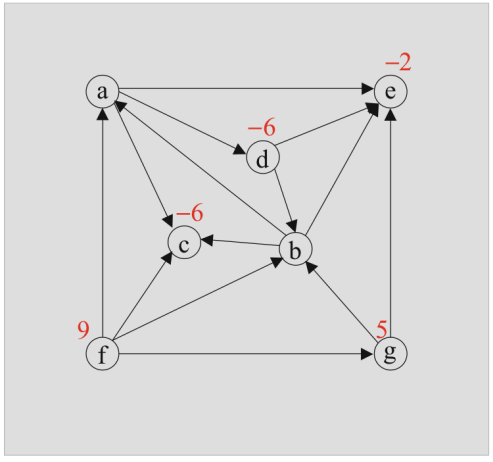
$$\sum_i x_{ik} + b_k = \sum_j x_{kj}$$

$$\sum_i x_{ik} - \sum_j x_{kj} = -b_k$$

$$\min \quad c^T x$$

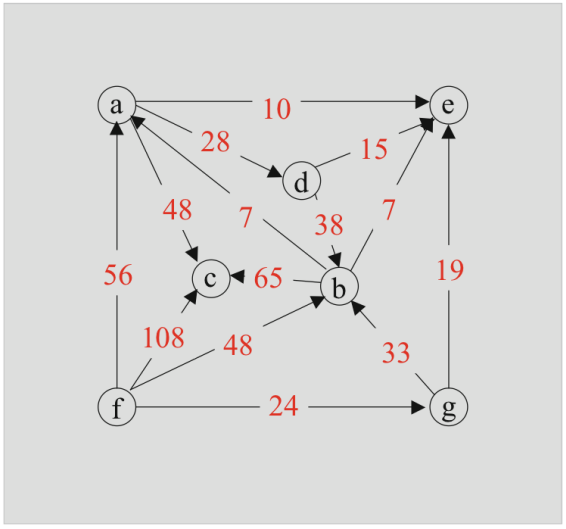
$$Ax = -b, \quad x \geq 0$$

Network Flow ([V] Chapter 14)



bi

FIGURE 14.1. A network having 7 nodes and 14 arcs. The numbers written next to the nodes denote the supply at the node (negative values indicate demands; missing values indicate no supply or demand).



Cij

FIGURE 14.2. The costs on the arcs for the network in Figure 14.1.

arcs

incidence matrix

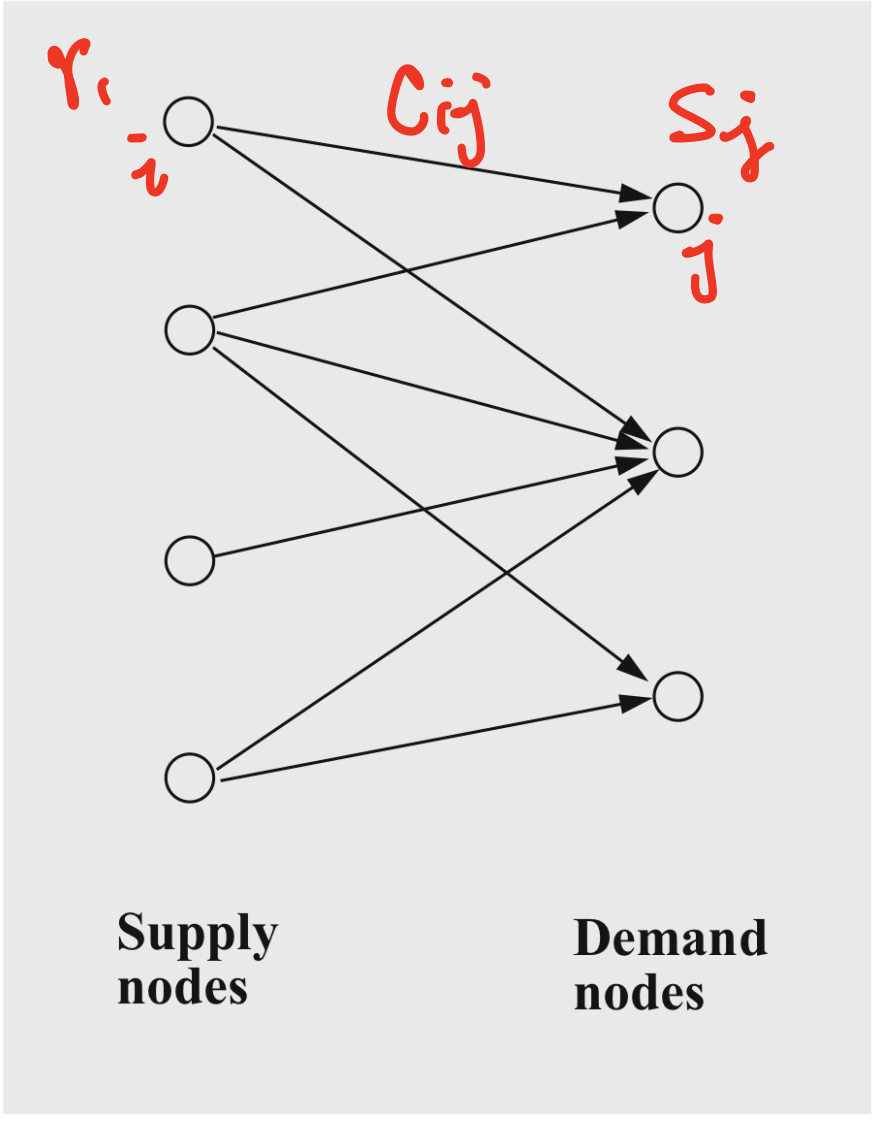
$$x^T = \begin{bmatrix} x_{ac} & x_{ad} & x_{ae} & x_{ba} & x_{bc} & x_{be} & x_{db} & x_{de} & x_{fa} & x_{fb} & x_{fc} & x_{fg} & x_{gb} & x_{ge} \end{bmatrix},$$

(a)
(b)
(c)
(d)
(e)
(f)
(g)

$$A = \begin{bmatrix} -1 & -1 & -1 & 1 & & & & & 1 & & & & & \\ & & & -1 & -1 & -1 & 1 & & & 1 & & & 1 & \\ 1 & & & & 1 & & & & & & 1 & & & \\ & 1 & & & & & -1 & -1 & & & & & & \\ & & 1 & & & 1 & & 1 & & & & & & 1 \\ & & & & & & & & -1 & -1 & -1 & -1 & & \\ & & & & & & & & & & & 1 & -1 & -1 \end{bmatrix}, \quad b = \begin{bmatrix} 0 \\ 0 \\ -6 \\ -6 \\ -2 \\ 9 \\ 5 \end{bmatrix}$$

$$c^T = \begin{bmatrix} 48 & 28 & 10 & 7 & 65 & 7 & 38 & 15 & 56 & 48 & 108 & 24 & 33 & 19 \end{bmatrix}.$$

Transportation Problem [V] p. 258



$$\text{minimize} \quad \sum_{i \in \mathcal{S}} \sum_{j \in \mathcal{D}} c_{ij} x_{ij}$$

$$\text{subject to} \quad \sum_{j \in \mathcal{D}} x_{ij} = r_i \qquad i \in \mathcal{S}$$

$$\sum_{i \in \mathcal{S}} x_{ij} = s_j \qquad j \in \mathcal{D}$$

$$x_{ij} \geq 0 \qquad i \in \mathcal{S}, j \in \mathcal{D}.$$

(bipartite graph)

A Miracle

The matrices for assignment, network flow (and transportation) problems are totally uni-modular

and hence

given integer supplies and demands, the solutions are automatically integers.