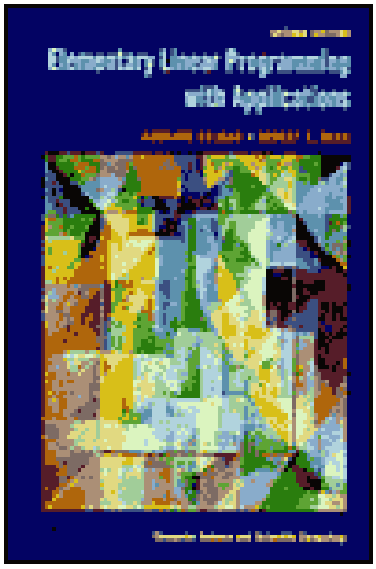
Elementary Linear Programming with Applications

by Bernard Kolman, Robert E. Beck



- Textbook Hardcover - REV
- ISBN: 012417910X; ISBN-13: 9780124179103
- Format: Textbook Hardcover, 449pp
- Publisher: Elsevier Science & Technology Books
- Pub. Date: June 1995

Table of Contents

1

Introduction to Linear Programming

THIS CHAPTER AND the next two, which represent the heart of this book, introduce the basic ideas and techniques of linear programming. This area of applied mathematics was developed in the late 1940s to solve a number of resource allocation problems for the federal government. It has become an essential tool in operations research and has been applied to a remarkably varied number of real problems, producing enormous savings in money and resources. In this chapter we first introduce the linear programming problem and then discuss a simple geometric solution for small problems. Finally, we connect the algebraic and geometric descriptions of the solutions of a linear programming problem.

1.1 THE LINEAR PROGRAMMING PROBLEM

We start by giving several examples of linear programming problems.

EXAMPLE 1 (ACTIVITY ANALYSIS OR PRODUCT MIX). A lumber mill saws both finish-grade and construction-grade boards from the logs that it receives. Suppose that it takes 2 hr to rough-saw each 1000 board feet of the finish-grade boards and 5 hr to plane each 1000 board feet of these boards. Suppose also that it takes 2 hr to rough-saw each 1000 board feet of the construction-grade boards, but it takes only 3 hr to plane each 1000 board feet of these boards. The saw is available 8 hr per day, and the plane is available 15 hr per day. If the profit on each 1000 board feet of finish-grade boards is \$120 and the profit on each 1000 board feet of construction-grade boards is \$100, how many board feet of each type of lumber should be sawed to maximize the profit?

MATHEMATICAL MODEL. Let x and y denote the amount of finish-grade and construction-grade lumber, respectively, to be sawed per day. Let the units of x and y be thousands of board feet. The number of hours required daily for the saw is

$$2x + 2y.$$

Since only 8 hours are available daily, x and y must satisfy the inequality

$$2x + 2y \leq 8.$$

Similarly, the number of hours required for the plane is

$$5x + 3y,$$

so x and y must satisfy

$$5x + 3y \leq 15.$$

Of course, we must also have

$$x \geq 0 \quad \text{and} \quad y \geq 0.$$

The profit (in dollars) to be maximized is given by

$$z = 120x + 100y.$$

Thus, our mathematical model is:

Find values of x and y that will

$$\text{maximize} \quad z = 120x + 100y$$

subject to the restrictions

$$2x + 2y \leq 8$$
$$5x + 3y \leq 15$$
$$x \geq 0$$
$$y \geq 0.$$

△

EXAMPLE 2 (THE DIET PROBLEM). A nutritionist is planning a menu consisting of two main foods A and B. Each ounce of A contains 2 units

of fat, 1 unit of carbohydrates, and 4 units of protein. Each ounce of B contains 3 units of fat, 3 units of carbohydrates, and 3 units of protein. The nutritionist wants the meal to provide at least 18 units of fat, at least 12 units of carbohydrates, and at least 24 units of protein. If an ounce of A costs 20 cents and an ounce of B costs 25 cents, how many ounces of each food should be served to minimize the cost of the meal yet satisfy the nutritionist's requirements?

MATHEMATICAL MODEL. Let x and y denote the number of ounces of foods A and B, respectively, that are served. The number of units of fat contained in the meal is

$$2x + 3y,$$

so that x and y have to satisfy the inequality

$$2x + 3y \geq 18.$$

Similarly, to meet the nutritionist's requirements for carbohydrate and protein, we must have x and y satisfy

$$x + 3y \geq 12$$

and

$$4x + 3y \geq 24.$$

Of course, we also require that

$$x \geq 0 \quad \text{and} \quad y \geq 0.$$

The cost of the meal, which is to be minimized, is

$$z = 20x + 25y.$$

Thus, our mathematical model is:

Find values of x and y that will

$$\begin{aligned} &\text{minimize} \quad z = 20x + 25y \\ &\text{subject to the restrictions} \\ &2x + 3y \geq 18 \\ &x + 3y \geq 12 \\ &4x + 3y \geq 24 \\ &x \geq 0 \\ &y \geq 0. \end{aligned}$$

△

EXAMPLE 3 (THE TRANSPORTATION PROBLEM). A manufacturer of sheet polyethylene has two plants, one located in Salt Lake City and the other located in Denver. There are three distributing warehouses, one in Los Angeles, another in Chicago, and the third in New York City. The Salt

Lake City plant can supply 120 tons of the product per week, whereas the Denver plant can supply 140 tons per week. The Los Angeles warehouse needs 100 tons weekly to meet its demand, the Chicago warehouse needs 60 tons weekly, and the New York City warehouse needs 80 tons weekly. The following tables gives the shipping cost (in dollars) per ton of the product:

From	*To*		
	Los Angeles	Chicago	New York City
Salt Lake City	5	7	9
Denver	6	7	10

How many tons of polyethylene should be shipped from each plant to each warehouse to minimize the total shipping cost while meeting the demand?

MATHEMATICAL MODEL. Let P_1 and P_2 denote the plants in Salt Lake City and in Denver, respectively. Let W_1, W_2, and W_3 denote the warehouses in Los Angeles, Chicago, and New York City, respectively. Let

$$x_{ij} = \text{number of tons shipped from } P_i \text{ to } W_j$$

$$c_{ij} = \text{cost of shipping 1 ton from } P_i \text{ to } W_j$$

for $i = 1, 2$ and $j = 1, 2, 3$. The total amount of polyethylene sent from P_1 is

$$x_{11} + x_{12} + x_{13}.$$

Since P_1 can supply only 120 tons, we must have

$$x_{11} + x_{12} + x_{13} \leq 120.$$

Similarly, since P_2 can supply only 140 tons, we must have

$$x_{21} + x_{22} + x_{23} \leq 140.$$

The total amount of polyethylene received at W_1 is

$$x_{11} + x_{21}.$$

Since the demand at W_1 is 100 tons, we would like to have

$$x_{11} + x_{21} \geq 100.$$

Similarly, since the demands at W_2 and W_3 are 60 and 80 tons, respectively, we would like to have

$$x_{12} + x_{22} \geq 60$$

and

$$x_{13} + x_{23} \geq 80.$$

Of course, we must also have

$$x_{ij} \geq 0 \quad \text{for} \quad i = 1, 2 \quad \text{and} \quad j = 1, 2, 3.$$

The total transportation cost, which we want to minimize, is

$$z = c_{11}x_{11} + c_{12}x_{12} + c_{13}x_{13} + c_{21}x_{21} + c_{22}x_{22} + c_{23}x_{23}.$$

Thus, our mathematical model is:

Find values of x_{11}, x_{12}, x_{13}, x_{21}, x_{22}, and x_{23} that will

$$\text{minimize} \quad z = \sum_{i=1}^{2} \sum_{j=1}^{3} c_{ij}x_{ij}$$

subject to the restrictions

$$\sum_{j=1}^{3} x_{ij} \leq s_i, \quad i = 1, 2$$

$$\sum_{i=1}^{2} x_{ij} \geq d_j, \quad j = 1, 2, 3$$

$$x_{ij} \geq 0, \quad i = 1, 2 \quad \text{and} \quad j = 1, 2, 3$$

where available supplies are

$$s_1 = 120 \quad \text{and} \quad s_2 = 140$$

and where the required demands are

$$d_1 = 100, \quad d_2 = 60, \quad \text{and} \quad d_3 = 80. \qquad \triangle$$

EXAMPLE 4 (A BLENDING PROBLEM). A manufacturer of artificial sweetener blends 14 kg of saccharin and 18 kg of dextrose to prepare two new products: SWEET and LO-SUGAR. Each kilogram of SWEET contains 0.4 kg of dextrose and 0.2 kg of saccharin, whereas each kilogram of LO-SUGAR contains 0.3 kg of dextrose and 0.4 kg of saccharin. If the profit on each kilogram of SWEET is 20 cents and the profit on each kilogram of LO-SUGAR is 30 cents, how many kilograms of each product should be made to maximize the profit?

MATHEMATICAL MODEL. Let x and y denote the number of kilograms of SWEET and LO-SUGAR, respectively, being made. The number of kilograms of dextrose being used is

$$0.4x + 0.3y,$$

so that we must have

$$0.4x + 0.3y \leq 18.$$

Similarly, the number of kilograms of saccharin being used is

$$0.2x + 0.4y,$$

so that we must have

$$0.2x + 0.4y \leq 14.$$

Of course, we also require that

$$x \geq 0 \quad \text{and} \quad y \geq 0.$$

The total profit (in cents), which we seek to maximize, is

$$z = 20x + 30y.$$

Thus, our mathematical model is:

Find values of x and y that will

$$\begin{aligned} &\text{maximize} \quad z = 20x + 30y \\ &\text{subject to the restrictions} \\ &0.4x + 0.3y \leq 18 \\ &0.2x + 0.4y \leq 14 \\ &x \geq 0 \\ &y \geq 0. \end{aligned}$$

△

EXAMPLE 5 (A FINANCIAL PROBLEM). Suppose that the financial advisor of a university's endowment fund must invest *exactly* \$100,000 in two types of securities: bond AAA, paying a dividend of 7%, and stock BB, paying a dividend of 9%. The advisor has been told that no more than \$30,000 can be invested in stock BB, whereas the amount invested in bond AAA must be at least twice the amount invested in stock BB. How much should be invested in each security to maximize the university's return?

MATHEMATICAL MODEL. Let x and y denote the amounts invested in bond AAA and stock BB, respectively. We must then have

$$\begin{aligned} x + y &= 100{,}000 \\ x &\geq 2y \\ y &\leq 30{,}000. \end{aligned}$$

Of course, we also require that

$$x \geq 0 \quad \text{and} \quad y \geq 0.$$

The return to the university, which we seek to maximize, is

$$z = 0.07x + 0.09y.$$

Thus, our mathematical model is:

Find values of x and y that will

$$\begin{aligned} &\text{maximize} \quad z = 0.07x + 0.09y \\ &\text{subject to the restrictions} \\ &x + y = 100{,}000 \\ &x - 2y \geq 0 \\ &y \leq 30{,}000 \\ &x \geq 0 \\ &y \geq 0. \end{aligned}$$

△

Following the form of the previous examples, the **general linear programming problem** can be stated as follows:

Find values of $x_1, x_2, \ldots, x_n$ that will

$$\text{maximize or minimize} \quad z = c_1x_1 + c_2x_2 + \cdots + c_nx_n \tag{1}$$

subject to the restrictions

$$\left.\begin{array}{l} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n \leq (\geq)(=)b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n \leq (\geq)(=)b_2 \\ \quad\vdots \qquad\quad \vdots \qquad\qquad \vdots \qquad\quad \vdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n \leq (\geq)(=)b_m \end{array}\right\}, \tag{2}$$

where in each inequality in (2) one and only one of the symbols, $\leq$, $\geq$, $=$ occurs. The linear function in (1) is called the **objective function**. The equalities or inequalities in (2) are called **constraints**. Note that the left-hand sides of all the inequalities or equalities in (2) are linear functions of the variables $x_1, x_2, \ldots, x_n$, just as the objective function is. A problem in which not all the constraints or the objective function are linear functions of the variables is a nonlinear programming problem. Such problems are discussed in more advanced texts.

We shall say that a linear programming problem is in **standard form** if it is in the following form:

Find values of $x_1, x_2, \ldots, x_n$ that will

$$\text{maximize} \quad z = c_1x_1 + c_2x_2 + \cdots + c_nx_n \tag{3}$$

subject to the constraints

$$\left.\begin{array}{l} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n \leq b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n \leq b_2 \\ \quad\vdots \qquad\quad \vdots \qquad\qquad \vdots \quad \vdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n \leq b_m \end{array}\right\} \tag{4}$$

$$x_j \geq 0, \quad j = 1, 2, \ldots, n. \tag{5}$$

Examples 1 and 4 are in standard form. The other examples are not. Why?

We shall say that a linear programming problem is in **canonical form** if it is in the following form:

Find values of $x_1, x_2, \ldots, x_s$ that will

$$\text{maximize} \quad z = c_1x_1 + c_2x_2 + \cdots + c_sx_s$$

subject to the constraints

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1s}x_s &= b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2s}x_s &= b_2 \\ \vdots \quad\quad \vdots \quad\quad\quad\quad \vdots \quad \vdots & \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{ms}x_s &= b_m \\ x_j \geq 0, \quad j = 1, 2, \ldots, s. & \end{aligned}$$

EXAMPLE 6. The following linear programming problem is in canonical form:

$$\begin{aligned} &\text{Maximize} \quad z = 3x + 2y + 3u - 4v \\ &\text{subject to the constraints} \\ &\qquad 2x + y + 2u - v = 4 \\ &\qquad 5x + 3y \qquad - 2v = 15 \\ &x \geq 0, \quad y \geq 0, \quad u \geq 0, \quad v \geq 0. \end{aligned}$$

△

Some other authors use different names for what we call standard and canonical linear programming problems. Some also require that all the variables be nonnegative in a linear programming problem. The reader should carefully check the definitions when referring to other books or papers.

EXAMPLE 7. The following linear programming problems are neither in standard form nor in canonical form. Why?

(a)

$$\begin{aligned} &\text{Minimize } z = 3x + 2y \\ &\text{subject to the constraints} \\ &\qquad 2x + y \leq 4 \\ &\qquad 3x - 2y \leq 6 \\ &\qquad x \geq 0, \quad y \geq 0. \end{aligned}$$

(b)

$$\begin{aligned} &\text{Maximize } z = 2x_1 + 3x_2 + 4x_3 \\ &\text{subject to the constraints} \\ &\qquad 3x_1 + 2x_2 - 3x_3 \leq 4 \\ &\qquad 2x_1 + 3x_2 + 2x_3 \leq 6 \\ &\qquad 3x_1 - x_2 + 2x_3 \geq -8 \\ &\qquad x_1 \geq 0, \quad x_2 \geq 0, \quad x_3 \geq 0. \end{aligned}$$

(c) Maximize $z = 3x + 2y + 3v - 2w$
subject to the constraints
$$\begin{aligned} 2x + 6y + 2v - 4w &= 7 \\ 3x + 2y - 5v + w &= 8 \\ 6x + 7y + 2v + 5w &\leq 4 \\ x \geq 0, \quad y \geq 0, \quad v \geq 0, \quad w &\geq 0. \end{aligned}$$

(d) Minimize $z = 2x + 5y + u + v + 4w$
subject to the constraints
$$\begin{aligned} 3x + 2y - u + 2w &= 4 \\ 4x + 5y + 3u + 2v &= 7 \\ x \geq 0, \quad y \geq 0, \quad u \geq 0, \quad v \geq 0, \quad w &\geq 0. \end{aligned}$$

(e) Maximize $z = 2x + 5y$
subject to the constraints
$$\begin{aligned} 3x + 2y &\leq 6 \\ 2x + 9y &\leq 8 \\ x &\geq 0. \end{aligned}$$

(f) Minimize $z = 2x_1 + 3x_2 + x_3$
subject to the constraints
$$\begin{aligned} 2x_1 + x_2 - x_3 &= 4 \\ 3x_1 + 2x_2 + x_3 &= 8 \\ x_1 - x_2 &= 6 \\ x_1 \geq 0, \quad x_2 &\geq 0. \end{aligned}$$
△

We shall now show that every linear programming problem that has unconstrained variables can be solved by solving a corresponding linear programming problem in which all the variables are constrained to be nonnegative. Moreover, we show that every linear programming problem can be formulated as a corresponding standard linear programming problem or as a corresponding canonical linear programming problem. That is, we can show that there is a standard linear programming problem (or canonical linear program problem) whose solution determines a solution to the given arbitrary linear programming problem.

Minimization Problem as a Maximization Problem

Every minimization problem can be viewed as a maximization problem and conversely. This can be seen from the observation that

$$\min \sum_{i=1}^{n} c_i x_i = -\max\left(-\sum_{i=1}^{n} c_i x_i \right).$$

That is, to minimize the objective function we could maximize its negative instead and then change the sign of the answer.

Reversing an Inequality

If we multiply the inequality

$$k_1x_1 + k_2x_2 + \cdots + k_nx_n \geq b$$

by -1, we obtain the inequality

$$-k_1x_1 - k_2x_2 - \cdots - k_nx_n \leq -b.$$

EXAMPLE 8. Consider the linear programming problem given in Example 7b. If we multiply the third constant,

$$3x_1 - x_2 + 2x_3 \geq -8,$$

by -1, we obtain the equivalent linear programming problem:

$$\begin{aligned} &\text{Maximize} \quad z = 2x_1 + 3x_2 + 4x_3 \\ &\text{subject to} \\ &\qquad 3x_1 + 2x_2 - 3x_3 \leq 4 \\ &\qquad 2x_1 + 3x_2 + 2x_3 \leq 6 \\ &\qquad -3x_1 + x_2 - 2x_3 \leq 8 \\ &\qquad x_1 \geq 0, \quad x_2 \geq 0, \quad x_3 \geq 0, \end{aligned}$$

which is in standard form. △

Changing an Equality to an Inequality

Observe that we can write the equation $x = 6$ as the pair of inequalities $x \leq 6$ and $x \geq 6$ and hence as the pair $x \leq 6$ and $-x \leq -6$. In the general case the equation

$$\sum_{j=1}^{n} a_{ij}x_j = b_i$$

can be written as the pair of inequalities

$$\sum_{j=1}^{n} a_{ij}x_j \leq b_i$$

$$\sum_{j=1}^{n} -a_{ij}x_j \leq -b_i.$$

EXAMPLE 9. Consider the linear programming problem given in Example 7c. It contains the two equality constraints

$$\begin{aligned} 2x + 6y + 2v - 4w &= 7 \\ 3x + 2y - 5v + w &= 8. \end{aligned}$$

These may be written as the equivalent four inequalities

$$\begin{aligned} 2x + 6y + 2v - 4w &\leq 7 \\ 2x + 6y + 2v - 4w &\geq 7 \\ 3x + 2y - 5v + w &\leq 8 \\ 3x + 2y - 5v + w &\geq 8. \end{aligned}$$

Thus, we obtain the equivalent linear programming problem:

$$\begin{aligned} &\text{Maximize} \quad z = 3x + 2y + 3v - 2w \\ &\text{subject to} \\ &\qquad 2x + 6y + 2v - 4w \leq 7 \\ &\qquad -2x - 6y - 2v + 4w \leq -7 \\ &\qquad 3x + 2y - 5v + w \leq 8 \\ &\qquad -3x - 2y + 5v - w \leq -8 \\ &\qquad 6x + 7y + 2v + 5w \leq 4 \\ &\qquad x \geq 0, \quad y \geq 0, \quad v \geq 0, \quad w \geq 0, \end{aligned}$$

which is in standard form. △

Unconstrained Variables

The problems in Examples 7e and 7f have variables that are not constrained to be nonnegative. Suppose that x_j is not constrained to be nonnegative. We replace x_j with two new variables, x_j^+ and x_j^-, letting

$$x_j = x_j^+ - x_j^-,$$

where $x_j^+ \geq 0$ and $x_j^- \geq 0$. That is, any number is the difference of two nonnegative numbers. In this manner we may introduce constraints on unconstrained variables.

EXAMPLE 10. Consider the problem in Example 7e. Letting $y = y^+ - y^-$, our problem becomes the following linear programming problem:

$$\begin{aligned} &\text{Maximize} \quad z = 2x + 5y^+ - 5y^- \\ &\text{subject to} \\ &\qquad 3x + 2y^+ - 2y^- \leq 6 \\ &\qquad 2x + 9y^+ - 9y^- \leq 8 \\ &\qquad x \geq 0, \quad y^+ \geq 0, \quad y^- \geq 0, \end{aligned}$$

which is in standard form. △

EXAMPLE 11. The problem in Example 7f can be converted to a maximization problem. We also let $x_3 = x_3^+ - x_3^-$. With these changes we

obtain the following problem:

$$\begin{aligned} &\text{Maximize} \quad z = -2x_1 - 3x_2 - x_3^+ + x_3^- \\ &\text{subject to} \\ &\qquad 2x_1 + x_2 - x_3^+ + x_3^- = 4 \\ &\qquad 3x_1 + 2x_2 + x_3^+ - x_3^- = 8 \\ &\qquad x_1 - x_2 = 6 \\ &x_1 \geq 0, \quad x_2 \geq 0, \quad x_3^+ \geq 0, \quad x_3^- \geq 0, \end{aligned}$$

which is in canonical form. △

We have thus shown that every linear programming problem that is not in standard form can be transformed into an equivalent linear programming problem that is in standard form.

Scaling

It is not difficult to see that if both sides of one or more constraints of a linear programming problem are multiplied by constants, then the optimal solution to the new problem is identical to the optical solution to the given problem. This technique can be used to make all coefficients in a linear programming problem approximately the same size. This method, called **scaling**, will be discussed further in Section 3.6.

A diagrammatic representation of the various types of linear programming problems is given in Figure 1.1.

In Section 1.2 we will show how to convert a linear programming problem in standard form to one in canonical form. Thus, any linear programming problem can be put in either standard form or canonical form.

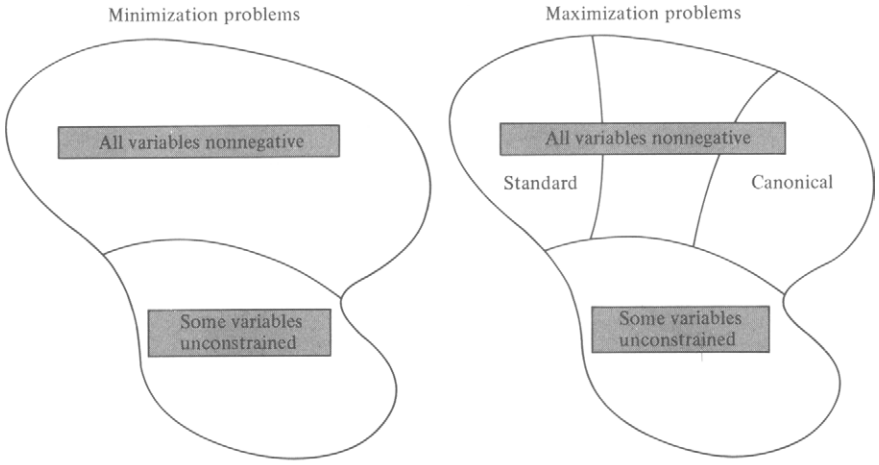


FIGURE 1.1

1.1 EXERCISES

In Exercises 1–11 set up a linear programming model of the situation described. Determine if the model is in standard form. If it is not, state what must be changed to put the model into standard form.

1. **Blending problem.** A new rose dust is being prepared by using two available products: PEST and BUG. Each kilogram of PEST contains 30 g of carbaryl and 40 g of Malathion, whereas each kilogram of BUG contains 40 g of carbaryl and 20 g of Malathion. The final blend must contain at least 120 g of carbaryl and at most 80 g of Malathion. If each kilogram of PEST costs \$3.00 and each kilogram of BUG costs \$2.50, how many kilograms of each pesticide should be used to minimize the cost?
2. **Equipment purchasing problem.** A container manufacturer is considering the purchase of two different types of cardboard-folding machines: model A and model B. Model A can fold 30 boxes per minute and requires 1 attendant, whereas model B can fold 50 boxes per minute and requires 2 attendants. Suppose the manufacturer must fold at least 320 boxes per minute and cannot afford more than 12 employees for the folding operation. If a model A machine costs \$15,000 and a model B machine costs \$20,000, how many machines of each type should be bought to minimize the cost?
3. **Disease treatment problem.** Dr. R. C. McGonigal treats cases of tactutis with a combination of the brand-name compounds Palium and Timade. The Palium costs \$0.40/pill and the Timade costs \$0.30/pill. Each compound contains SND plus an activator. The typical dosage requires at least 10 mg of SND per day. Palium contains 4 mg of SND and Timade contains 2 mg of SND. In excessive amounts the activators can be harmful. Consequently Dr. McGonigal limits the total amount of activator to no more than 2 mg per day. Palium and Timade each contain 0.5 mg of activator per pill. How many of each pill per day should Dr. McGonigal prescribe to minimize the cost of the medication, provide enough SND, and yet not exceed the maximum permissible limit of activator?
4. **Agricultural problem.** A farmer owns a farm that produces corn, soybeans, and oats. There are 12 acres of land available for cultivation. Each crop that is planted has certain requirements for labor and capital. These data along with the net profit figures are given in the accompanying table.

	Labor (hr)	*Capital (\$)*	*Net profit (\$)*
Corn (per acre)	6	36	40
Soybeans (per acre)	6	24	30
Oats (per acre)	2	18	20

The farmer has \$360 available for capital and knows that there are 48 hr available for working these crops. How much of each crop should be planted to maximize profit?

5. **Blending problem.** A coffee packer blends Brazilian coffee and Colombian coffee to prepare two products: Super and Deluxe brands. Each kilogram of Super coffee contains 0.5 kg of Brazilian coffee and 0.5 kg of Colombian coffee, whereas each kilogram of Deluxe coffee contains 0.25 kg of Brazilian coffee and 0.75 kg of Colombian coffee. The packer has 120 kg of Brazilian coffee and 160 kg of Colombian coffee on hand. If the profit on each kilogram of Super coffee is 20 cents and the profit on each kilogram of Deluxe coffee is 30 cents, how many kilograms of each type of coffee should be blended to maximize profit?
6. **Air pollution problem.** Consider an airshed in which there is one major contributor to air pollution—a cement-manufacturing plant whose annual production capacity is 2,500,000 barrels of cement. Figures are not available to determine whether the plant has been operating at capacity. Although the kilns are equipped with mechanical collectors for air pollution control, the plant still emits 2.0 lb of dust per barrel of cement produced. There are two types of electrostatic precipitators that can be installed to control dust emission. The four-field type would reduce emissions by 1.5 lb of dust/barrel and would cost \$0.14/barrel to operate. The five-field type would reduce emissions by 1.8 lb of dust/barrel and would cost \$0.18/barrel to operate. The EPA requires that particulate emissions be reduced by at least 84%. How many barrels of cement should be produced using each new control process to minimize the cost of controls and still meet the EPA requirements[1]?
7. **Mixing problem.** The R. H. Lawn Products Co. has available 80 metric tons of nitrate and 50 metric tons of phosphate to use in producing its three types of fertilizer during the coming week. The mixture ratios and profit figures are given in the accompanying table. Determine how the current inventory should be used to maximize the profit.

	Metric tons / 1000 bags		*Profit*
	Nitrate	*Phosphate*	*(\$/1000 bags)*
Regular lawn	4	2	300
Super lawn	4	3	500
Garden	2	2	400

8. **Investment problem.** The administrator of a \$200,000 trust fund set up by Mr. Smith's will must adhere to certain guidelines. The total amount of \$200,000 need not be fully invested at any one time. The money may be invested in three different types of securities: a utilities stock paying a 9% dividend, an electronics stock paying a 4% dividend, and a bond paying 5% interest. Suppose that the amount invested in the stocks cannot be more than half the total amount invested; the amount invested in the utilities stock cannot exceed \$40,000; and the amount invested in the bond must be at least \$70,000. What investment policy should be pursued to maximize the return?

[1]Kohn, Robert E. "A Mathematical Programming Model for Air Pollution Control." *School Sci. Math.* (June 1969).

9. A book publisher is planning to bind the latest potential bestseller in three different bindings: paperback, book club, and library. Each book goes through a sewing and gluing process. The time required for each process is given in the accompanying table.

	Paperback	*Book club*	*Library*
Sewing (min)	2	2	3
Gluing (min)	4	6	10

Suppose the sewing process is available 7 hr per day and the gluing process 10 hr per day. Assume that the profits are \$0.50 on a paperback edition, \$0.80 on a book club edition, and \$1.20 on a library edition. How many books will be manufactured in each binding when the profit is maximized?

10. Major oil companies use linear programming to model many phases of their operations. Consider the following simplified version of part of a refinery operation. Two kinds of aviation gasoline, high octane and low octane, are made by blending four components from the refinery output. For the low-octane gasoline the components are augmented with a small amount of tetraethyllead (TEL) to give the low-octane ratings shown in the accompanying table. The high-octane gasoline is made from the same components when these have been augmented with a larger amount of TEL, giving the high-octane ratings in the table. Assume that the octane rating (OR) of the mixture is the volumetric average of the octane ratings of the components. That is, letting V denote volume, we have

$$\mathrm{OR}_{\mathrm{mix}} = \frac{\mathrm{OR}_{\mathrm{comp1}} \times V_{\mathrm{comp1}} + \mathrm{OR}_{\mathrm{comp2}} \times V_{\mathrm{comp2}} + \cdots}{V_{\mathrm{comp1}} + V_{\mathrm{comp2}} + \cdots}.$$

The vapor pressure (a measure of the tendency of the gasoline to evaporate) of both gasolines must be 7. Assume that the vapor pressure of a mixture is the volumetric average of the vapor pressures of the components. Vapor pressure and octane rating are the only two physical properties for which there are constraints. Data for the components and desired mixtures are given in the accompanying table.

	Vapor pressure	*OR*		*Demand*	*Supply*	*Revenue*	*Cost*
		High	*Low*				
Component							
Alkylate	5	108	98		700		7.20
Catalytic cracked	6.5	94	87		600		4.35
Straight run	4	87	80		900		3.80
Isopentane	18	108	100		500		4.30
Mixture							
High octane	7	100	—	1300		6.50	
Low octane	7	—	90	800		7.50	

Assume that the demands must be met exactly. Measure the profit by using revenue less cost. Set up a model of this situation that maximizes this measure of profit.

11. A local health food store packages three types of snack foods—Chewy, Crunchy, and Nutty—by mixing sunflower seeds, raisins, and peanuts. The specifications for each mixture are given in the accompanying table.

Mixture	*Sunflower seeds*	*Raisins*	*Peanuts*	*Selling price per kilogram ($)*
Chewy		At least 60%	At most 20%	2.00
Crunchy	At least 60%			1.60
Nutty	At most 20%		At least 60%	1.20

The suppliers of the ingredients can deliver each week at most 100 kg of sunflower seeds at $1.00/kg, 80 kg of raisins at $1.50/kg, and 60 kg of peanuts at $0.80/kg. Determine a mixing scheme that will maximize the store's profit.

1.1 PROJECTS

1. **Feed problem.** A laboratory needs to supply its research dogs a mixture of commercially available dog foods that meet the National Research Council (NRC) nutrient requirements (Table 1.1). The nutritional composition of each of the eight available foods is given in Table 1.2. Note that these data are given in terms of percentages.
   (a) Set up a constraint for each of the food constituents listed in Table 1.1 based on the NRC requirements.
   (b) An additional constraint must be provided, because the requirements are given in terms of percentages. It must say that the sum of the amounts used is 1. That is, each variable represents a fraction of the total amount to be blended. Write this constraint.

TABLE 1.1 National Research Council Nutrient Requirements

Food must have at least	*(%)*	*Food must have at most*	*(%)*
Protein	20	Fiber	8
Fat	5	Moisture	5
Linoleic acid	1.4		
Calcium	1		
Phosphorus	0.8		
Potassium	0.5		
Salt	1		
Magnesium	0.4		
NFE	25		

TABLE 1.2 Dog Food Constituents (Percentage by Weight)

	Wayne	*Wayne TW*	*Purina Meal*	*Purina Chow*	*Purina HP*	*Gaines*	*Burgerbits*	*Agway 2000*
Protein	25.0	24.0	27.0	23.8	26.0	21.0	23.0	25.5
Fat	8.0	9.0	10.5	9.4	10.0	8.0	7.0	10.5
Linoleic acid	2.1	1.6	1.6	1.6	1.6	0.9	1.5	1.5
Calcium	2.15	1.20	2.50	1.75	1.60	1.0	1.50	1.50
Phosphorus	1.43	1.00	1.40	1.03	1.20	0.80	0.80	1.70
Potassium	0.73	0.98	0.80	0.71	0.90	0.50	0.50	0.69
Salt	1.15	1.15	0.78	0.64	1.10	1.00	1.50	1.00
Magnesium	0.170	0.220	0.290	0.270	0.150	0.036	0.050	0.230
Fiber	3.5	4.7	4.3	3.7	4.0	5.0	5.0	2.9
NFE	45.77	46.15	41.83	48.10	41.45	51.76	47.15	45.28
Moisture	10.0	10.0	9.0	9.0	12.0	10.0	12.0	9.2
Cost ($/kg)	0.17	0.17	0.17	0.16	0.21	0.20	0.17	0.16

(c) Set up the objective function using the cost data given in Table 1.2.

(d) Consider an arbitrary constraint,

$$a_1x_1 + a_2x_2 + \cdots + a_8x_8 \geq b. \tag{1}$$

Using the constraint in b, show that the constraint in (1) is automatically satisfied if, for all i, $a_i \geq b$. Similarly, show that (1) is impossible to satisfy if, for all i, $a_i < b$. Formulate and prove similar results if the inequality in (1) is reversed.

(e) Using the results in d, identify the redundant constraints and the impossible constraints in a. Rewrite the model, eliminating impossible and redundant constraints.

(f) Discuss why $x_1 = x_2 = \cdots = x_7 = 0$, $x_8 = 1$, is an optimal solution.

2. **Advertising**. The advertising programs of major companies are designed to achieve certain goals in the hope of stimulating sales. There are many media that accept advertising, and the company must decide how to allocate its advertising budget among the different media to achieve the greatest possible benefit. To aid in making this type of decision, there are a number of research firms that collect data concerning the audience of each medium. Suppose a car manufacturer, who requires a four-color one-page unit in a weekly magazine, is presented by the research firm with the accompanying table representing readership characteristics and advertising limitations of three different weekly magazines.

	TV Guide	*Newsweek*	*Time*
Cost per four-color one-page unit ($)	55,000	35,335	49,480
Total male readers per unit	19,089,000	11,075,000	10,813,000
Men 50 years or older per unit	4,312,000	2,808,000	2,714,000
Men who are college graduates per unit	2,729,000	3,387,000	3,767,000

The advertising manager has a monthly budget limitation of $200,000 and must decide what amount to spend for each magazine. Because she is worried about the possible duplication of *TV Guide* with her television advertising schedule, she decides to limit *TV Guide* to a maximum of two advertising units. She can use as many as four advertising units per month in each of *Newsweek* and *Time*. Each time a person reads a magazine, it is counted as an exposure to the advertising in the magazine. The advertising manager wants to obtain at least 12,000,000 exposures to men who are college graduates, and, because men 50 years and older are not good prospects for her products, she wants to limit the number of exposures to no more than 16,000,000 such men. Set up a linear programming model to determine how many advertising units the advertising manager should buy in each magazine if she wants to keep within her budget and wants to maximize the total number of male readers.

3. **Construction problem.** A private contractor has five machines that are capable of doing excavation work available at certain times during the day for a period of one week. He wants to determine which combination of machines he should use in order to get the job done the cheapest way. The size of the excavation is 1000 cubic yards of material, and the material has to be removed in one week's time. His machine operators will work at most an 8-hr day, 5 days per week. In the accompanying table is the capacity of each machine, the cycle time for each machine (the time it takes the machine to dig to its capacity and move the excavated material to a truck), the availability, and the cost (which includes wages for the operator).

Machine	*Capacity (cubic yard)*	*Rate ($/hr)*	*Availability (hr/day)*	*Time needed to excavate 1 unit of capacity (min)*
Shovel dozer	2	17.50	6.0	4.25
Large backhoe	2.5	40.00	6.0	1.00
Backhoe A	1.5	27.50	6.0	1.00
Backhoe B	1	22.00	8.0	1.00
Crane with clamshell	1.5	47.00	5.5	2.25

Set up a linear programming problem to determine what combination of machines to use to complete the job at minimum cost. From the data given you will have to compute the number of cubic yards each machine can excavate in 1 hr. Remember to include a constraint that says that the job must be finished. (What would the solution be if this last constraint were not included?)

4. **Literature search.** Among the many journals that deal with linear programming problems are *Operations Research*, *Management Science*, *Naval Logistics Research Quarterly*, *Mathematics in Operations Research*, *Operational Research Quarterly*, and the *Journal of the Canadian Operational Research Society*. Write a short report on a paper that appears in one of these journals and describes a real situation. The report should include a description of the situation and a discussion of the assumptions that were made in constructing the model.

1.2 MATRIX NOTATION

It is convenient to write linear programming problems in matrix notation. Consider the standard linear programming problem:

$$\text{Maximize} \quad z = c_1x_1 + c_2x_2 + \cdots + c_nx_n \tag{1}$$

subject to

$$\left.\begin{aligned} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n &\le b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n &\le b_2 \\ \vdots \quad\quad \vdots \quad\quad\quad\quad \vdots \quad\quad \vdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n &\le b_m \end{aligned}\right\} \tag{2}$$

$$x_j \ge 0, \quad j = 1, 2, \ldots, n. \tag{3}$$

Letting

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}, \quad \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix},$$

$$\mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}, \quad \text{and } \mathbf{c} = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \end{bmatrix},$$

we can write our given linear programming problem as:

Find a vector $\mathbf{x} \in R^n$ that will

$$\text{maximize} \quad z = \mathbf{c}^T\mathbf{x} \tag{4}$$

subject to

$$\mathbf{Ax} \le \mathbf{b} \tag{5}$$

$$\mathbf{x} \ge \mathbf{0}. \tag{6}$$

Here, writing $\mathbf{v} \le \mathbf{w}$ for two vectors $\mathbf{v}$ and $\mathbf{w}$ means that each entry of $\mathbf{v}$ is less than or equal to the corresponding entry of $\mathbf{w}$. Specifically, $\mathbf{x} \ge \mathbf{0}$ means that each entry of $\mathbf{x}$ is nonnegative.

EXAMPLE 1. The linear programming problem in Example 1 of Section 1.1 can be written in matrix form as follows:

Find a vector $\mathbf{x}$ in R^2 that will

$$\text{maximize} \quad z = \begin{bmatrix} 120 & 100 \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix}$$

subject to

$$\begin{bmatrix} 2 & 2 \\ 5 & 3 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \leq \begin{bmatrix} 8 \\ 15 \end{bmatrix}$$

$$\begin{bmatrix} x \\ y \end{bmatrix} \geq \mathbf{0}.$$

△

DEFINITION. A vector $\mathbf{x} \in R^n$ satisfying the constraints of a linear programming problem is called a **feasible solution** to the problem. A feasible solution that maximizes or minimizes the objective function of a linear programming problem is called an **optimal solution.** △

EXAMPLE 2. Consider the linear programming problem in Example 1. The vectors

$$\mathbf{x}_1 = \begin{bmatrix} 1 \\ 2 \end{bmatrix}, \quad \mathbf{x}_2 = \begin{bmatrix} 2 \\ 1 \end{bmatrix}, \quad \text{and} \quad \mathbf{x}_3 = \begin{bmatrix} 1 \\ 3 \end{bmatrix}$$

are feasible solutions. For example,

$$\begin{bmatrix} 2 & 2 \\ 5 & 3 \end{bmatrix} \begin{bmatrix} 2 \\ 1 \end{bmatrix} = \begin{bmatrix} 6 \\ 13 \end{bmatrix} \leq \begin{bmatrix} 8 \\ 15 \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} 2 \\ 1 \end{bmatrix} \geq \mathbf{0}.$$

Therefore, $\mathbf{x}_2$ is a feasible solution. The vectors $\mathbf{x}_1$ and $\mathbf{x}_3$ can be checked in the same manner. Also, the same technique can be used to show that

$$\mathbf{x}_4 = \begin{bmatrix} 3 \\ 1 \end{bmatrix} \quad \text{and} \quad \mathbf{x}_5 = \begin{bmatrix} 2 \\ 2 \end{bmatrix}$$

are not feasible solutions. Moreover,

$$\mathbf{x}_6 = \begin{bmatrix} -2 \\ 3 \end{bmatrix}$$

is not a feasible solution because one of its entries is negative. Later we will show that

$$\mathbf{x}_0 = \begin{bmatrix} \frac{3}{2} \\ \frac{5}{2} \end{bmatrix}$$

is an optimal solution to the problem. △

We now describe the method for converting a standard linear programming problem into a problem in canonical form. To do this we must be able to change the inequality constraints into equality constraints. In canonical form the constraints form a system of linear equations, and we can use the methods of linear algebra to solve such systems. In particular, we shall be able to employ the steps used in Gauss–Jordan reduction.

Changing an Inequality to an Equality

Consider the constraint

$$a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{in}x_n \leq b_i. \tag{7}$$

We may convert (7) into an equation by introducing a new variable, u_i, and writing

$$a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{in}x_n + u_i = b_i. \tag{8}$$

The variable u_i is nonnegative and is called a **slack variable** because it "takes up the slack" between the left side of constraint (7) and its right side.

We now convert the linear programming problem in standard form given by (1), (2), and (3) to a problem in canonical form by introducing a slack variable in each of the constraints. Note that each constraint will get a different slack variable. In the ith constraint

$$a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{in}x_n \leq b_i,$$

we introduce the slack variable x_{n+i} and write

$$a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{in}x_n + x_{n+i} = b_i.$$

Because of the direction of the inequality, we know that $x_{n+i} \geq 0$. Therefore, the canonical form of the problem is:

$$\text{Maximize} \quad z = c_1x_1 + c_2x_2 + \cdots + c_nx_n \tag{9}$$

subject to

$$\left.\begin{array}{llllll} a_{11}x_1 + & a_{12}x_2 + \cdots + & a_{1n}x_n + x_{n+1} & & & = b_1 \\ a_{21}x_1 + & a_{22}x_2 + \cdots + & a_{2n}x_n & +x_{n+2} & & = b_2 \\ \vdots & \vdots & \vdots & & & \vdots \\ a_{m1}x_1 + & a_{m2}x_2 + \cdots + & a_{mn}x_n & & +x_{n+m} & = b_m \end{array}\right\} \tag{10}$$

$$x_1 \geq 0, \quad x_2 \geq 0, \ldots, \quad x_n \geq 0, \quad x_{n+1} \geq 0, \ldots, \quad x_{n+m} \geq 0. \tag{11}$$

The new problem has m equations in $m + n$ unknowns in addition to the nonnegativity restrictions on the variables $x_1, x_2, \ldots, x_n, x_{n+1}, \ldots, x_{n+m}$.

If $\mathbf{y} = [y_1 \quad y_2 \quad \cdots \quad y_n]^T$ is a feasible solution to the problem given by (1), (2), and (3), then we define y_{n+i}, $i = 1, 2, \ldots, m$, by

$$y_{n+i} = b_i - a_{i1}y_1 - a_{i2}y_2 - \cdots - a_{in}y_n,$$

That is, y_{n+i} is the difference between the right side of the ith constraint in (2) and the value of the left side of this constraint at the feasible solution $\mathbf{y}$. Since each constraint in (2) is of the $\leq$ form, we conclude that

$$y_{n+i} \geq 0, \quad i = 1, 2, \ldots, m.$$

Thus, $[y_1 \quad y_2 \quad \cdots \quad y_n \quad y_{n+1} \quad \cdots \quad y_{n+m}]^T$ satisfies (10) and (11), and the vector $\hat{\mathbf{y}} = [y_1 \quad y_2 \quad \cdots \quad y_{n+m}]^T$ is a feasible solution to the problem in (9), (10), and (11).

Conversely, suppose $\hat{\mathbf{y}} = [y_1 \quad y_2 \quad \cdots \quad y_{n+m}]^T$ is a feasible solution to the linear programming problem in canonical form given by (9), (10), and (11). Then clearly $y_1 \geq 0$, $y_2 \geq 0, \ldots, y_n \geq 0$. Since $y_{n+i} \geq 0$, $i = 1, 2, \ldots, m$, we see that

$$a_{i1}y_1 + a_{i1}y_2 + \cdots + a_{in}y_n \leq b_i \quad \text{for} \quad i = 1, 2, \ldots, m.$$

Hence, $\mathbf{y} = [y_1 \quad y_2 \quad \cdots \quad y_n]^T$ is a feasible solution to the linear programming problem in standard form given by (1), (2), and (3).

The discussion above has shown that a feasible solution to a standard linear programming problem yields a feasible solution to a canonical linear programming problem by adjoining the values of the slack variables. Conversely, a feasible solution to a canonical linear programming problem yields a feasible solution to the corresponding standard linear programming problem by truncating the slack variables.

EXAMPLE 3. Consider Example 1 again. Introducing the slack variables u and v, our problem becomes:

$$\begin{aligned} &\text{Maximize} \quad z = 120x + 100y \\ &\text{subject to} \\ &\quad 2x + 2y + u \qquad = 8 \\ &\quad 5x + 3y \qquad + v = 15 \\ &x \geq 0, \quad y \geq 0, \quad u \geq 0, \quad v \geq 0. \end{aligned}$$

In terms of the model, the slack variable u is the difference between the total amount of time that the saw is available, 8 hr, and the amount of time that it is actually used, $2x + 2y$ (in hours). Similarly, the slack variable v is the difference between the total amount of time that the plane is available, 15 hr, and the amount of time that it is actually used, $5x + 3y$ (in hours).

We showed in Example 2 of this section that $x = 2$, $y = 1$ is a feasible solution to the problem in standard form. For this feasible solution we have

$$\begin{aligned} u &= 8 - 2 \cdot 2 - 2 \cdot 1 = 2 \\ v &= 15 - 5 \cdot 2 - 3 \cdot 1 = 2. \end{aligned}$$

Thus,

$$x = 2, \quad y = 1, \quad u = 2, \quad v = 2$$

is a feasible solution to the new form of the problem.

Consider now the values

$$x = 1, \quad y = 1, \quad u = 4, \quad v = 7.$$

These values are a feasible solution to the new problem, since

$$2 \cdot 1 + 2 \cdot 1 + 4 = 8$$

and

$$5 \cdot 1 + 3 \cdot 1 + 7 = 15.$$

Consequently,

$$x = 1, \quad y = 1$$

is a feasible solution to the given problem.

We will show in Example 3 of Section 1.3 that an optimal solution to this problem is

$$x = \frac{3}{2}, \quad y = \frac{5}{2}.$$

In canonical form this feasible solution gives the following values for u and v.

$$u = 8 - 2 \cdot \tfrac{3}{2} - 2 \cdot \tfrac{5}{2} = 0$$
$$v = 15 - 5 \cdot \tfrac{3}{2} - 3 \cdot \tfrac{5}{2} = 0.$$

That is, an optimal feasible solution to the canonical form of the problem is

$$x = \tfrac{3}{2}, \quad y = \tfrac{5}{2}, \quad u = 0, \quad v = 0. \qquad \triangle$$

The linear programming problem given by (9), (10), and (11) can also be written in matrix form as follows. We now let

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} & 1 & 0 & \cdots & 0 \\ a_{21} & a_{22} & \cdots & a_{2n} & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & & \vdots & \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} & 0 & 0 & \cdots & 1 \end{bmatrix},$$

$$\mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}, \quad \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \\ x_{n+1} \\ \vdots \\ x_{n+m} \end{bmatrix}, \quad \mathbf{c} = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \\ 0 \\ \vdots \\ 0 \end{bmatrix}.$$

Then this problem can be written as:

$$\text{Maximize} \quad z = \mathbf{c}^{\mathrm{T}}\mathbf{x} \tag{12}$$

subject to

$$\mathbf{A}\mathbf{x} = \mathbf{b} \tag{13}$$

$$\mathbf{x} \geq \mathbf{0}. \tag{14}$$

Note that this problem is in canonical form.

EXAMPLE 4. The linear programming problem that was formulated in Example 3 can be written in matrix form as:

$$\text{Maximize} \quad z = [120 \quad 100 \quad 0 \quad 0]\begin{bmatrix} x \\ y \\ u \\ v \end{bmatrix}$$

subject to

$$\begin{bmatrix} 2 & 2 & 1 & 0 \\ 5 & 3 & 0 & 1 \end{bmatrix}\begin{bmatrix} x \\ y \\ u \\ v \end{bmatrix} = \begin{bmatrix} 8 \\ 15 \end{bmatrix}$$

$$\begin{bmatrix} x \\ y \\ u \\ v \end{bmatrix} \geq \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}.$$

Note that this problem is in canonical form. △

1.2 EXERCISES

In Exercises 1–4, write the indicated linear programming problem from Section 1.1 in matrix form.

1. Example 4
2. Example 8
3. Example 9
4. Exercise 4

In Exercises 5–10, convert the indicated linear programming problem from Section 1.1 to canonical form and express this form in matrix notation.

5. Example 7a
6. Example 7e
7. Example 7c
8. Exercise 1
9. Exercise 5
10. Exercise 7
11. (a) For the linear programming problem in Example 1, show that

$$\mathbf{x}_1 = \begin{bmatrix} 1 \\ 2 \end{bmatrix} \quad \text{and} \quad \mathbf{x}_3 = \begin{bmatrix} 1 \\ 3 \end{bmatrix}$$

are feasible solutions. Also compute the values of the objective function for these feasible solutions.

(b) Show that

$$\mathbf{x}_4 = \begin{bmatrix} 3 \\ 1 \end{bmatrix}, \quad \mathbf{x}_5 = \begin{bmatrix} 2 \\ 2 \end{bmatrix}, \quad \mathbf{x}_6 = \begin{bmatrix} -2 \\ 3 \end{bmatrix}$$

are not feasible solutions.

12. Write the following linear programming problem in canonical form.

$$\text{Maximize} \quad z = [2 \quad 3 \quad 5] \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

subject to

$$\begin{bmatrix} 3 & 2 & 1 \\ 1 & 1 & -2 \\ 2 & 5 & 4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \leq \begin{bmatrix} 5 \\ 8 \\ 10 \end{bmatrix}$$

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \geq \mathbf{0}.$$

13. Consider the linear programming problem

$$\begin{aligned} &\text{Maximize} \quad z = x + 4y \\ &\text{subject to} \\ &\qquad 3x + 4y \leq 21 \\ &\qquad x + 2y \leq 12 \\ &\qquad x \geq 0, \quad y \geq 0. \end{aligned}$$

Let u and v be the slack variables in the first and second inequalities, respectively.

(a) Determine, if possible, a feasible solution to the canonical form problem in which $u = 3$ and $v = 4$.

(b) Determine, if possible, a feasible solution to the canonical form problem in which $u = 18$ and $v = 10$.

14. Consider the linear programming problem

$$\begin{aligned} &\text{Maximize} \quad z = 2x + 5y \\ &\text{subject to} \\ &\qquad 2x + 3y \leq 10 \\ &\qquad 5x + \ y \leq 12 \\ &\qquad x + 5y \leq 15 \\ &\qquad x \geq 0, \quad y \geq 0. \end{aligned}$$

(a) Verify that $\mathbf{x} = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$ is a feasible solution.

(b) For the feasible solution in a, find the corresponding values of the slack variables.

15. Consider the linear programming problem

$$\begin{aligned} &\text{Maximize} \quad z = \mathbf{c}^T\mathbf{x} \\ &\text{subject to} \\ &\qquad \mathbf{Ax} \leq \mathbf{b} \\ &\qquad \mathbf{x} \geq \mathbf{0}. \end{aligned}$$

If $\mathbf{x}_1$ and $\mathbf{x}_2$ are feasible solutions, show that

$$\mathbf{x} = \tfrac{1}{3}\mathbf{x}_1 + \tfrac{2}{3}\mathbf{x}_2$$

is a feasible solution.

16. Generalize the previous exercise to show that

$$\mathbf{x} = r\mathbf{x}_1 + s\mathbf{x}_2$$

is a feasible solution if $r + s = 1$.

1.3 GEOMETRY OF LINEAR PROGRAMMING PROBLEMS

In this section we consider the geometry of linear programming problems by first looking at the geometric interpretation of a single constraint, then at a set of constraints, and finally at the objective function. These ideas give rise to a geometric method for solving a linear programming problem that is successful only for problems with two or three variables. However, the geometric concepts that we discuss can be built into an algebraic algorithm that can effectively solve very large problems. After casting the geometric ideas in a linear algebra setting, we will present this algorithm—the simplex method—in Section 2.1.

Geometry of a Constraint

A single constraint of a linear programming problem in standard form, say the ith one,

$$a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{in}x_n \leq b_i,$$

can be written as

$$\mathbf{a}^T\mathbf{x} \leq b_i,$$

where

$$\mathbf{a}^T = [a_{i1} \quad a_{i2} \quad \cdots \quad a_{in}].$$

The set of points $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ in R^n that satisfy this constraint is called a **closed half-space**. If the inequality is reversed, the set of points $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ in R^n satisfying

$$\mathbf{a}^T\mathbf{x} \geq b_i$$

is also called a closed half-space.

EXAMPLE 1. Consider the constraint $2x + 3y \leq 6$ and the closed half-space

$$H = \left\{ \begin{bmatrix} x \\ y \end{bmatrix} \middle| \begin{bmatrix} 2 & 3 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \leq 6 \right\},$$

which consists of the points satisfying the constraint. Note that the points $(3, 0)$ and $(1, 1)$ satisfy the inequality and therefore are in H. Also, the points $(3, 4)$ and $(-1, 3)$ do not satisfy the inequality and therefore are not in H. Every point on the line $2x + 3y = 6$ satisfies the constraint and thus lies in H. △

We can graph a closed half-space in R^2 by graphing the line and then using a test point to decide which side of the line is included in the half-space. A simple way to graph the line is to find its x- and y-intercepts. By setting $y = 0$ in the equation of the line and solving for x, we obtain the x-intercept and plot it on the x-axis. Similarly, by setting $x = 0$ in the equation of the line and solving for y, we obtain the y-intercept and plot it on the y-axis. We now connect the two points to sketch the graph of the line. To choose a test point, we check whether the origin is on the line. If it is not, we use it as a test point, checking whether the origin satisfies the inequality. If it does, the side of the line containing the origin (our test point) is the closed half-space H. If it does not, then the other side of the line is the closed half-space. If the origin is on the line, some other point not on the line must be selected as the test point. Some possible choices are $(1, 0)$, $(0, 1)$, or $(1, 1)$.

EXAMPLE 1 (continued). We compute the x-intercept to be $x = 3$ and the y-intercept to be $y = 2$. These points have been plotted and the line connecting them has been drawn in Figure 1.2a. Since the origin does not lie on the line $2x + 3y = 6$, we use the origin as the test point. The coordinates of the origin satisfy the inequality, so that H lies below the line and contains the origin as shown in Figure 1.2b. △

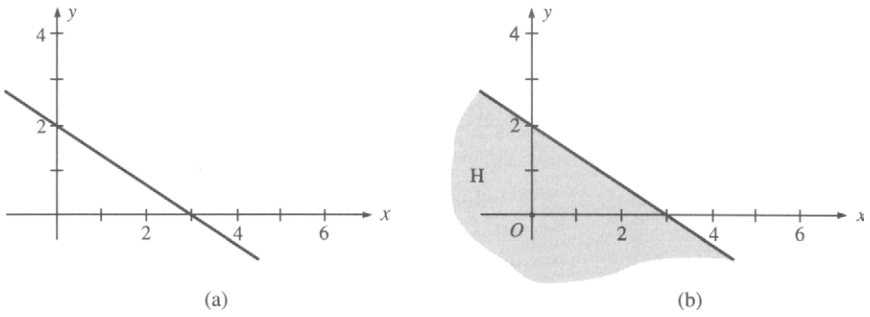


FIGURE 1.2 Closed half-space in two dimensions.

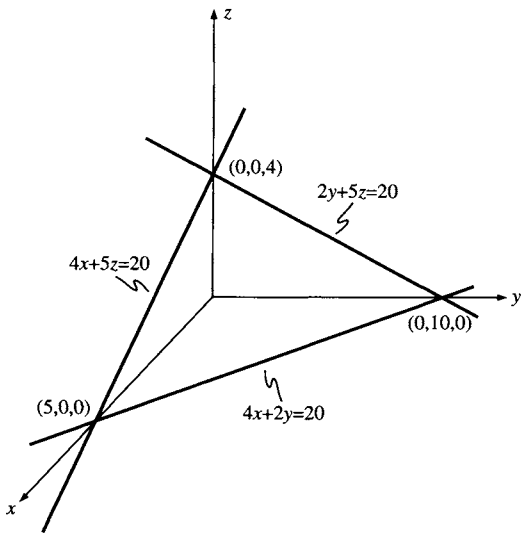


FIGURE 1.3 Closed half-space in three dimensions.

EXAMPLE 2. The constraint in three variables, $4x + 2y + 5z \leq 20$, defines the closed half-space H in R^3, where

$$H = \left\{ \begin{bmatrix} x \\ y \\ z \end{bmatrix} \middle| \begin{bmatrix} 4 & 2 & 5 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \leq 20 \right\}.$$

We can graph H in R^3 by graphing the plane $4x + 2y + 5z = 20$ and checking a test point. To graph the plane, we graph the intersection of the plane with each of the coordinate planes. These intersections are lines in the coordinate planes. Thus, letting $z = 0$ yields the line $4x + 2y = 20$ in the xy plane, which can be graphed as described in Example 1. Similarly, letting $y = 0$ yields the line $4x + 5z = 20$ in the xz plane. Finally, setting $x = 0$ yields the line $2y + 5z = 20$ in the yz plane. The graph of the plane containing these lines is shown in Figure 1.3. The origin does not lie on the plane and thus can be used as a test point. It satisfies the inequality so that the closed half-space contains the origin as shown in Figure 1.3. △

In more than three dimensions, it is impossible to sketch a closed half-space. However, we can think about the geometry of closed half-spaces in any dimension and use the lower dimension examples as models for our computations.

A typical constraint of a linear programming problem in canonical form has the equation

$$\mathbf{a}^T\mathbf{x} = b. \tag{1}$$

Its graph in R^n is a **hyperplane**. If this equation were an inequality, namely,

$$\mathbf{a}^T\mathbf{x} \leq b,$$

then the set of points satisfying the inequality would be a closed half-space. Thus, a hyperplane is the **boundary** of a closed half-space. Intuitively, it consists of the points that are in the half-space, but on its edge.

EXAMPLE 3. The equation $4x + 2y + 5z = 20$ defines a hyperplane $\hat{H}$ in R^3. The graph of this hyperplane, which is really a plane in this case, is shown in Figure 1.3. The hyperplane $\hat{H}$ is the boundary of the closed half-space H_1 defined by the inequality $4x + 2y + 5z \leq 20$, considered in Example 2. The half-space H_1 extends below the hyperplane $\hat{H}$ and lies behind the page. We also see that $\hat{H}$ is the boundary of the closed half-space H_2 defined by the inequality $4x + 2y + 5z \geq 20$. The half-space H_2 extends above the hyperplane and reaches out of the page. △

The hyperplane H defined by (1) divides R^n into the two closed half-spaces

$$H_1 = \{x \in R^n \mid \mathbf{a}^T \leq b\}$$

and

$$H_2 = \{x \in R^n \mid \mathbf{a}^T \geq b\}.$$

We also see that $H_1 \cap H_2 = \hat{H}$, the original hyperplane. In other words, a hyperplane is the intersection of two closed half-spaces.

Recall from Section 1.2 that a feasible solution to a linear programming problem is a point in R^n that satisfies all the constraints of the problem. It then follows that this set of feasible solutions is the intersection of all the closed half-spaces determined by the constraints. Specifically, the set of solutions to an inequality ($\leq$ or $\geq$) constraint is a single closed half-space, whereas the set of solutions to an equality constraint is the intersection of two closed half-spaces.

EXAMPLE 4. Sketch the set of all feasible solutions satisfying the set of inequalities

$$\begin{aligned} 2x + 3y &\leq 6 \\ -x + 2y &\leq 4 \\ x &\geq 0 \\ y &\geq 0. \end{aligned}$$

Solution. The set of solutions to the first inequality, $2x + 3y \leq 6$, is shown as the shaded region in Figure 1.4a and the set of solutions to the

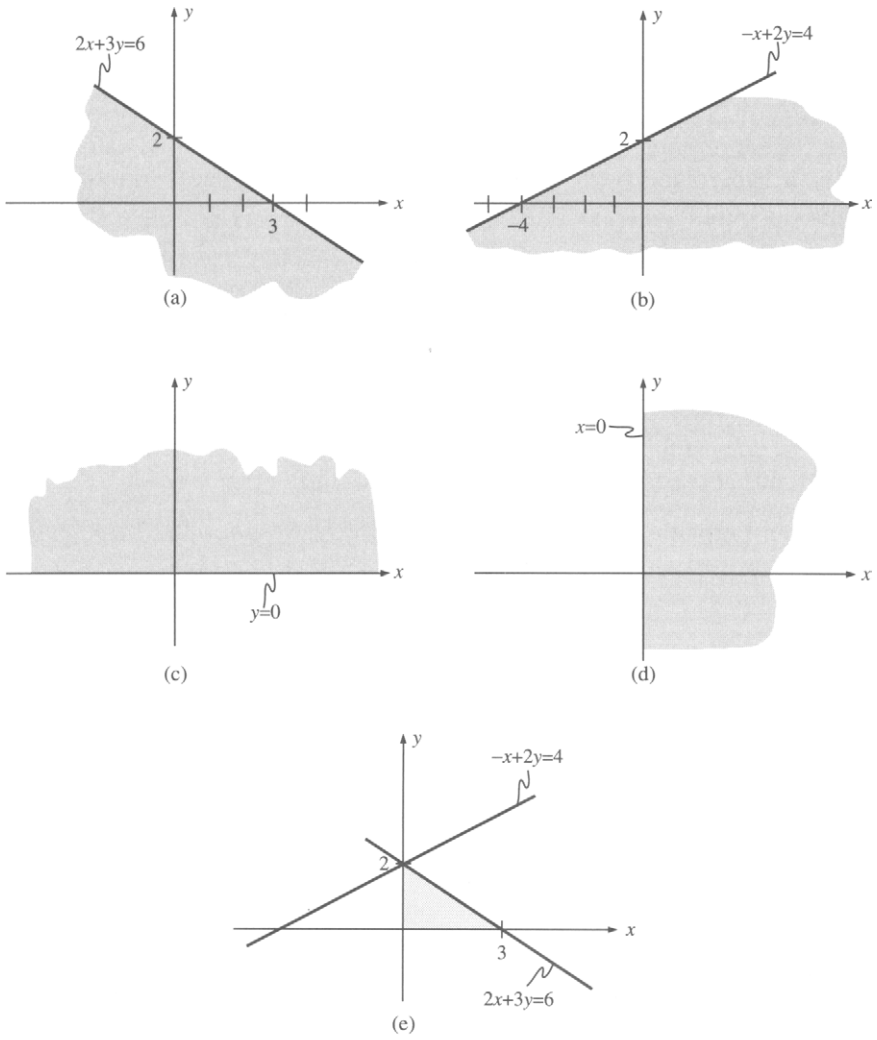


FIGURE 1.4 Set of all feasible solutions (two dimensions).

second inequality, $-x + 2y \leq 4$, form the shaded region in Figure 1.4b. In determining these regions, we have used the origin as a test point. The regions satisfying the third and fourth constraints are shown in Figures 1.4c and 1.4d, respectively. The point $(1, 1)$ was used as a test point to determine these regions. The intersection of the regions in Figures 1.4a–1.4d is shown in Figure 1.4e; it is the set of all feasible solutions to the given set of constraints. △

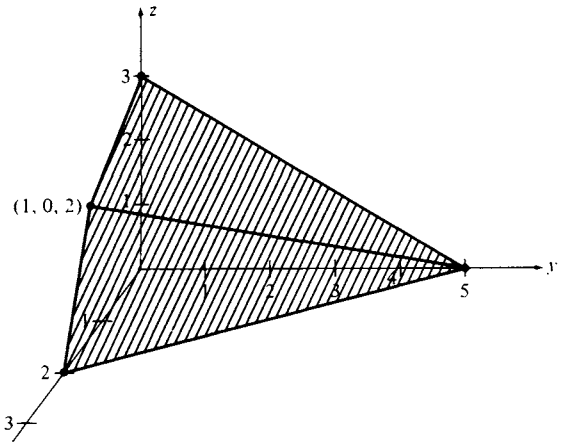


FIGURE 1.5 Set of all feasible solutions (three dimensions).

EXAMPLE 5. Using the same technique as in the previous example, we find the sketch of the region in R^3 defined by the inequalities

$$\begin{aligned} x &\geq 0 \\ y &\geq 0 \\ z &\geq 0 \\ 5x + 3y + 5z &\leq 15 \\ 10x + 4y + 5z &\leq 20. \end{aligned}$$

The first three inequalities limit us to the first octant of R^3. The other two inequalities define certain closed half-spaces. The region, which is the intersection of these two half-spaces in the first octant, is shown in Figure 1.5.

Geometry of the Objective Function

The objective function of any linear programming problem can be written as

$$\mathbf{c}^T\mathbf{x}.$$

If k is a constant, then the graph of the equation

$$\mathbf{c}^T\mathbf{x} = k$$

is a hyperplane. Assume that we have a linear programming problem that asks for a maximum value of the objective function. In solving this problem, we are searching for points $\mathbf{x}$ in the set of feasible solutions for which the value of k is as large as possible. Geometrically we are looking

for a hyperplane that intersects the set of feasible solutions and for which k is a maximum. The value of k measures the distance from the origin to the hyperplane. We can think of starting with very large values of k and then decreasing them until we find a hyperplane that just touches the set of feasible solutions.

EXAMPLE 6. Consider the linear programming problem

$$\begin{aligned} &\text{Maximize} \quad z = 4x + 3y \\ &\text{subject to} \\ &\qquad x + \ y \le 4 \\ &\qquad 5x + 3y \le 15 \\ &\qquad x \ge 0, \quad y \ge 0 \end{aligned}$$

The set of feasible solutions (the shaded region) and the hyperplanes

$$z = 9, \quad z = 12, \quad z = \tfrac{27}{2}, \quad \text{and} \quad z = 15$$

are shown in Figure 1.6. Note that it appears that the maximum value of the objective function is $\frac{27}{2}$, which is obtained when $x = \frac{3}{2}, y = \frac{5}{2}$. This conjecture will be verified in a later section. △

A linear programming problem may not have a solution if the set of feasible solutions is unbounded. In the following example, we are asked to maximize the value of the objective function, but we discover that no such maximum exists.

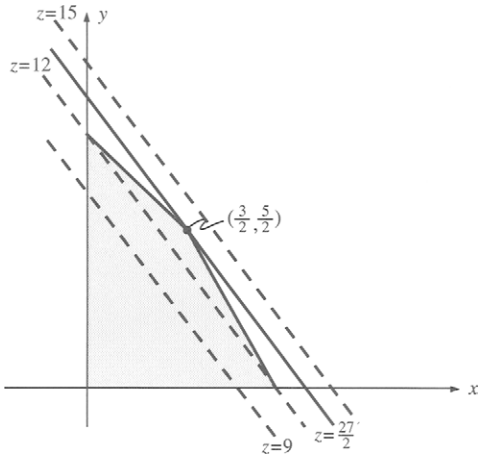


FIGURE 1.6 Objective function hyperplanes (two dimensions).

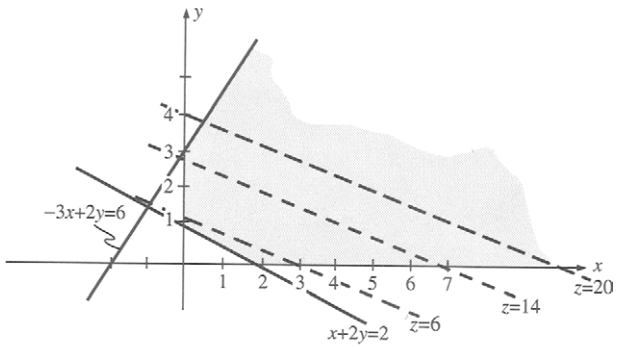


FIGURE 1.7 Unbounded objective function (two dimensions).

EXAMPLE 7. Consider the linear programming problem

$$\begin{aligned} &\text{Maximize} \quad z = 2x + 5y \\ &\text{subject to} \\ &\qquad -3x + 2y \leq 6 \\ &\qquad\quad x + 2y \geq 2 \\ &\qquad x \geq 0, \quad y \geq 0 \end{aligned}$$

The graph of the set of feasible solutions is shown as the shaded region in Figure 1.7. We have also drawn the graphs of the hyperplanes

$$z = 6, \quad z = 14, \quad \text{and} \quad z = 20.$$

We see that in each case there are points that lie to the right of the hyperplane and that are still in the set of feasible solutions. Evidently the value of the objective function can be made arbitrarily large. △

EXAMPLE 8. Consider a linear programming problem that has the same set of constraints as in Example 7. However, assume that it is a minimization problem with objective function

$$z = 3x + 5y.$$

We have drawn the graph of the set of feasible solutions in Figure 1.8 (the shaded region) and the hyperplanes

$$z = 6, \quad z = 9, \quad \text{and} \quad z = 15.$$

It appears that the optimum value of the objective function is $z = 5$ which is obtained when $x = 0$, $y = 1$. Smaller values of the objective function, such as $z = 3$, yield graphs of hyperplanes that do not intersect the set of

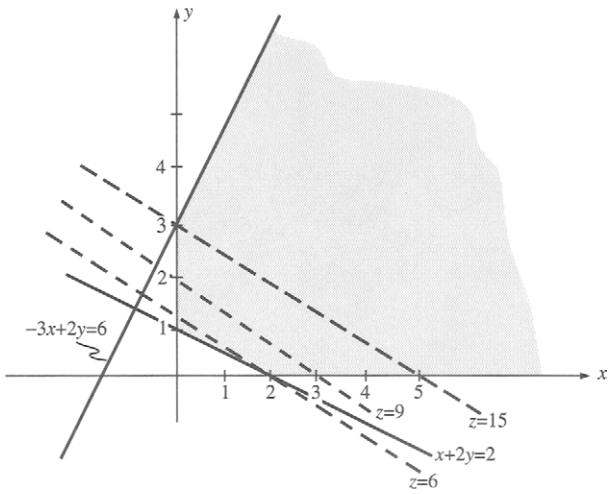


FIGURE 1.8 Minimization (two dimensions).

feasible solutions. You may want to sketch the hyperplanes corresponding to $z = 3$ and $z = 5$. △

Geometry of the Set of Feasible Solutions

We now explore the question of where in the set of feasible solutions we are likely to find a point at which the objective function takes on its optimal value. We first show that if $\mathbf{x}_1$ and $\mathbf{x}_2$ are two feasible solutions, then any point on the line segment joining these two points is also a feasible solution. The **line segment** joining $\mathbf{x}_1$ and $\mathbf{x}_2$ is defined as

$$\{\mathbf{x} \in R^n \,|\, \mathbf{x} = \lambda \mathbf{x}_1 + (1 - \lambda)\mathbf{x}_2, \quad 0 \leq \lambda \leq 1\}.$$

Observe that, if $\lambda = 0$, we get $\mathbf{x}_2$ and, if $\lambda = 1$, we get $\mathbf{x}_1$. The points of the line segment at which $0 < \lambda < 1$ are called the **interior points** of the line segment, and $\mathbf{x}_1$ and $\mathbf{x}_2$ and called its **end points**.

Now suppose that $\mathbf{x}_1$ and $\mathbf{x}_2$ are feasible solutions of a linear programming problem. If

$$\mathbf{a}^T\mathbf{x} \leq b_i$$

is a constraint of the problem, then we have

$$\mathbf{a}^T\mathbf{x}_1 \leq b_i \quad \text{and} \quad \mathbf{a}^T\mathbf{x}_2 \leq b_i.$$

For any point $\mathbf{x} = \lambda\mathbf{x}_1 + (1 - \lambda)\mathbf{x}_2$, $0 \leq \lambda \leq 1$, on the line segment joining $\mathbf{x}_1$ and $\mathbf{x}_2$, we have

$$\begin{aligned}\mathbf{a}^T\mathbf{x} &= \mathbf{a}^T(\lambda\mathbf{x}_1 + (1 - \lambda)\mathbf{x}_2) \\ &= \lambda\mathbf{a}^T\mathbf{x}_1 + (1 - \lambda)\mathbf{a}^T\mathbf{x}_2 \\ &\leq \lambda b_i + (1 - \lambda)b_i \\ &= b_i.\end{aligned}$$

Hence, $\mathbf{x}$ also satisfies the constraint. This result also holds if the inequality in the constraint is reversed or if the constraint is an equality. Thus, the line segment joining any two feasible solutions to a linear programming problem is contained in the set of feasible solutions.

Consider now two feasible solutions $\mathbf{x}_1$ and $\mathbf{x}_2$ to a linear programming problem in standard form with objective function $\mathbf{c}^T\mathbf{x}$. If the objective function has the same value k at $\mathbf{x}_1$ and $\mathbf{x}_2$, then proceeding as above we can easily show that it has the value k at any point on the line segment joining $\mathbf{x}_1$ and $\mathbf{x}_2$ (Exercise 32). Suppose that the value of the objective function is different at $\mathbf{x}_1$ and $\mathbf{x}_2$ and say

$$\mathbf{c}^T\mathbf{x}_1 < \mathbf{c}^T\mathbf{x}_2.$$

If $\mathbf{x} = \lambda\mathbf{x}_1 + (1 - \lambda)\mathbf{x}_2$, $0 < \lambda < 1$, is any interior point of the line segment joining $\mathbf{x}_1$ and $\mathbf{x}_2$, then

$$\begin{aligned}\mathbf{c}^T\mathbf{x} &= \mathbf{c}^T(\lambda\mathbf{x}_1 + (1 - \lambda)\mathbf{x}_2) \\ &= \lambda\mathbf{c}^T\mathbf{x}_1 + (1 - \lambda)\mathbf{c}^T\mathbf{x}_2 \\ &< \lambda\mathbf{c}^T\mathbf{x}_2 + (1 - \lambda)\mathbf{c}^T\mathbf{x}_2 \\ &= \mathbf{c}^T\mathbf{x}_2.\end{aligned}$$

That is, the value of the objective function at any interior point of the line segment is less than its value at one end point. In the same manner we may show that the value of the objective function at any interior point of the line segment is greater than its value at the other end point (verify). Summarizing, we conclude that, on a given line segment joining two feasible solutions to a linear programming problem, the objective function either is a constant or attains a maximum at one end point and a minimum at the other. Thus, the property that a set contains the line segment joining any two points in it has strong implications for linear programming. The following definition gives a name to this property.

DEFINITION. A subset S of R^n is called **convex** if for any two distinct points $\mathbf{x}_1$ and $\mathbf{x}_2$ in S the line segment joining $\mathbf{x}_1$ and $\mathbf{x}_2$ lies in S. That is,

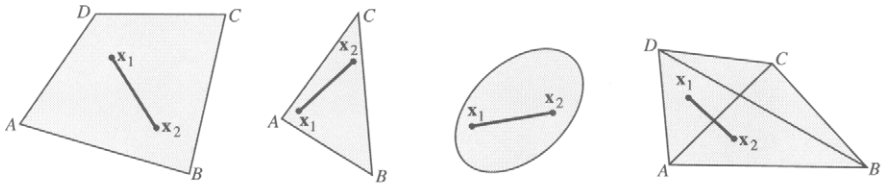


FIGURE 1.9

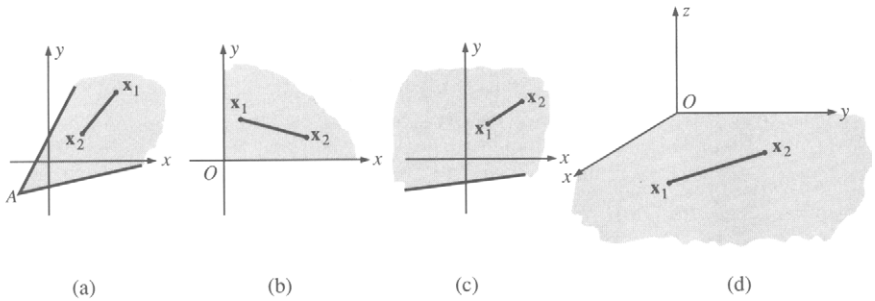


FIGURE 1.10

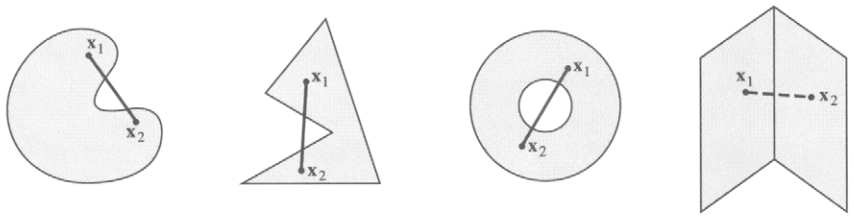


FIGURE 1.11

S is convex if, whenever $\mathbf{x}_1$ and $\mathbf{x}_2 \in S$, so does

$$\mathbf{x} = \lambda\mathbf{x}_1 + (1 - \lambda)\mathbf{x}_2 \quad \text{for} \quad 0 \leq \lambda \leq 1. \qquad \Delta$$

EXAMPLE 9. The sets in R^2 in Figures 1.9 and 1.10 are convex. The sets in R^2 in Figure 1.11 are not convex. Δ

The following results help to identify convex sets.

THEOREM 1.1. *A closed half-space is a convex set.*

Proof. Let the half-space H_1 be defined by $\mathbf{c}^T\mathbf{x} \leq k$. Let $\mathbf{x}_1$ and $\mathbf{x}_2 \in H_1$ and consider $\mathbf{x} = \lambda\mathbf{x}_1 + (1 - \lambda)\mathbf{x}_2$, $(0 \leq \lambda \leq 1)$.

Then

$$\begin{aligned}\mathbf{c}^T\mathbf{x} &= \mathbf{c}^T[\lambda\mathbf{x}_1 + (1 - \lambda)\mathbf{x}_2] \\ &= \lambda\mathbf{c}^T\mathbf{x}_1 + (1 - \lambda)\mathbf{c}^T\mathbf{x}_2.\end{aligned}$$

Since $\lambda \geq 0$ and $1 - \lambda \geq 0$, we obtain

$$\mathbf{c}^T\mathbf{x} \leq \lambda k + (1 - \lambda)k = k.$$

Thus,

$$\mathbf{c}^T\mathbf{x} \leq k,$$

so that $\mathbf{x} \in H_1$. △

THEOREM 1.2. *A hyperplane is a convex set.*

Proof. Exercise. △

THEOREM 1.3. *The intersection of a finite collection of convex sets is convex.*

Proof. Exercise. △

THEOREM 1.4. *Let* $\mathbf{A}$ *be an* $m \times n$ *matrix, and let* $\mathbf{b}$ *be a vector in* R^m. *The set of solutions to the system of linear equations* $\mathbf{Ax} = \mathbf{b}$, *if it is not empty, is a convex set.*

Proof. Exercise. △

Convex sets are of two types: bounded and unbounded. To define a bounded convex set, we first need the concept of a rectangle. A **rectangle** in R^n is a set,

$$R = \{\mathbf{x} \in R^n \mid a_i \leq x_i \leq b_i\},$$

where $a_i < b_i$, $i = 1, 2, \ldots, n$, are real numbers. A **bounded** convex set is one that can be enclosed in a rectangle in R^n. An **unbounded** convex set cannot be so enclosed. The convex sets illustrated in Figure 1.9 are bounded; those in Figure 1.10 are unbounded.

1.3 EXERCISES

In Exercises 1–6 sketch the convex set formed by the intersections of the half-space determined by the given inequalities. Also indicate whether the convex set is bounded.

1.
$$\begin{aligned} x + y &\leq 5 \\ 2x + y &\leq 8 \\ x \geq 0, \quad y &\geq 0 \end{aligned}$$

2.
$$\begin{aligned} x - y &\leq -2 \\ 2x - y &\leq 0 \\ 3x + y &\leq 6 \\ x \geq 0, \quad y &\geq 0 \end{aligned}$$

3.
$$\begin{aligned} 4x + \ y &\geq 8 \\ 3x + 2y &\geq 6 \\ x \geq 0, \quad y &\geq 0 \end{aligned}$$

4.
$$\begin{aligned} 3x + \ y &\leq 6 \\ 2x + 3y &\geq 4 \\ x \geq 0, \quad y &\geq 0 \end{aligned}$$

5.
$$\begin{aligned} 2x + 5y + 5z &\le 20 \\ 4x + 2y + z &\le 8 \\ x \ge 0, \quad y \ge 0, \quad z &\ge 0 \end{aligned}$$

6.
$$\begin{aligned} 4x + 5y + 4z &\le 20 \\ 20x + 12y + 15z &\le 60 \\ x \ge 0, \quad y \ge 0, \quad z &\ge 0 \end{aligned}$$

In Exercises 7–12 sketch the set of feasible solutions to the given set of inequalities.

7.
$$\begin{aligned} -x + y &\le 2 \\ 2x + y &\le 4 \\ x \ge 0, \quad y &\ge 0 \end{aligned}$$

8.
$$\begin{aligned} x + y &\le 3 \\ 2x + y &\le 4 \\ x \ge 0, \quad y &\ge 0 \end{aligned}$$

9.
$$\begin{aligned} x + y &\ge 3 \\ -3x + 2y &\le 6 \\ x \ge 0, \quad y &\ge 0 \end{aligned}$$

10.
$$\begin{aligned} -x + y &\le 2 \\ 2x + y &\le 2 \\ y &\le 1 \\ x &\ge 0 \end{aligned}$$

11.
$$\begin{aligned} 6x + 4y + 9z &\le 36 \\ 2x + 5y + 4z &\le 20 \\ x \ge 0, \quad y \ge 0, \quad z &\ge 0 \end{aligned}$$

12.
$$\begin{aligned} 12x + 6y + 16z &\le 84 \\ 8x + 5y + 12z &\le 60 \\ x \ge 0, \quad y \ge 0, \quad z &\ge 0 \end{aligned}$$

In Exercises 13–16 (a) sketch the set of feasible solutions to the given linear programming problem, (b) draw the objective function $z = \mathbf{c}^T\mathbf{x} = k$, for the indicated values of k, and (c) conjecture the optimal value of z.

13. Maximize $z = 3x + 4y$
subject to
$$\begin{aligned} x + 3y &\le 6 \\ 4x + 3y &\le 12 \\ x \ge 0, \quad y &\ge 0 \end{aligned}$$
$k = 6$, 8, 10, and 12.

14. Maximize $z = 2x + 3y$
subject to
$$\begin{aligned} x + y &\le 4 \\ 3x + y &\le 6 \\ x + 3y &\le 6 \\ x \ge 0, \quad y &\ge 0 \end{aligned}$$
$k = 4$, 6, 8, and 10.

15. Maximize $z = 3x + y$
subject to
$$\begin{aligned} -2x + 3y &\le 6 \\ x + y &\le 4 \\ 3x + y &\le 6 \\ x \ge 0, \quad y &\ge 0 \end{aligned}$$
$k = 2$, 6, 8, and 12.

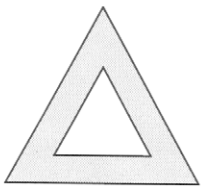
16. Maximize $z = 4x_1 + 8x_2 + x_3$
subject to

$$\begin{aligned} 8x_1 + 2x_2 + 5x_3 &\le 68 \\ 5x_1 + 9x_2 + 7x_3 &\le 120 \\ 13x_1 + 11x_2 + 43x_3 &\le 250 \\ x_1 \ge 0, \quad x_2 &\ge 0, \quad x_3 \ge 0 \end{aligned}$$

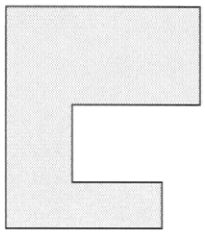
$k = 80$, 90, 100, and 110.

In Exercises 17–24 determine whether the given set is convex.

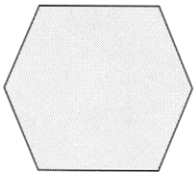
17.



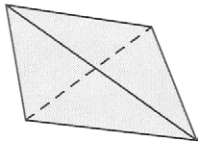
18.



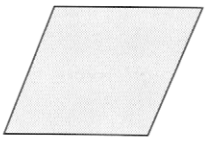
19.



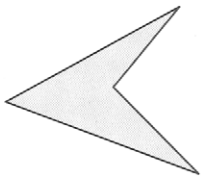
20.



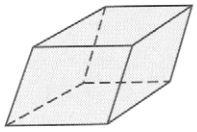
21.



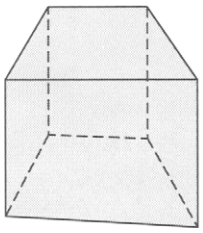
22.



23.



24.



25. Prove that R^n is a convex set.
26. Prove that a subspace of R^n is a convex set.
27. Show that a rectangle in R^n is a convex set.
28. Let H_2 be the half-space in R^n defined by $\mathbf{c}^T\mathbf{x} \geq k$. Show that H_2 is convex.
29. Show that a hyperplane H in R^n is convex (Theorem 1.2).
30. Show that the intersection of a finite collection of convex sets is convex (Theorem 1.3).
31. Give two proofs of Theorem 1.4. One proof should use the definition of convex sets and the other should use Theorem 1.3.
32. Consider the linear programming problem
    Maximize $z = \mathbf{c}^T\mathbf{x}$
    subject to
    $$\mathbf{A}\mathbf{x} \leq \mathbf{b}$$
    $$\mathbf{x} \geq \mathbf{0}.$$
    Let $\mathbf{x}_1$ and $\mathbf{x}_2$ be feasible solutions to the problem. Show that, if the objective function has the value k at both $\mathbf{x}_1$ and $\mathbf{x}_2$, then it has the value k at any point on the line segment joining $\mathbf{x}_1$ and $\mathbf{x}_2$.
33. Show that the set of all solutions to $\mathbf{A}\mathbf{x} \leq \mathbf{b}$, if it is nonempty, is a convex set.
34. Show that the set of solutions to $\mathbf{A}\mathbf{x} > \mathbf{b}$, if it is nonempty, is a convex set.
35. A function mapping R^n into R^m is called a **linear transformation** if $f(\mathbf{u} + \mathbf{v}) = f(\mathbf{u}) + f(\mathbf{v})$, for any $\mathbf{u}$ and $\mathbf{v}$ in R^n, and $f(r\mathbf{u}) = rf(\mathbf{u})$, for any $\mathbf{u}$ in R^n and r in R. Prove that if S is a convex set in R^n and f is a linear transformation mapping R^n into R^m, then $f(S) = \{f(\mathbf{v}) \mid \mathbf{v} \in S\}$ is a convex set. A function f defined on a convex set S in R^n is called a **convex function** if
    $$f(\lambda\mathbf{x}_1 + (1 - \lambda)\mathbf{x}_2) \leq \lambda f(\mathbf{x}_1) + (1 - \lambda)f(\mathbf{x}_2)$$
    for $0 \leq \lambda \leq 1$ and any $\mathbf{x}_1, \mathbf{x}_2 \in S$.
36. Show that a function f defined on a convex set S in R^n is convex if the line segment joining any two points $(\mathbf{x}_1, f(\mathbf{x}_1))$ and $(\mathbf{x}_2, f(\mathbf{x}_2))$ does not lie below its graph. (See Figure 1.12.)
37. Show that the objective function $z = \mathbf{c}^T\mathbf{x}$ of a linear programming problem is a convex function.

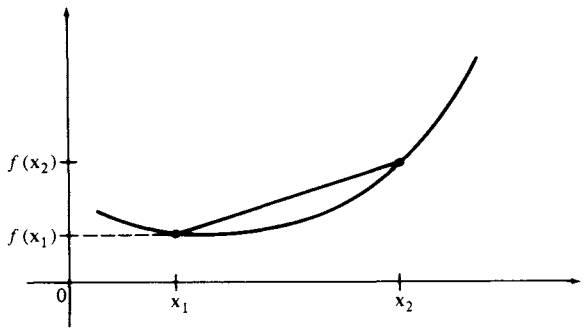


FIGURE 1.12

1.4 THE EXTREME POINT THEOREM

We continue in this section toward our goal of understanding the geometry of a linear programming problem. We first combine the results of the last section to describe the geometry of a general linear programming problem and then introduce the concept of an extreme point, or corner point. These become candidates for solutions to the problem.

We now consider a general linear programming problem. The graph of each constraint defined by an inequality is a closed half-space. The graph of each constraint defined by an equality is a hyperplane, or intersection of two closed half-spaces. Thus, the set of all points that satisfy all the constraints of the linear programming problem is exactly the intersection of the closed half-spaces determined by the constraints. From the previous results we see that this set of points, if it is nonempty, is a convex set, because it is the intersection of a finite number of convex sets. In general, the intersection of a finite set of closed half-spaces is called a **convex polyhedron**, and thus, if it is not empty, the set of feasible solutions to a general linear programming problem is a convex polyhedron.

We now turn to describing the points at which an optimal solution to a general linear programming problem can occur. We first make the following definition.

DEFINITION. A point $\mathbf{x} \in R^n$ is a **convex combination** of the points $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_r$ in R^n if for some real numbers $c_1, c_2, \ldots, c_r$ which satisfy

$$\sum_{i=1}^{r} c_i = 1 \quad \text{and} \quad c_i \geq 0, \quad 1 \leq i \leq r,$$

we have

$$\mathbf{x} = \sum_{i=1}^{r} c_i \mathbf{x}_i.$$

THEOREM 1.5. *The set of all convex combinations of a finite set of points in R^n is a convex set.*

Proof. Exercise. △

DEFINITION. A point $\mathbf{u}$ in a convex set S is called an **extreme point** of S if it is not an interior point of any line segment in S. That is, $\mathbf{u}$ is an extreme point of S if there are no distinct points $\mathbf{x}_1$ and $\mathbf{x}_2$ in S such that

$$\mathbf{u} = \lambda \mathbf{x}_1 + (1 - \lambda)\mathbf{x}_2, \quad 0 < \lambda < 1.$$

EXAMPLE 1. The only extreme points of the convex set in Figure 1.13 are A, B, C, D, and E (verify). △

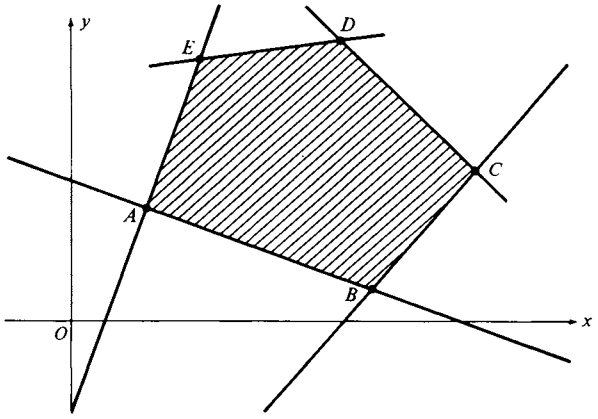


FIGURE 1.13

EXAMPLE 2. The extreme points of the convex sets shown in Figures 1.9 and 1.10 are given in the following table:

Figure	*Extreme points*
1.9a	*A, B, C, D*
1.9b	*A, B, C*
1.9c	The entire edge of the ellipse
1.9d	*A, B, C, D*
1.10a	*A*
1.10b	*O*
1.10c	None
1.10d	*O*

△

THEOREM 1.6. *Let S be a convex set in R^n. A point* **u** *in S is an extreme point of S if and only if* **u** *is not a convex combination of other points of S.*

Proof. Exercise. △

Since the set of all feasible solutions to a general linear programming problem is a convex polyhedron, it contains an infinite number of points. An optimal solution to the problem occurs at one of these points. But how do we find the right point among an infinite number? The following theorem, whose proof we do not give, shows that, if a linear programming problem has an optimal solution, then this solution must occur at an extreme point. Although an optimal solution can occur at a feasible solution that is not an extreme point, from the geometry of the situation it suffices to consider only extreme points. In fact, there are only a finite number of extreme points, but this number may be very large. In the next

chapter we show how to search through the extreme points in an orderly manner to find an optimal solution after a relatively small number of steps.

THEOREM 1.7 (EXTREME POINT). *Let S be the set of feasible solutions to a general linear programming problem.*

1. *If S is nonempty and bounded, then an optimal solution to the problem exists and occurs at an extreme point.*
2. *If S is nonempty and not bounded and if an optimal solution to the problem exists, then an optimal solution occurs at an extreme point.*
3. *If an optimal solution to the problem does not exist, then either S is empty or S is unbounded.* △

EXAMPLE 3. Consider the linear programming problem of Example 1 in Section 1.1

$$\begin{aligned} &\text{Maximize} \quad z = 120x + 100y \\ &\text{subject to} \\ &\qquad 2x + 2y \leq 8 \\ &\qquad 5x + 3y \leq 15 \\ &\qquad x \geq 0, \quad y \geq 0. \end{aligned}$$

The convex set of all feasible solutions is shown as the shaded region in Figure 1.14.

The extreme points of the convex set S are $(0,0)$, $(3,0)$, $(0,4)$, and $(\frac{3}{2}, \frac{5}{2})$. Since S is nonempty and bounded, the objective function attains its maximum at an extreme point of S (Theorem 1.7). We can find which extreme point is the optimal solution by evaluating the objective function at each extreme point. This evaluation is shown in Table 1.3. The maximum value of z occurs at the extreme point $(\frac{3}{2}, \frac{5}{2})$. Thus, an optimal solution is $x = \frac{3}{2}$ and $y = \frac{5}{2}$. In terms of the model this means that the lumber mill should saw 1500 board feet of finish-grade lumber and 2500

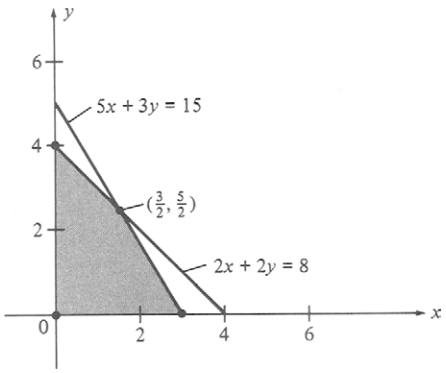


FIGURE 1.14

TABLE 1.3

Extreme point (x, y)	*Value of objective function* $z = 120x + 100y$
$(0, 0)$	0
$(3, 0)$	360
$(0, 4)$	400
$(\frac{3}{2}, \frac{5}{2})$	430

board feet of construction-grade lumber per day. These amounts will yield the maximum profit of \$430 per day. △

Some linear programming problems have no solution.

EXAMPLE 4. Consider the linear programming problem:

$$\text{Maximize} \quad z = 2x + 5y$$

subject to

$$2x + 3y \geq 12$$
$$3x + 4y \leq 12$$
$$x \geq 0, \quad y \geq 0.$$

The convex set of all feasible solutions consists of the points that lie in all four half-spaces defined by the constraints. The sketch of these half-spaces in Figure 1.15 shows that there are no such points. The set of feasible solutions is empty. This situation will arise when conflicting constraints are put on a problem. The assumptions for the model must be changed to yield a nonempty set of feasible solutions. △

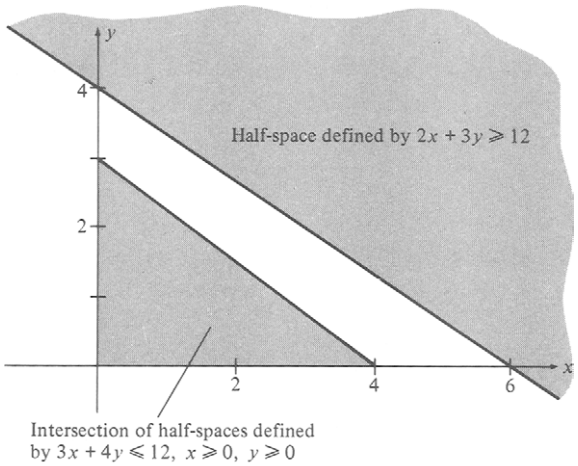


FIGURE 1.15

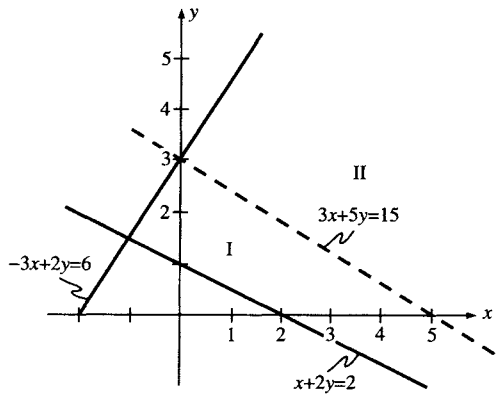


FIGURE 1.16

We have already seen in Example 7 in Section 1.3 that a linear programming problem with an unbounded convex set of feasible solutions may have no finite optimal value for the objective function. On the other hand, a linear programming problem with an unbounded convex set of feasible solutions may have an optimal solution.

EXAMPLE 5. In our previous discussion of this example (Example 8, Section 1.3), it seemed that the optimum value of the objective function was $z = 5$, which occurs when $x = 0$ and $y = 1$. We shall now show that this is the case.

We divide the set of feasible solutions into two regions with the arbitrarily chosen hyperplane $3x + 5y = 15$, as shown in Figure 1.16. All the points in region II satisfy $15 < 3x + 5y$, and all the points in region I satisfy $15 > 3x + 5y$. Thus, we need consider only the points in region I to solve the minimization problem, since it is only those points at which the objective function takes on values smaller than 15. Region I is closed and bounded and has a finite number of extreme points: $(0, 3)$, $(0, 1)$, $(2, 0)$, $(5, 0)$. Consequently, Theorem 1.7 applies. By evaluating the objective function at these four points, we find that the minimum value is $z = 5$ at $(0, 1)$. Note that other choices for the dividing hyperplane are possible. Δ

EXAMPLE 6. Consider the linear programming problem

$$\begin{aligned} &\text{Maximize} \quad z = 2x + 3y \\ &\text{subject to} \\ &\qquad x + 3y \leq 9 \\ &\qquad 2x + 3y \leq 12 \\ &\qquad x \geq 0, \quad y \geq 0. \end{aligned}$$

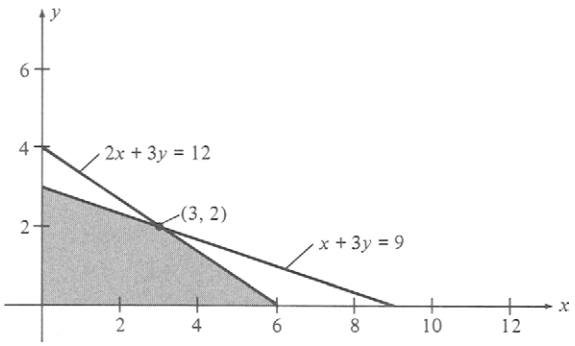


FIGURE 1.17

TABLE 1.4

Extreme point	*Value of* $z = 2x + 3y$
(0, 0)	0
(0, 3)	9
(6, 0)	12
(3, 2)	12

The convex set of all feasible solutions is shown in Figure 1.17. The extreme points and corresponding values of the objective function are given in Table 1.4. We see that both (6, 0) and (3, 2) are optimal solutions to the problem. The line segment joining these points is

$$\begin{aligned}(x, y) &= \lambda(6, 0) + (1 - \lambda)(3, 2) \\ &= (6\lambda, 0) + (3 - 3\lambda, 2 - 2\lambda) \\ &= (3 + 3\lambda, 2 - 2\lambda) \quad \text{for} \quad 0 \leq \lambda \leq 1.\end{aligned}$$

For any point (x, y) on this line segment we have

$$\begin{aligned}z = 2x + 3y &= 2(3 + 3\lambda) + 3(2 - 2\lambda) \\ &= 6 + 6\lambda + 6 - 6\lambda \\ &= 12.\end{aligned}$$

Any point on this line segment is an optimal solution. △

1.4 EXERCISES

In Exercises 1–12 (a) find the extreme points of the set of feasible solutions for the given linear programming program and (b) find the optimal solution(s).

1. Maximize $z = x + 2y$
subject to
$$3x + y \leq 6$$
$$3x + 4y \leq 12$$
$$x \geq 0, \quad y \geq 0.$$

2. Minimize $z = 5x - 3y$
subject to
$$x + 2y \leq 4$$
$$x + 3y \geq 6$$
$$x \geq 0, \quad y \geq 0.$$

3. Maximize $z = 3x + y$
subject to
$$-3x + y \geq 6$$
$$3x + 5y \leq 15$$
$$x \geq 0, \quad y \geq 0.$$

4. Maximize $z = 2x + 3y$
subject to
$$3x + y \leq 6$$
$$x + y \leq 4$$
$$x + 2y \leq 6$$
$$x \geq 0, \quad y \geq 0.$$

5. Minimize $z = 3x + 5y$
subject to the same constraints as those in Exercise 4.

6. Maximize $z = \frac{1}{2}x + \frac{3}{2}y$
subject to
$$x + 3y \leq 6$$
$$x + y \geq 4$$
$$x \geq 0, \quad y \geq 0.$$

7. Maximize $z = 2x + 5y$
subject to
$$2x + y \geq 2$$
$$x + y \leq 8$$
$$x + y \geq 3$$
$$2x + y \leq 12$$
$$x \geq 0, \quad y \geq 0.$$

8. Maximize $z = 2x_1 + 4x_2$
subject to
$$5x_1 + 3x_2 + 5x_3 \leq 15$$
$$10x_1 + 8x_2 + 15x_3 \leq 40$$
$$x_1 \geq 0, \quad x_2 \geq 0, \quad x_3 \geq 0.$$

9. Maximize $z = 2x_1 + 4x_2 + 3x_3$
subject to
$$x_1 + x_2 + x_3 \leq 12$$
$$x_1 + 3x_2 + 3x_3 \leq 24$$
$$3x_1 + 6x_2 + 4x_3 \leq 90$$
$$x_1 \geq 0, \quad x_2 \geq 0, \quad x_3 \geq 0.$$

10. Minimize $z = 2x_1 + 3x_2 + x_3$
subject to the same constraints as those in Exercise 9.

11. Maximize $z = 5x_1 + 2x_2 + 3x_3$
subject to
$$x_1 + x_2 + x_3 = 1$$
$$2x_1 + 5x_2 + 3x_3 \leq 4$$
$$4x_1 + x_2 + 3x_3 \leq 2$$
$$x_1 \geq 0, \quad x_2 \geq 0, \quad x_3 \geq 0.$$

12. Minimize $z = 2x_1 + x_3$
subject to
$$x_1 + x_2 + x_3 = 1$$
$$2x_1 + x_2 + 2x_3 \geq 3$$
$$x_1 \geq 0, \quad x_2 \geq 0, \quad x_3 \geq 0.$$

13. Prove Theorem 1.5.

14. Prove Theorem 1.6.

15. Show that a set S in R^n is convex if and only if every convex combination of a finite number of points in S is in S.

16. Show that if the optimal value of the objective function of a linear programming problem is attained at several extreme points, then it is also attained at any convex combination of these extreme points.

1.5 BASIC SOLUTIONS

In this section we connect the geometric ideas of Section 1.3 and Section 1.4 with the algebraic notions developed in Section 1.2. We have already seen the important role played by the extreme points of the set of feasible solutions in obtaining an optimal solution to a linear programming problem. However, the extreme points are difficult to compute geometrically when there are more than three variables in the problem. In this section we give an algebraic description of extreme points that will facilitate their computation. This description uses the concept of a basic solution to a linear programming problem. To lay the foundation for the definition of a basic solution, we shall now prove two very important general theorems about linear programming problems in canonical form.

Consider the linear programming problem in canonical form

$$\text{Maximize} \quad z = \mathbf{c}^{\mathrm{T}}\mathbf{x} \tag{1}$$

subject to

$$\mathbf{A}\mathbf{x} = \mathbf{b} \tag{2}$$

$$\mathbf{x} \geq \mathbf{0}, \tag{3}$$

where $\mathbf{A}$ is an $m \times s$ matrix, $\mathbf{c} \in R^s$, $\mathbf{x} \in R^s$, and $\mathbf{b} \in R^m$. Let the columns of $\mathbf{A}$ be denoted by $\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_s$. We can then write (2) as

$$x_1\mathbf{A}_1 + x_2\mathbf{A}_2 + \cdots + x_s\mathbf{A}_s = \mathbf{b}. \tag{4}$$

We make two assumptions about the constraint matrix $\mathbf{A}$. We assume that $m \leq s$ and that there are m columns of $\mathbf{A}$ that are linearly independent. That is, the rank of $\mathbf{A}$ is m. These assumptions are true for a linear programming problem in canonical form that arose from a problem in standard form as given in Equations (4), (5), and (6) in Section 1.2.

This set of m columns, and indeed any set of m linearly independent columns of $\mathbf{A}$, forms a basis for R^m. We can always renumber the columns of $\mathbf{A}$ (by reordering the components of $\mathbf{x}$), so that the last m columns of $\mathbf{A}$ are linearly independent. Let S be the convex set of all feasible solutions to the problem determined by (1), (2), and (3).

THEOREM 1.8. *Suppose that the last m columns of* $\mathbf{A}$, *which we denote by* $\mathbf{A}'_1, \mathbf{A}'_2, \ldots, \mathbf{A}'_m$, *are linearly independent and suppose that*

$$x'_1\mathbf{A}'_1 + x'_2\mathbf{A}'_2 + \cdots + x'_m\mathbf{A}'_m = \mathbf{b}, \tag{5}$$

where $x'_i \geq 0$ *for* $i = 1, 2, \ldots, m$. *Then the point*

$$\mathbf{x} = (0, 0, \ldots, 0, x'_1, x'_2, \ldots, x'_m)$$

is an extreme point of S.

Proof. We assumed $\mathbf{x} \geq 0$ in the statement of the theorem. Equation (5) represents $\mathbf{Ax} = \mathbf{b}$, since the first $s - m$ components of $\mathbf{x}$ are zero. Thus, $\mathbf{x}$ is a feasible solution to the linear programming problem given by (1), (2), and (3). Assume that $\mathbf{x}$ is not an extreme point of S. Then, $\mathbf{x}$ lies in the interior of a line segment in S. That is, there are points $\mathbf{v}$ and $\mathbf{w}$ in S both different from $\mathbf{x}$ and a number λ, $0 < \lambda < 1$, such that

$$\mathbf{x} = \lambda\mathbf{v} + (1 - \lambda)\mathbf{w}. \tag{6}$$

Now

$$\mathbf{v} = (v_1, v_2, \ldots, v_{s-m}, v'_1, v'_2, \ldots, v'_m)$$

and

$$\mathbf{w} = (w_1, w_2, \ldots, w_{s-m}, w'_1, w'_2, \ldots, w'_m),$$

where all the components of $\mathbf{v}$ and $\mathbf{w}$ are nonnegative, since $\mathbf{v}$ and $\mathbf{w}$ are feasible solutions. Substituting the expressions for $\mathbf{x}$, $\mathbf{v}$, and $\mathbf{w}$ into (6), we have

$$\begin{aligned} 0 &= \lambda v_i + (1 - \lambda)w_i, \quad 1 \leq i \leq s - m \\ x'_j &= \lambda v'_j + (1 - \lambda)w'_j, \quad 1 \leq j \leq m. \end{aligned} \tag{7}$$

Since all the terms in (7) are nonnegative and λ and $1 - \lambda$ are positive, we conclude that $v_i = 0$ and $w_i = 0$ for $i = 1, 2, \ldots, s - m$. Since $\mathbf{v}$ is a feasible solution, we know that $\mathbf{Av} = \mathbf{b}$ and, because the first $s - m$ components of $\mathbf{v}$ are zero, this equation can be written as

$$v'_1\mathbf{A}'_1 + v'_2\mathbf{A}'_2 + \cdots + v'_m\mathbf{A}'_m = \mathbf{b}. \tag{8}$$

If we now subtract Equation (8) from Equation (5), we have

$$(x'_1 - v'_1)\mathbf{A}'_1 + (x'_2 - v'_2)\mathbf{A}'_2 + \cdots + (x'_m - v'_m)\mathbf{A}'_m = \mathbf{0}.$$

Since we assumed that $\mathbf{A}'_1, \mathbf{A}'_2, \ldots, \mathbf{A}'_m$ were linearly independent, we conclude that

$$x'_i = v'_i \quad \text{for} \quad 1 \leq i \leq m$$

and consequently that $\mathbf{x} = \mathbf{v}$. But we had assumed $\mathbf{x} \neq \mathbf{v}$. This contradiction implies that our assumption that $\mathbf{x}$ is not an extreme point of S is false. Thus, $\mathbf{x}$ is an extreme point of S, as we wanted to show. △

THEOREM 1.9. *If* $\mathbf{x} = (x_1, x_2, \ldots, x_s)$ *is an extreme point of* S, *then the columns of* $\mathbf{A}$ *that correspond to positive* x_j *form a linearly independent set of vectors in* R^m.

Proof. To simplify the notation, renumber the columns of $\mathbf{A}$ and the components of $\mathbf{x}$, if necessary, so that the last k components, denoted by $x'_1, x'_2, \ldots, x'_k$, are positive. Thus, Equation (4) can be written as

$$x'_1\mathbf{A}'_1 + x'_2\mathbf{A}'_2 + \cdots + x'_k\mathbf{A}'_k = \mathbf{b}. \tag{9}$$

We must show that $\mathbf{A}'_1, \mathbf{A}'_2, \ldots, \mathbf{A}'_k$ are linearly independent. Suppose they are linearly dependent. This means that there are numbers c_j, $1 \leq j \leq k$, not all of which are zero, such that

$$c_1\mathbf{A}'_1 + c_2\mathbf{A}'_2 + \cdots + c_k\mathbf{A}'_k = \mathbf{0}. \tag{10}$$

Say that $c_t \neq 0$. Multiply Equation (10) by a positive scalar d, and first add the resulting equation to Equation (9), getting Equation (11), and second subtract it from Equation (9), getting Equation (12). We now have

$$(x'_1 + dc_1)\mathbf{A}'_1 + (x'_2 + dc_2)\mathbf{A}'_2 + \cdots + (x'_k + dc_k)\mathbf{A}'_k = \mathbf{b} \tag{11}$$

$$(x'_1 - dc_1)\mathbf{A}'_1 + (x'_2 - dc_2)\mathbf{A}'_2 + \cdots + (x'_k - dc_k)\mathbf{A}'_k = \mathbf{b}. \tag{12}$$

Now consider the points in R^s,

$$\mathbf{v} = (0, 0, \ldots, 0, x'_1 + dc_1, x'_2 + dc_2, \ldots, x'_k + dc_k)$$

and

$$\mathbf{w} = (0, 0, \ldots, 0, x'_1 - dc_1, x'_2 - dc_2, \ldots, x'_k - dc_k).$$

Since d is any positive scalar, we may choose it so that

$$0 < d < \min_j \frac{x'_j}{|c_j|}, \quad c_j \neq 0.$$

With this choice of d, the last k coordinates of both $\mathbf{v}$ and $\mathbf{w}$ are positive. This fact together with Equations (11) and (12) implies that $\mathbf{v}$ and $\mathbf{w}$ are feasible solutions. But we also have

$$\mathbf{x} = \tfrac{1}{2}\mathbf{v} + \tfrac{1}{2}\mathbf{w},$$

contradicting the hypothesis that $\mathbf{x}$ is an extreme point of S. Thus our assumption that the last k columns of $\mathbf{A}$ are linearly dependent is false; they are linearly independent. △

COROLLARY 1.1. *If* $\mathbf{x}$ *is an extreme point and* $x_{i_1}, \ldots, x_{i_r}$ *are the* r *positive components of* $\mathbf{x}$, *then* $r \leq m$, *and the set of columns* $\mathbf{A}_{i_1}, \ldots, \mathbf{A}_{i_r}$ *can be extended to a set of* m *linearly independent vectors in* R^m *by adjoining a suitably chosen set of* $m - r$ *columns of* $\mathbf{A}$.

Proof. Exercise. △

THEOREM 1.10. *At most* m *components of any extreme point of* S *can be positive. The rest must be zero.*

Proof. Theorem 1.9 says that the columns of **A** corresponding to the positive components of an extreme point **x** of the set S of feasible solutions are linearly independent vectors in R^m. But there can be no more than m linearly independent vectors in R^m. Therefore, at most m of the components of **x** are nonzero. △

An important feature of the canonical form of a linear programming problem is that the constraints $\mathbf{Ax} = \mathbf{b}$ form a system of m equations in s unknowns. Our assumption that there are m linearly independent column vectors means that the rank of **A** is m, so the rows of **A** are also linearly independent. That is, redundant equations do not occur in the system of constraints. In Theorems 1.8 and 1.9 we showed the relationships between the extreme points of the set S of feasible solutions and the linearly independent columns of **A**. We now use information about solutions to a system of equations to describe further the points of S. Note that S is just the set of solutions to $\mathbf{Ax} = \mathbf{b}$ with nonnegative components ($\mathbf{x} \geq \mathbf{0}$).

Consider a system of m equations in s unknowns ($m \leq s$) and write it in matrix form as $\mathbf{Ax} = \mathbf{b}$. Assume that at least one set of m columns of **A** is linearly independent. Choosing any set T of m linearly independent columns of **A** (which is choosing a basis for R^m), set the $s - m$ variables corresponding to the remaining columns equal to zero. The equation $\mathbf{Ax} = \mathbf{b}$ may be written as

$$x_1\mathbf{A}_1 + x_2\mathbf{A}_2 + \cdots + x_s\mathbf{A}_s = \mathbf{b}, \tag{13}$$

where $\mathbf{A}_i$ is the ith column of **A**. But we have set $s - m$ of the variables equal to zero. Let $i_1, i_2, \ldots, i_m$ be the indices of the variables that were not set to zero. They are also the indices of the columns of **A** in the set T. Consequently, (13) reduces to

$$x_{i_1}\mathbf{A}_{i_1} + x_{i_2}\mathbf{A}_{i_2} + \cdots + x_{i_m}\mathbf{A}_{i_m} = \mathbf{b},$$

which is a system of m equations in m unknowns and has a unique solution. (Why?) The values of the m variables obtained from solving this system along with the $s - m$ zeros form a vector **x** that is called a **basic solution** to $\mathbf{Ax} = \mathbf{b}$.

EXAMPLE 1. Consider the system of three equations in six unknowns

$$\begin{bmatrix} 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & -1 & -1 & 0 & -1 & -1 \\ 1 & 2 & 2 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}.$$

Setting $x_2 = x_3 = x_5 = 0$, we get the system of three equations in three unknowns given by

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & -1 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_4 \\ x_6 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}.$$

The columns of this coefficient matrix are linearly independent, and this system has the solution $x_1 = b_1$, $x_4 = b_2 + b_3 - b_1$, $x_6 = -b_2$. Consequently, a basic solution to the original system is

$$\mathbf{x} = (b_1, 0, 0, b_2 + b_3 - b_1, 0, -b_2).$$

On the other hand, if we set $x_1 = x_3 = x_5 = 0$, we obtain the system

$$\begin{bmatrix} 0 & 0 & 0 \\ -1 & 0 & -1 \\ 2 & 1 & 1 \end{bmatrix} \begin{bmatrix} x_2 \\ x_4 \\ x_6 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}.$$

Here the columns of the coefficient matrix are not linearly independent. In fact, column 1 is the sum of columns 2 and 3. This system cannot be solved if $b_1 \neq 0$. Consequently, this choice of variables does not lead to a basic solution. △

In any basic solution, the $s - m$ variables that are set equal to zero are called **nonbasic variables**, and the m variables solved for are called **basic variables**. Although the term *basic solution* appears in all the literature describing linear programming, it can be misleading. A basic solution is a solution to the system $\mathbf{Ax} = \mathbf{b}$; it does not necessarily satisfy $\mathbf{x} \geq \mathbf{0}$, and therefore it is not necessarily a feasible solution to the linear programming problem given by (1), (2), and (3).

DEFINITION. A **basic feasible solution** to the linear programming problem given by (1), (2), and (3) is a basic solution that is also feasible.

THEOREM 1.11. *For the linear programming problem determined by* (1), (2), *and* (3), *every basic feasible solution is an extreme point, and, conversely, every extreme point is a basic feasible solution.*

Proof. Exercise. △

THEOREM 1.12. *The problem determined by* (1), (2), *and* (3) *has a finite number of basic feasible solutions.*

Proof. The number of basic solutions to the problem is

$$\binom{s}{m} = \frac{s!}{m!(s-m)!} = \binom{s}{s-m}$$

because $s - m$ variables must be chosen to be set to zero, out of a total of s variables. The number of basic feasible solutions may be smaller than the number of basic solutions, since not all basic solutions need to be feasible. △

We now examine the relationship between the set S of feasible solutions to a standard linear programming problem given in Equations (4), (5), and (6) in Section 1.2 and the set S' of feasible solutions to the associated canonical linear programming problem given in Equations (12), (13), and (14) in Section 1.2. We have already discussed the method of adding slack variables to go from a point in S to a point in S'. Conversely, we truncated variables to move from S' to S. More specifically, we have the following theorem, whose proof we leave as an exercise.

THEOREM 1.13. *Every extreme point of S yields an extreme point of S' when slack variables are added. Conversely, every extreme point of S', when truncated, yields an extreme point of S.*

Proof. Exercise. △

THEOREM 1.14. *The convex set S of all feasible solutions to a linear programming problem in standard form has a finite number of extreme points.*

Proof. Exercise. △

By combining the Extreme Point Theorem (Theorem 1.7) and Theorem 1.14 we can give a procedure for solving a standard linear programming problem given by Equations (4), (5), and (6) in Section 1.2. First, set up the associated canonical form of the problem. Then find all basic solutions and eliminate those that are not feasible. Find those that are optimal among the basic feasible solutions. Since the objective function does not change between the standard and canonical forms of the problem, any optimal solution to the canonical form, found as described above, is an optimal solution to the standard problem.

From the situation under consideration, the number of basic solutions to be examined is no more than

$$\binom{m+n}{n}.$$

Although this number is finite, it is still too large for actual problems. For example, a moderate-size problem with $m = 200$ and $n = 300$ would have about 10^{144} basic solutions.

EXAMPLE 2. Consider the linear programming problem given in Example 3 in Section 1.2. In this example we can select two of the four variables x, y, u, v as nonbasic variables by setting them equal to zero and then

solve for the basic variables. If in

$$\begin{bmatrix} 2 & 2 & 1 & 0 \\ 5 & 3 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ u \\ v \end{bmatrix} = \begin{bmatrix} 8 \\ 15 \end{bmatrix}$$

we set

$$x = y = 0 \quad \text{(nonbasic variables)},$$

then

$$u = 8 \quad \text{and} \quad v = 15 \quad \text{(basic variables)}.$$

The vector $[0 \quad 0 \quad 8 \quad 15]^T$ is a basic feasible solution to the canonical form of the problem and hence an extreme point of S'. By truncating the slack variables we get $[0 \quad 0]^T$, which is an extreme point of S and a feasible solution to the standard form of the problem.

If instead we set

$$x = v = 0 \quad \text{(nonbasic variables)},$$

then

$$y = 5 \quad \text{and} \quad u = -2 \quad \text{(basic variables)}.$$

The vector $[0 \quad 5 \quad -2 \quad 0]^T$ is a basic solution, but it is not feasible, since u is negative.

In Table 1.5 we list all the basic solutions, indicate whether they are feasible, and give the truncated vectors. The student should locate these truncated vectors on Figure 1.14. Once we discard the basic solutions that are not feasible, we select a basic feasible solution for which the objective function is a maximum. Thus, we again obtain the optimal solution

$$x = \tfrac{3}{2}, \quad y = \tfrac{5}{2}, \quad u = 0, \quad v = 0$$

TABLE 1.5

x	y	u	v	*Type of solution*	*Value of* $z = 120x + 100y$	*Truncated vector*
0	0	8	15	Basic feasible	0	(0, 0)
0	4	0	3	Basic feasible	400	(0, 4)
0	5	−2	0	Basic, not feasible	—	(0, 5)
4	0	0	−5	Basic, not feasible	—	(4, 0)
3	0	2	0	Basic feasible	360	(3, 0)
$\frac{3}{2}$	$\frac{5}{2}$	0	0	Basic feasible	430	$(\frac{3}{2}, \frac{5}{2})$

1.5 EXERCISES

1. Suppose the canonical form of a liner programming problem is given by the constraint matrix **A** and resource vector **b**, where

$$\mathbf{A} = \begin{bmatrix} 3 & 0 & 1 & 1 & 0 \\ 2 & 1 & 0 & 0 & 0 \\ 4 & 0 & 3 & 0 & 1 \end{bmatrix} \quad \text{and} \quad \mathbf{b} = \begin{bmatrix} 5 \\ 3 \\ 6 \end{bmatrix}.$$

Determine which of the following points is

(i) a feasible solution to the linear programming problem.
(ii) an extreme point of the set of feasible solutions.
(iii) a basic solution.
(iv) a basic feasible solution.

For each basic feasible solution **x** given below, list the basic variables.

$$\text{(a)} \begin{bmatrix} 0 \\ 3 \\ 0 \\ 5 \\ 6 \end{bmatrix} \quad \text{(b)} \begin{bmatrix} 0 \\ 3 \\ 5 \\ 0 \\ -9 \end{bmatrix} \quad \text{(c)} \begin{bmatrix} \frac{3}{2} \\ 0 \\ 0 \\ \frac{1}{2} \\ 0 \end{bmatrix} \quad \text{(d)} \begin{bmatrix} \frac{1}{2} \\ 1 \\ 1 \\ 0 \\ 2 \end{bmatrix} \quad \text{(e)} \begin{bmatrix} 1 \\ 1 \\ \frac{1}{2} \\ \frac{3}{2} \\ \frac{1}{2} \end{bmatrix}$$

In Exercises 2 and 3, set up a linear programming model for the situation described. Sketch the set of feasible solutions and find an optimal solution by examining the extreme points.

2. The Savory Potato Chip Company makes pizza-flavored and chili-flavored potato chips. These chips must go through three main processes: frying, flavoring, and packing. Each kilogram of pizza-flavored chips takes 3 min to fry, 5 min to flavor, and 2 min to pack. Each kilogram of chili-flavored chips takes 3 min to fry, 4 min to flavor, and 3 min to pack. The net profit on each kilogram of pizza chips is \$0.12, whereas the net profit on each kilogram of chili chips is \$0.10. The fryer is available 4 hr each day, the flavorer is available 8 hr each day, and the packer is available 6 hr each day. Maximize the net profit with your model.
3. Sugary Donut Bakers, Inc., is known for its excellent glazed doughnuts. The firm also bakes doughnuts, which are then dipped in powdered sugar. It makes a profit of \$0.07 per glazed doughnut and \$0.05 per powdered sugar doughnut. The three main operations are baking, dipping (for the powdered sugar doughnuts only), and glazing (for the glazed doughnuts only). Each day the plant has the capacity to bake 1400 doughnuts, dip 1200 doughnuts, and glaze 1000 doughnuts. The manager has instructed that at least 600 glazed doughnuts must be made each day. Maximize the total profit with your model.
4. For Exercise 2, write the linear programming problem in canonical form, compute the values of the slack variables for an optimal solution, and give a physical interpretation for these values. Also identify the basic variables of the optimal solution.

5. For Exercise 3, write the linear programming problem in canonical form, compute the values of the slack variables for an optimal solution, and give a physical interpretation for these values. Also identify the basic variables of the optimal solution.
6. Consider the system of equations $\mathbf{Ax} = \mathbf{b}$, where

$$\mathbf{A} = \begin{bmatrix} 2 & 3 & 4 & 0 & 4 \\ 1 & 0 & 0 & -2 & 1 \end{bmatrix} \quad \text{and} \quad \mathbf{b} = \begin{bmatrix} 2 \\ 0 \end{bmatrix}.$$

Determine whether each of the following 5-tuples is a basic solution to the system.

(a) $(1, 0, 1, 0, 0)$ (b) $(0, 2, -1, 0, 0)$
(c) $(2, -2, 3, 0, -2)$ (d) $(0, 0, \frac{1}{2}, 0, 0)$

7. Consider the system of equations $\mathbf{Ax} = \mathbf{b}$, where

$$\mathbf{A} = \begin{bmatrix} 2 & 3 & 1 & 0 & 0 \\ -1 & 1 & 0 & 2 & 1 \\ 0 & 6 & 1 & 0 & 3 \end{bmatrix} \quad \text{and} \quad \mathbf{b} = \begin{bmatrix} 1 \\ 1 \\ 4 \end{bmatrix}.$$

Determine which of the following 5-tuples are basic solutions to the system. Give reasons.

(a) $(1, 0, -1, 1, 0)$
(b) $(0, 2, -5, 0, -1)$
(c) $(0, 0, 1, 0, 1)$

8. Consider the linear programming problem

$$\begin{aligned} &\text{Maximize} \quad z = 3x + 2y \\ &\text{subject to} \\ &\qquad 2x - y \leq 6 \\ &\qquad 2x + y \leq 10 \\ &\qquad x \geq 0, \quad y \geq 0. \end{aligned}$$

(a) Transform this problem to a problem in canonical form.
(b) For each extreme point of the new problem, identify the basic variables.
(c) Solve the problem geometrically.

9. Consider the linear programming problem

$$\begin{aligned} &\text{Maximize} \quad z = 4x_1 + 2x_2 + 7x_3 \\ &\text{subject to} \\ &\qquad 2x_1 - x_2 + 4x_3 \leq 18 \\ &\qquad 4x_1 + 2x_2 + 5x_3 \leq 10 \\ &\qquad x_1 \geq 0, \quad x_2 \geq 0, \quad x_3 \geq 0. \end{aligned}$$

(a) Transform this problem to a problem in canonical form.
(b) For each extreme point of the new problem, identify the basic variables.
(c) Which of the extreme points are optimal solutions to the problem?

10. Consider the linear programming in standard form

$$\begin{aligned} &\text{Maximize} \quad z = \mathbf{c}^T\mathbf{x} \\ &\text{subject to} \\ &\qquad \mathbf{Ax} \leq \mathbf{b} \\ &\qquad \mathbf{x} \geq \mathbf{0}. \end{aligned}$$

Show that the constraints $\mathbf{Ax} \leq \mathbf{b}$ may be written as

(i) $[\mathbf{A} \mid \mathbf{I}]\begin{bmatrix}\mathbf{x} \\ \mathbf{x}'\end{bmatrix} = \mathbf{b}$

or as

(ii) $\mathbf{Ax} + \mathbf{Ix}' = \mathbf{b}$,

where $\mathbf{x}'$ is a vector of slack variables.

11. Prove Corollary 1.1 (*Hint:* Use Theorem 0.13.)
12. Prove for a linear programming problem in canonical form that a point in the set of feasible solutions is an extreme point if and only if it is a basic feasible solution (Theorem 1.11). (*Hint:* Use Theorem 0.13.)
13. Prove that the set of feasible solutions to a linear programming problem in standard form has a finite number of extreme points (Theorem 1.14).

Further Reading

Chvátal, Vašek. *Linear Programming*. Freeman, New York, 1980.
Grünbaum, B. *Convex Polytopes*. Wiley–Interscience, New York, 1967.
Hadley, G. *Linear Algebra*. Addison–Wesley, Reading, MA, 1961.
Murty, Katta G. *Linear Programming*. Wiley, New York, 1983.
Taha, Hamdy A. *Operations Research: An Introduction*, third ed., Macmillan, New York, 1982.

2

The Simplex Method

IN THIS CHAPTER we describe an elementary version of the method that can be used to solve a linear programming problem systematically. In Chapter 1 we developed the algebraic and geometric notions that allowed us to characterize the solutions to a linear programming problem. However, for problems of more than three variables, the characterization did not lead to a practical method for actually finding the solutions. We know that the solutions are extreme points of the set of feasible solutions. The method that we present determines the extreme points in the set of feasible solutions in a particular order that allows us to find an optimal solution in a small number of trials. We first consider problems in standard form because when applying the method to these problems it is easy to find a starting point. The second section discusses a potential pitfall with the method. However, the difficulty rarely arises and has almost never been found when solving practical problems. In the third section, we extend the method to arbitrary linear programming problems by developing a way of constructing a starting point.

2.1 THE SIMPLEX METHOD FOR PROBLEMS IN STANDARD FORM

We already know from Section 1.5 that a linear programming problem in canonical form can be solved by finding all the basic solutions, discarding those that are not feasible, and finding an optimal solution among the remaining. Since this procedure can still be a lengthy one, we seek a more efficient method for solving linear programming problems. The simplex algorithm is such a method; in this section we shall describe and carefully illustrate it. Even though the method is an algebraic one, it is helpful to examine it geometrically.

Consider a linear programming problem in standard form

$$\text{Maximize} \quad z = \mathbf{c}^T\mathbf{x} \tag{1}$$

subject to

$$\mathbf{Ax} \leq \mathbf{b} \tag{2}$$

$$\mathbf{x} \geq \mathbf{0}, \tag{3}$$

where $\mathbf{A} = [a_{ij}]$ is an $m \times n$ matrix and

$$\mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}, \quad \mathbf{c} = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \end{bmatrix}, \quad \text{and} \quad \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}.$$

In this section we shall make the additional assumption that $\mathbf{b} \geq \mathbf{0}$. In Section 2.3 we will describe a procedure for handling problems in which $\mathbf{b}$ is not nonnegative.

We now transform each of the constraints in (2) into an equation by introducing a slack variable. We obtain the canonical form of the problem, namely

$$\text{Maximize} \quad z = \mathbf{c}^T\mathbf{x} \tag{4}$$

subject to

$$\mathbf{Ax} = \mathbf{b} \tag{5}$$

$$\mathbf{x} \geq \mathbf{0}, \tag{6}$$

where in this case $\mathbf{A}$ is the $m \times (n + m)$ matrix

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} & 1 & 0 & \cdots & 0 \\ a_{21} & a_{22} & \cdots & a_{2n} & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & & \vdots & \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} & 0 & 0 & \cdots & 1 \end{bmatrix}$$

$$\mathbf{c} = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \quad \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \\ x_{n+1} \\ \vdots \\ x_{n+m} \end{bmatrix},$$

and **b** is as before.

Recall from Section 1.5 that a basic feasible solution to the canonical form of the problem (4), (5), (6) is an extreme point of the convex set S' of all feasible solutions to the problem.

DEFINITION. Two distinct extreme points in S' are said to be **adjacent** if as basic feasible solutions they have all but one basic variable in common. △

EXAMPLE 1. Consider Example 2 of Section 1.5 and especially Table 1.5 in that example. The extreme points $(0, 0, 8, 15)$ and $(0, 4, 0, 3)$ are adjacent, since the basic variables in the first extreme point are u and v and the basic variables in the second extreme point are y and v. In fact, the only extreme point that is not adjacent to $(0, 0, 8, 15)$ is $(\frac{3}{2}, \frac{5}{2}, 0, 0)$. △

The simplex method developed by George B. Dantzig in 1947 is a method that proceeds from a given extreme point (basic feasible solution) to an adjacent extreme point in such a way that the value of the objective function increases or, at worst, remains the same. The method proceeds until we either obtain an optimal solution or find that the given problem has no finite optimal solution. The simplex algorithm consists of two steps: (1) a way of finding out whether a given basic feasible solution is an optimal solution and (2) a way of obtaining an adjacent basic feasible solution with the same or larger value for the objective function. In actual use, the simplex method does not examine every basic feasible solution; it checks only a relatively small number of them. However, examples have been given in which a large number of basic feasible solutions have been examined by the simplex method.

We shall demonstrate parts of our description of the simplex method on the linear programming problem in Example 1 of Section 1.1. The associated canonical form of the problem was described in Example 4 of Section

1.2. In this form it is:

$$\text{Maximize} \quad z = 120x + 100y \tag{7}$$

subject to

$$\left.\begin{aligned} 2x + 2y + u &= 8 \\ 5x + 3y + v &= 15 \end{aligned}\right\} \tag{8}$$

$$x \geq 0, \quad y \geq 0, \quad u \geq 0, \quad v \geq 0. \tag{9}$$

The Initial Basic Feasible Solution

To start the simplex method, we must find a basic feasible solution. The assumption that $\mathbf{b} \geq \mathbf{0}$ allows the following procedure to work. If it is not true that $\mathbf{b} \geq \mathbf{0}$, another procedure (discussed in Section 2.3) must be used. We take all the nonslack variables as nonbasic variables; that is, we set all the nonslack variables in the system $\mathbf{Ax} = \mathbf{b}$ equal to zero. The basic variables are then just the slack variables. We have

$$x_1 = x_2 = \cdots = x_n = 0 \quad \text{and} \quad x_{n+1} = b_1, \quad x_{n+2} = b_2, \ldots, x_{n+m} = b_m.$$

This is a feasible solution, since $\mathbf{b} \geq \mathbf{0}$; and it is a basic solution, since $(n + m) - m = n$ of the variables are zero.

In our example, we let

$$x = y = 0.$$

Solving for u and v, we obtain

$$u = 8, \quad v = 15.$$

The initial basic feasible solution constructed by this method is $(0, 0, 8, 15)$. The basic feasible solution yields the extreme point $(0, 0)$ in Figure 1.14 (Section 1.4).

It is useful to set up our example and its initial basic feasible solution in tabular form. To do this, we write (7) as

$$-120x - 100y + z = 0, \tag{10}$$

where z is now viewed as another variable. The **initial tableau** is now formed (Tableau 2.1). At the top we list the variables x, y, u, v, and z as labels on the corresponding columns. The last row, called the **objective row**, is Equation (10). The constraints (8) are on the first two rows. Along the left side of each row we indicate which variable is basic in the corresponding equation. Thus, in the first equation u is the basic variable, and v is the basic variable in the second equation.

Tableau 2.1

	x	y	u	v	z	
u	2	2	1	0	0	8
v	5	3	0	1	0	15
	−120	−100	0	0	1	0

In the tableau, a basic variable has the following properties:

1. It appears in exactly one equation and in that equation it has a coefficient of $+1$.
2. The column that it labels has all zeros (including the objective row entry) except for the $+1$ in the row that is labeled by the basic variable.
3. The value of a basic variable is the entry in the same row in the rightmost column.

The initial tableau for the general problem (4), (5), (6) is shown in Tableau 2.2. The value of the objective function

$$z = c_1x_1 + c_2x_2 + \cdots + c_nx_n + 0 \cdot x_{n+1} + \cdots + 0 \cdot x_{n+m}$$

for the initial basic feasible solution is

$$z = c_1 \cdot 0 + c_2 \cdot 0 + \cdots + c_n \cdot 0 + 0 \cdot b_1 + 0 \cdot b_2 + \cdots + 0 \cdot b_m = 0.$$

Notice that the entry in the last row and rightmost column is the value of the objective function for the initial basic feasible solution.

Tableau 2.2

	x_1	x_2	$\cdots$	x_n	x_{n+1}	x_{n+2}	$\cdots$	x_{n+m}	z	
x_{n+1}	a_{11}	a_{12}	$\cdots$	a_{1n}	1	0	$\cdots$	0	0	b_1
x_{n+2}	a_{21}	a_{22}	$\cdots$	a_{2n}	0	1	$\cdots$	0	0	b_2
$\vdots$	$\vdots$	$\vdots$		$\vdots$	$\vdots$	$\vdots$		$\vdots$	$\vdots$	$\vdots$
x_{n+m}	a_{m1}	a_{m2}	$\cdots$	a_{mn}	0	0	$\cdots$	1	0	b_m
	$-c_1$	$-c_2$	$\cdots$	$-c_n$	0	0	$\cdots$	0	1	0

In our example we have

$$z = 120 \cdot 0 + 100 \cdot 0 + 0 \cdot 8 + 0 \cdot 15 = 0.$$

At this point the given linear programming problem has been transformed to the initial tableau. This tableau displays the constraints and objective function along with an initial basic feasible solution and the corresponding value of the objective function for this basic feasible solution. We are now ready to describe the steps in the simplex method that

are used repeatedly to create a sequence of tableaux, terminating in a tableau that yields an optimal solution to the problem.

Checking an Optimality Criterion

We shall now turn to the development of a criterion that will determine whether the basic feasible solution represented by a tableau is, in fact, optimal. For our example we can increase the value of z from its value of 0 by increasing any one of the nonbasic variables having a positive coefficient from its current value of 0 to some positive value. For our example,

$$z = 120x + 100y + 0 \cdot u + 0 \cdot v,$$

so that z can be increased by increasing either x or y.

For an arbitrary tableau, if we write the objective function so that the coefficients of the basic variables are zero, we then have

$$z = \sum_{\text{nonbasic}} d_j x_j + \sum_{\text{basic}} 0 \cdot x_i, \tag{11}$$

where the d_j's are the negatives of the entries in the objective row of the tableau. We see that (11) has some terms with positive coefficients if and only if the objective row has negative entries under some of the columns labeled by nonbasic variables. Now the value of z can be increased by increasing the value of any nonbasic variable with a negative entry in the objective row from its current value of 0. If this is done, then some basic variable must be set to zero since the number of basic variables is to remain unchanged. Setting this basic variable to zero will not change the value of the objective function since the coefficient of the basic variable was zero. We summarize this discussion by stating the following optimality criterion for testing whether a feasible solution shown in a tableau is an optimal solution.

Optimality Criterion. If the objective row of a tableau has zero entries in the columns labeled by basic variables and no negative entries in the columns labeled by nonbasic variables, then the solution represented by the tableau is optimal.

As soon as the optimality criterion has been met, we can stop our computations, for we have found an optimal solution.

Selecting the Entering Variable

Suppose now that the objective row of a tableau has negative entries in the labeled columns. Then the solution shown in the tableau is not optimal, and some adjustment of the values of the variables must be made.

The simplex method proceeds from a given extreme point (basic feasible solution) to an *adjacent* extreme point in such a way that the objective function increases in value. From the definition of adjacent extreme point, it is clear that we reach such a point by increasing a single variable from zero to a positive value and decreasing a variable with a positive value to zero. The largest increase in z per unit increase in a variable occurs for the most negative entry in the objective row. We shall see below that, if the feasible set is bounded, there is a limit on the amount by which we can increase a variable. Because of this limit, it may turn out that a larger increase in z may be achieved by *not* increasing the variable with the most negative entry in the objective row. However, this rule is most commonly followed because of its computational simplicity. Some computer implementations of the simplex algorithm provide other strategies for choosing the variable to be increased, including one as simple as choosing the first negative entry. Another compares increases in the objective function for several likely candidates for the entering variable. In Tableau 2.1, the most negative entry, -120, in the objective row occurs under the x column, so that x is chosen to be the variable to be increased from zero to a positive value. The variable to be increased is called the **entering variable**, since in the next iteration it will become a basic variable; that is, it will *enter* the set of basic variables. If there are several possible entering variables, choose one. (This situation will occur when the most negative entry in the objective row occurs in more than one column.) Now an increase in one variable must be accompanied by a decrease in some of the other variables to maintain a solution to $\mathbf{Ax} = \mathbf{b}$.

Choosing the Departing Variable

Solving (8) for the basic variables u and v, we have

$$\begin{aligned} u &= 8 - 2x - 2y \\ v &= 15 - 5x - 3y. \end{aligned}$$

We increase only x and keep y at zero. We have

$$\left.\begin{aligned} u &= 8 - 2x \\ v &= 15 - 5x \end{aligned}\right\}, \tag{12}$$

which shows that as x increases both u and v decrease. By how much can we increase x? It can be increased until either u or v becomes negative.

That is, from (9) and (12) we have

$$0 \le u = 8 - 2x$$
$$0 \le v = 15 - 5x.$$

Solving these inequalities for x, we find

$$2x \le 8 \quad \text{or} \quad x \le 8/2 = 4$$

and

$$5x \le 15 \quad \text{or} \quad x \le 15/5 = 3.$$

We see that we cannot increase x by more than the smaller of the two ratios $8/2$ and $15/5$. Letting $x = 3$, we obtain a new feasible solution,

$$x = 3, \quad y = 0, \quad u = 2, \quad v = 0.$$

In fact, this is a basic feasible solution, and it was constructed to be adjacent to the previous basic feasible solution, since only one variable changed from basic to nonbasic. The new basic variables are x and u; the nonbasic variables are y and v. The objective function now has the value

$$z = 120 \cdot 3 + 100 \cdot 0 + 0 \cdot 2 + 0 \cdot 0 = 360,$$

which is a considerable improvement over the previous value of zero.

The new basic feasible solution yields the extreme point $(3, 0)$ in Figure 1.14, and it is adjacent to $(0, 0)$. In the new basic feasible solution to our example, we have the variable $v = 0$. It is no longer a basic variable because it is zero, and it is called a **departing variable** since it has *departed* from the set of basic variables. The column of the entering variable is called the **pivotal column**; the row that is labeled with the departing variable is called the **pivotal row**.

We now examine more carefully the selection of the departing variable. Recall that the ratios of the rightmost column entries to the corresponding entries in the pivotal column were determined by how much we could increase the entering variable (x in our example). These ratios are called **θ-ratios**. The smallest nonnegative θ-ratio is the largest possible value for the entering variable. The basic variable labeling the row where the smallest nonnegative θ-ratio occurs is the departing variable, and the row is the pivotal row. In our example,

$$\min\{8/2, 15/5\} = 3,$$

and the second row in Tableau 2.1 is the pivotal row.

If the smallest nonnegative θ-ratio is not chosen, then the next basic solution is not feasible. Suppose we had chosen u as the departing variable by choosing the θ-ratio as 4. Then $x = 4$, and from (12) we have

$$u = 8 - 2 \cdot 4 = 0$$
$$v = 15 - 5 \cdot 4 = -5,$$

and the next basic solution is

$$x = 4, \quad y = 0, \quad u = 0, \quad v = -5,$$

which is not feasible.

In the general case, we have assumed that the rightmost column will contain only nonnegative entries. However, the entries in the pivotal column may be positive, negative, or zero. Positive entries lead to nonnegative θ-ratios, which are fine. Negative entries lead to nonpositive θ-ratios. In this case, there is no restriction imposed on how far the entering variable can be increased. For example, suppose the pivotal column in our example were

$$\begin{bmatrix} -2 \\ 5 \end{bmatrix} \quad \text{instead of} \quad \begin{bmatrix} 2 \\ 5 \end{bmatrix}.$$

Then we would have, instead of (12),

$$\begin{aligned} u &= 8 + 2x \\ v &= 15 - 5x. \end{aligned}$$

Since u must be nonnegative, we find that

$$8 + 2x \geq 0 \quad \text{or} \quad x \geq -4,$$

which puts no restriction on how far we can increase x. Thus, in calculating θ-ratios we can ignore any negative entries in the pivotal column.

If an entry in the pivotal column is zero, the corresponding θ-ratio is undefined. However, checking the equations corresponding to (12), but with one of the entries in the pivotal column equal to zero, will show that no restriction is placed on the size of x by the zero entry. Consequently, in forming the θ-ratios we use only the positive entries in the pivotal column that are above the objective row.

If all the entries in the pivotal column above the objective row are either zero or negative, then the entering variable can be made as large as we wish. Hence, the given problem has no finite optimal solution, and we can stop.

Forming a New Tableau

Having determined the entering and departing variables, we must obtain a new tableau showing the new basic variables and the new basic feasible solution. We illustrate the procedure with our continuing example. Solving the second equation of (8) (it corresponds to the departing variable) for x, the entering variable, we have

$$x = 3 - \tfrac{3}{5}y - \tfrac{1}{5}v. \tag{13}$$

Substituting (13) into the first equation of (8), we get

$$2(3 - \tfrac{3}{5}y - \tfrac{1}{5}v) + 2y + u = 8$$

or

$$\tfrac{4}{5}y + u - \tfrac{2}{5}v = 2. \tag{14}$$

We also rewrite (13) as

$$x + \tfrac{3}{5}y + \tfrac{1}{5}v = 3. \tag{15}$$

Substituting (13) into (7), we have

$$(-120)(3 - \tfrac{3}{5}y - \tfrac{1}{5}v) - 100y + z = 0$$

or

$$-28y + 24v + z = 360. \tag{16}$$

Since in the new basic feasible solution we have $y = v = 0$, the value of z for this solution is 360. This value appears as the entry in the last row and rightmost column. Equations (14), (15), and (16) yield the new tableau (Tableau 2.3).

Tableau 2.3

	x	y	u	v	z	
u	0	$\frac{4}{5}$	1	$-\frac{2}{5}$	0	2
x	1	$\frac{3}{5}$	0	$\frac{1}{5}$	0	3
	0	−28	0	24	1	360

Observe that the basic variables in Tableau 2.3 are x and u. By comparing Tableaus 2.1 and 2.3, we see that the steps that were used to obtain Tableau 2.3 from Tableau 2.1 are as follows.

Step a. Locate and circle the entry at the intersection of the pivotal row and pivotal column. This entry is called the **pivot**. Mark the pivotal column by placing an arrow ↓ above the entering variable, and mark the pivotal row by placing an arrow ← to the left of the departing variable.

Step b. If the pivot is k, multiply the pivotal row by $1/k$, making the entry in the pivot position in the new tableau equal to 1.

Step c. Add suitable multiples of the new pivotal row to all other rows (including the objective row), so that all other elements in the pivotal column become zero.

Step d. In the new tableau, replace the label on the pivotal row by the entering variable.

These four steps constitute a process called **pivoting**. Steps b and c use elementary row operations (see Section 0.2) and form one iteration of the procedure used to transform a given matrix to reduced row echelon form.

We now repeat Tableau 2.1 with the arrows placed next to the entering and departing variables and with the pivot circled (Tableau 2.1a).

Tableau 2.1a

		x	y	u	v	z	
	u	2	2	1	0	0	8
←	v	⑤	3	0	1	0	15
		−120	−100	0	0	1	0

Tableau 2.3 was obtained from Tableau 2.1 by pivoting. We now repeat the process with Tableau 2.3. Since the most negative entry in the objective row of Tableau 2.3, -28, occurs in the second column, y is the entering variable of this tableau and the second column is the pivotal column. To find the departing variable we form the θ-ratios, that is, the ratios of the entries in the rightmost column (except for the objective row) to the corresponding entries of the pivotal column for those entries in the pivotal column that are positive. The θ-ratios are

$$\frac{2}{\frac{4}{5}} = \frac{5}{2} \quad \text{and} \quad \frac{3}{\frac{3}{5}} = 5.$$

The minimum of these is $\frac{5}{2}$, which occurs for the first row. Therefore, the pivotal row is the first row, the pivot is $\frac{4}{5}$, and the departing variable is u. We now show Tableau 2.3 with the pivot, entering, and departing variables marked (Tableau 2.3a).

Tableau 2.3a

		x	y	u	v	z	
←	u	0	⑷/5	1	$-\frac{2}{5}$	0	2
	x	1	$\frac{3}{5}$	0	$\frac{1}{5}$	0	3
		0	−28	0	24	1	360

We obtain Tableau 2.4 from Tableau 2.3 by pivoting. Since the objective row in Tableau 2.4 has no negative entries, we are finished, by the optimality criterion. That is, the indicated solution,

$$x = \tfrac{3}{2}, \quad y = \tfrac{5}{2}, \quad u = 0, \quad v = 0,$$

Tableau 2.4

	x	y	u	v	z	
y	0	1	$\frac{5}{4}$	$-\frac{1}{2}$	0	$\frac{5}{2}$
x	1	0	$-\frac{3}{4}$	$\frac{1}{2}$	0	$\frac{3}{2}$
	0	0	35	10	1	430

is optimal, and the maximum value of z is 430. Notice from Figure 1.14 that we moved from the extreme point $(0, 0)$ to the adjacent extreme point $(3, 0)$ and then to the adjacent extreme point $(\frac{3}{2}, \frac{5}{2})$. The value of the objective function started at 0, increased to 360, and then to 430, the entry in the last row and rightmost column.

Summary of the Simplex Method

We assume that the linear programming problem is in standard form and that $\mathbf{b} \geq \mathbf{0}$. In this case the initial basic feasible solution is

$$\mathbf{x} = \begin{bmatrix} \mathbf{0} \\ \mathbf{b} \end{bmatrix}.$$

In subsequent sections we will show how to extend the simplex method to other linear programming problems.

Step 1. Set up the initial tableau.

Step 2. Apply the optimality test: If the objective row has no negative entries in the labeled columns, then the indicated solution is optimal. Stop computation.

Step 3. Find the pivotal column by determining the column with the most negative entry in the objective row. If there are several possible pivotal columns, choose any one.

Step 4. Find the pivotal row. This is done by forming the **θ-ratios**—the ratios formed by dividing the entries of the rightmost column (except for the objective row) by the corresponding entries of the pivotal columns using only those entries in the pivotal column that are positive. The **pivotal row** is the row for which the minimum ratio occurs. If two or more θ-ratios are the same, choose one of the possible rows. If none of the entries in the pivotal column above the objective row is positive, the problem has no finite optimum. We stop our computation in this case.

Step 5. Obtain a new tableau by pivoting. Then return to Step 2.

In Figure 2.1 we give a flowchart and in Figure 2.2, a structure diagram for the simplex algorithm.

The reader can use the SMPX courseware described in Appendix C to experiment with different choices of pivot, observing how some choices

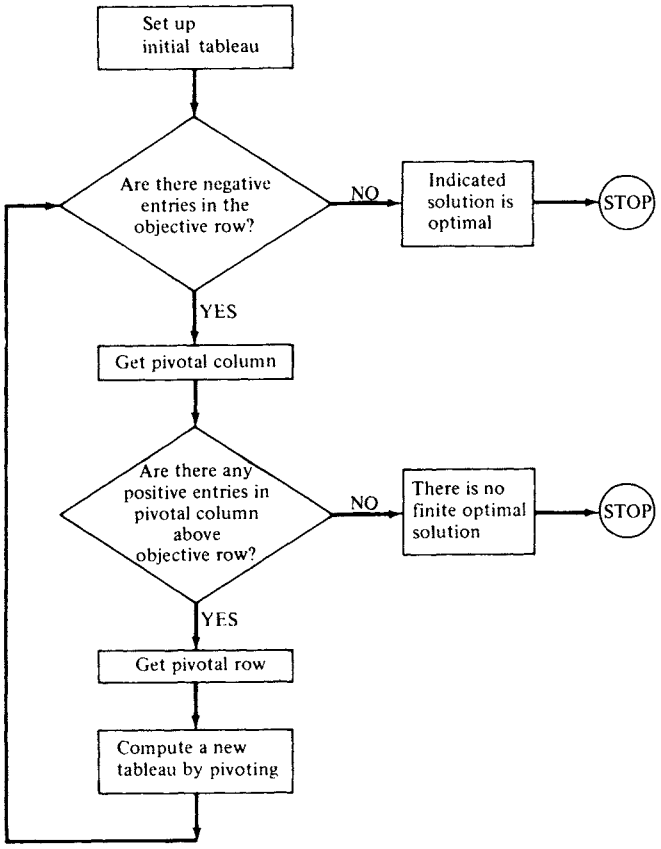


FIGURE 2.1 Flowchart for simplex algorithm (standard form, $\mathbf{b} \geq \mathbf{0}$).

lead to infeasible solutions. The courseware will also allow the user to step through the iterations of the simplex algorithm so that the intermediate tableaux can be examined.

The reader should note that the z column always appears in the form

z
0
0
⋮
0
1

in any simplex tableau. We included it initially to remind the reader that each row of a tableau including the objective row represents an equation

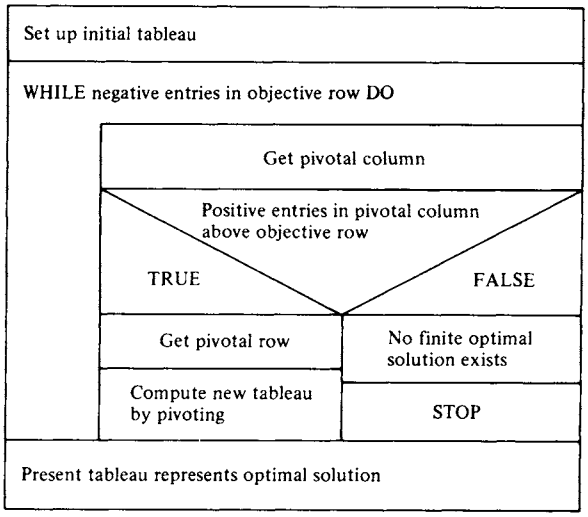


FIGURE 2.2 Structure diagram of simplex algorithm (standard form, $\mathbf{b} \geq \mathbf{0}$).

in the variables $x_1, x_2, \ldots, x_s, z$. From this point on we will not include the z column in tableaux. The student should remember to read the objective row of a tableau as an equation that involves z with coefficient $+1$.

EXAMPLE 2. We solve the following linear programming problem in standard form by using the simplex method:

$$\begin{aligned} &\text{Maximize} \quad z = 8x_1 + 9x_2 + 5x_3 \\ &\text{subject to} \\ &\qquad x_1 + \ x_2 + 2x_3 \leq 2 \\ &\qquad 2x_1 + 3x_2 + 4x_3 \leq 3 \\ &\qquad 6x_1 + 6x_2 + 2x_3 \leq 8 \\ &\qquad x_1 \geq 0, \quad x_2 \geq 0, \quad x_3 \geq 0. \end{aligned}$$

We first convert the problem to canonical form by adding slack variables, obtaining:

$$\begin{aligned} &\text{Maximize} \quad z = 8x_1 + 9x_2 + 5x_3 \\ &\text{subject to} \\ &x_1 + \ x_2 + 2x_3 + x_4 \qquad\quad = 2 \\ &2x_1 + 3x_2 + 4x_3 \qquad + x_5 \qquad = 3 \\ &6x_1 + 6x_2 + 2x_3 \qquad\quad + x_6 = 8 \\ &\qquad x_j \geq 0, \quad j = 1, 2, \ldots, 6. \end{aligned}$$

Tableau 2.5

		x_1	x_2 ↓	x_3	x_4	x_5	x_6	
	x_4	1	1	2	1	0	0	2
←	x_5	2	③	4	0	1	0	3
	x_6	6	6	2	0	0	1	8
		−8	−9	−5	0	0	0	0

Tableau 2.6

		x_1	x_2 ↓	x_3	x_4	x_5	x_6	
	x_4	$\frac{1}{3}$	0	$\frac{2}{3}$	1	$-\frac{1}{3}$	0	1
	x_2	$\frac{2}{3}$	1	$\frac{4}{3}$	0	$\frac{1}{3}$	0	1
←	x_6	②	0	−6	0	−2	1	2
		−2	0	7	0	3	0	9

Tableau 2.7

	x_1	x_2	x_3	x_4	x_5	x_6	
x_4	0	0	$\frac{5}{3}$	1	0	$-\frac{1}{6}$	$\frac{2}{3}$
x_2	0	1	$\frac{10}{3}$	0	1	$-\frac{1}{3}$	$\frac{1}{3}$
x_1	1	0	−3	0	−1	$\frac{1}{2}$	1
	0	0	1	0	1	1	11

The initial tableau is Tableau 2.5; the succeeding tableaux are Tableaux 2.6 and 2.7.

Hence, an optimal solution to the standard form of the problem is

$$x_1 = 1, \quad x_2 = \tfrac{1}{3}, \quad x_3 = 0.$$

The values of the slack variables are

$$x_4 = \tfrac{2}{3}, \quad x_5 = 0, \quad x_6 = 0.$$

The optimal value of z is 11. △

EXAMPLE 3. Consider the linear programming problem

$$\text{Maximize} \quad z = 2x_1 + 3x_2 + x_3 + x_4$$

subject to

$$\begin{aligned} x_1 - x_2 - x_3 &\le 2 \\ -2x_1 + 5x_2 - 3x_3 - 3x_4 &\le 10 \\ 2x_1 - 5x_2 + 3x_4 &\le 5 \\ x_j \ge 0, \quad j &= 1,2,3,4. \end{aligned}$$

To solve this problem by the simplex method, we first convert the problem to canonical form by adding slack variables obtaining

$$\begin{aligned} &\text{Maximize} \quad z = 2x_1 + 3x_2 + x_3 + x_4 \\ &\text{subject to} \\ &x_1 - x_2 - x_3 + x_5 = 2 \\ &-2x_1 + 5x_2 - 3x_3 - 3x_4 + x_6 = 10 \\ &2x_1 - 5x_2 + 3x_4 + x_7 = 5 \\ &x_j \geq 0, \quad j = 1, 2, \ldots, 7. \end{aligned}$$

The initial tableau is Tableau 2.8; the following tableaux are Tableaux 2.9 and 2.10.

In Tableau 2.10, the most negative entry in the objective row is $-\frac{34}{3}$, so the departing variable is x_3. However, none of the entries in the pivotal column (the third column) is positive, so we conclude that the given problem has no finite optimal solution. △

Tableau 2.8

		x_1	x_2	x_3	x_4	x_5	x_6	x_7	
	x_5	1	−1	−1	0	1	0	0	2
←	x_6	−2	⑤	−3	−3	0	1	0	10
	x_7	2	−5	0	3	0	0	1	5
		−2	−3	−1	−1	0	0	0	0

Tableau 2.9

		x_1	x_2	x_3	x_4	x_5	x_6	x_7	
	x_5	$\frac{3}{5}$	0	$-\frac{8}{5}$	$-\frac{3}{5}$	1	$\frac{1}{5}$	0	4
←	x_2	$-\frac{2}{5}$	1	$-\frac{3}{5}$	$-\frac{3}{5}$	0	$\frac{1}{5}$	0	2
	x_7	0	0	−3	0	0	1	1	15
		$-\frac{16}{5}$	0	$-\frac{14}{5}$	$-\frac{14}{5}$	0	$\frac{3}{5}$	0	6

Tableau 2.10

	x_1	x_2	x_3	x_4	x_5	x_6	x_7	
x_1	1	0	$-\frac{8}{3}$	−1	$\frac{5}{3}$	$\frac{1}{3}$	0	$\frac{20}{3}$
x_2	0	1	$-\frac{5}{3}$	−1	$\frac{2}{3}$	$\frac{1}{3}$	0	$\frac{14}{3}$
x_7	0	0	−3	0	0	1	1	15
	0	0	$-\frac{34}{3}$	−6	$\frac{16}{3}$	$\frac{5}{3}$	0	$\frac{82}{3}$

2.1 EXERCISES

In Exercises 1 and 2, set up the initial simplex tableau.

1. Maximize $z = 2x + 5y$
subject to

$$3x + 5y \leq 8$$
$$2x + 7y \leq 12$$
$$x \geq 0, \quad y \geq 0.$$

2. Maximize $z = x_1 + 3x_2 + 5x_3$
subject to

$$2x_1 - 5x_2 + x_3 \leq 3$$
$$x_1 + 4x_2 \leq 5$$
$$x_1 \geq 0, \quad x_2 \geq 0, \quad x_3 \geq 0.$$

3. Consider the following simplex tableau.

	x_1	x_2	x_3	x_4	x_5	x_6	x_7	
x_4	0	0	2	1	$\frac{5}{2}$	0	0	$\frac{6}{7}$
x_1	1	0	5	0	-3	0	-2	$\frac{2}{7}$
x_6	0	0	3	0	4	1	-4	$\frac{5}{7}$
x_2	0	1	0	0	$\frac{3}{2}$	0	0	$\frac{1}{7}$

Determine the departing variable if the entering variable is (a) x_5; (b) x_3; (c) x_7.

In Exercises 4–7 use one iteration of the simplex algorithm to obtain the next tableau from the given tableau.

4.

	x_1	x_2	x_3	x_4	
x_4	$\frac{3}{2}$	0	$\frac{5}{3}$	1	6
x_2	$\frac{2}{3}$	1	2	0	8
	-4	0	-2	0	12

5.

	x_1	x_2	x_3	x_4	
x_1	1	2	0	1	3
x_3	0	$\frac{1}{2}$	1	-1	$\frac{3}{2}$
	0	-4	0	-4	$\frac{11}{2}$

6.

	x_1	x_2	x_3	x_4	x_5	
x_3	$\frac{2}{3}$	0	1	$\frac{3}{5}$	0	$\frac{3}{2}$
x_2	$\frac{3}{2}$	1	0	1	0	$\frac{5}{2}$
x_5	5	0	0	$\frac{2}{9}$	1	$\frac{2}{3}$
	4	0	0	-5	0	$\frac{7}{3}$

7.

	x_1	x_2	x_3	x_4	
x_2	1	1	5	0	4
x_4	-1	0	2	1	6
	-3	0	-2	0	7

8. (a) The following tableau arose in the course of using the simplex algorithm to solve a linear programming problem. What basic feasible solution does this tableau represent?

x_1	x_2	x_3	x_4	x_5	x_6	x_7	
0	$\frac{4}{3}$	$\frac{2}{3}$	0	1	0	$-\frac{1}{3}$	4
0	$\frac{1}{3}$	$\frac{2}{3}$	1	0	1	$-\frac{1}{3}$	10
1	$\frac{1}{3}$	$\frac{1}{6}$	$\frac{1}{2}$	0	0	$\frac{1}{6}$	4
0	$-\frac{5}{3}$	$-\frac{4}{3}$	-1	0	0	$\frac{5}{3}$	12

(b) Perform one operation of the simplex algorithm on the tableau. What basic feasible solution does this new tableau represent?

9. Consider the following tableau, which arose in solving a linear programming problem by the simplex method.

x_1	x_2	x_3	u	v	w	
1	5	2	0	0	3	20
0	2	4	1	0	−4	6
0	2	−1	0	1	3	12
0	−5	−3	0	0	3	12

(a) Identify the basic feasible solution *and* basic variables in this tableau.
(b) Compute the next tableau using the simplex method.
(c) Identify the basic feasible solution *and* basic variables in the tableau in (b).

In Exercises 10–23 solve the indicated linear programming problem using the simplex method.

10. Example 4, Section 1.1.
11. Example 7a, Section 1.1.
12. Example 7b, Section 1.1.
13. Example 10, Section 1.1.
14. Exercise 4, Section 1.1.
15. Exercise 5, Section 1.1.
16. Exercise 7, Section 1.1.
17. Exercise 9, Section 1.1.
18. Exercise 2, Section 1.5.
19. Maximize $z = 2x_1 + 3x_2 - x_3$
subject to
$$\begin{aligned} x_1 + 2x_2 - x_3 &\le 6 \\ x_1 - 3x_2 - 3x_3 &\le 10 \\ x_j \ge 0, \quad j &= 1,2,3. \end{aligned}$$
20. Maximize $z = x_1 + 2x_2 + x_3 + x_4$
subject to
$$\begin{aligned} 2x_1 + x_2 + 3x_3 + x_4 &\le 8 \\ 2x_1 + 3x_2 + 4x_4 &\le 12 \\ 3x_1 + x_2 + 2x_3 &\le 18 \\ x_j \ge 0, \quad j &= 1,2,3,4. \end{aligned}$$
21. Maximize $z = 5x_1 + 2x_2 + x_3 + x_4$
subject to
$$\begin{aligned} 2x_1 + x_2 + x_3 + 2x_4 &\le 6 \\ 3x_1 + x_3 &\le 15 \\ 5x_1 + 4x_2 + x_4 &\le 24 \\ x_j \ge 0, \quad j &= 1,2,3,4. \end{aligned}$$

22. Maximize $z = -x_1 + 3x_2 + x_3$
subject to

$$\begin{aligned} -x_1 + 2x_2 - 7x_3 &\leq 6 \\ x_1 + x_2 - 3x_3 &\leq 15 \\ x_j &\geq 0, \quad j = 1,2,3. \end{aligned}$$

23. Maximize $z = 3x_1 + 3x_2 - x_3 + x_4$
subject to

$$\begin{aligned} 2x_1 - x_2 - x_3 + x_4 &\leq 2 \\ x_1 - x_2 + x_3 - x_4 &\leq 5 \\ 3x_1 + x_2 + 5x_4 &\leq 12 \\ x_j &\geq 0, \quad j = 1,2,3,4. \end{aligned}$$

24. Suppose a linear programming problem has a constraint of the form

$$3x_1 + 2x_2 + 5x_3 - 2x_4 \geq 12.$$

Why can we not solve this problem using the simplex method as described up to this point? (In Section 2.3 we develop techniques for handling this situation.)

2.2 DEGENERACY AND CYCLING (OPTIONAL)

In choosing the departing variable, we computed the minimum θ-ratio. If the minimum θ-ratio occurs, say, in the rth row of a tableau, we drop the variable that labels that row. Now suppose that there is a tie for minimum θ-ratio, so that several variables are candidates for departing variable. We choose one of the candidates by using an arbitrary rule such as dropping the variable with the smallest subscript. However, there are potential difficulties any time such an arbitrary choice must be made. We now examine these difficulties.

Suppose that the θ-ratios for the rth and sth rows of a tableau are the same and their value is the minimum value of all the θ-ratios. These two rows of the tableau are shown in Tableau 2.11 with the label on the rth row marked as the departing variable. The θ-ratios of these two rows are

$$b_r/a_{rj} = b_s/a_{sj}.$$

Tableau 2.11

		x_1	x_2	$\cdots$	x_j	$\cdots$	x_{n+m}	
	$\vdots$	$\vdots$	$\vdots$		$\vdots$		$\vdots$	$\vdots$
$\leftarrow$	x_{i_r}	a_{r_1}	a_{r_2}	$\cdots$	(a_{rj})	$\cdots$	$a_{r,n+m}$	b_r
	$\vdots$	$\vdots$	$\vdots$		$\vdots$		$\vdots$	$\vdots$
	x_{i_s}	a_{s1}	a_{s2}	$\cdots$	a_{sj}	$\cdots$	$a_{s,n+m}$	b_s
	$\vdots$	$\vdots$	$\vdots$		$\vdots$		$\vdots$	$\vdots$

$\downarrow$

Tableau 2.12

	x_1	x_2	$\cdots$	x_j	$\cdots$	x_{n+m}	
$\vdots$	$\vdots$	$\vdots$		$\vdots$		$\vdots$	$\vdots$
x_j	a_{r1}/a_{rj}	a_{r2}/a_{rj}	$\cdots$	1	$\cdots$	$a_{r,n+m}/a_{rj}$	b_r/a_{rj}
$\vdots$	$\vdots$	$\vdots$		$\vdots$		$\vdots$	$\vdots$
x_{i_s}	*	*	$\cdots$	0	$\cdots$	*	$b_s - a_{sj} \cdot b_r/a_{rj}$
$\vdots$	$\vdots$	$\vdots$		$\vdots$		$\vdots$	$\vdots$

When we pivot in Tableau 2.11, we obtain Tableau 2.12, where * indicates an entry whose value we are not concerned about. Setting the nonbasic variables in Tableau 2.12 equal to zero, we find that

$$x_j = b_r/a_{rj}$$

and

$$x_{i_s} = b_s - a_{sj} \cdot b_r/a_{rj} = a_{sj}(b_s/a_{sj} - b_r/a_{rj}) = 0.$$

Consequently, the tie among the θ-ratios has produced a basic variable whose value is 0.

DEFINITION. A basic feasible solution in which some basic variables are zero is called **degenerate.**

EXAMPLE 1 (DEGENERACY). Consider the linear programming problem in standard form

$$\text{Maximize} \quad z = 5x_1 + 3x_3$$

subject to

$$\begin{aligned} x_1 - x_2 &\le 2 \\ 2x_1 + x_2 &\le 4 \\ -3x_1 + 2x_2 &\le 6 \\ x_1 \ge 0, \quad x_2 &\ge 0. \end{aligned}$$

The region of all feasible solutions is shown in Figure 2.3. The extreme points and corresponding values of the objective function are given in Table 2.1. The simplex method leads to the following tableaux. In Tableau 2.13 we have two candidates for the departing variable: x_3 and x_4 since the θ-ratios are equal. Choosing x_3 gives Tableaux 2.13, 2.14, 2.15, and 2.16. Choosing x_4 gives Tableaux 2.13a, 21.4a, and 2.15a. Note that Tableaux 2.15a and 2.16 are the same except for the order of the constraint rows.

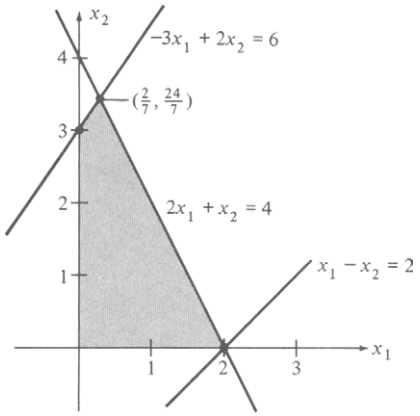


FIGURE 2.3

TABLE 2.1

Extreme point	*Value of* $z = 5x_1 + 3x_2$
(0, 0)	0
(2, 0)	10
(0, 3)	9
$(\frac{2}{7}, \frac{24}{7})$	$\frac{82}{7}$

Tableau 2.13

		x_1	x_2	x_3	x_4	x_5	
←	x_3	①	−1	1	0	0	2
	x_4	2	1	0	1	0	4
	x_5	−3	2	0	0	1	6
		−5	−3	0	0	0	0

Tableau 2.14

		x_1	x_2	x_3	x_4	x_5	
	x_1	1	−1	1	0	0	2
←	x_4	0	③	−2	1	0	0
	x_5	0	−1	3	0	1	12
		0	−8	5	0	0	10

Tableau 2.15

			↓			
	x_1	x_2	x_3	x_4	x_5	
x_1	1	0	$\frac{1}{3}$	$\frac{1}{3}$	0	2
x_2	0	1	$-\frac{2}{3}$	$\frac{1}{3}$	0	0
← x_5	0	0	(⑦/3)	$\frac{1}{3}$	1	12
	0	0	$-\frac{1}{3}$	$\frac{8}{3}$	0	10

Tableau 2.16

	x_1	x_2	x_3	x_4	x_5	
x_1	1	0	0	$\frac{2}{7}$	$-\frac{1}{7}$	$\frac{2}{7}$
x_2	0	1	0	$\frac{3}{7}$	$\frac{2}{7}$	$\frac{24}{7}$
x_3	0	0	1	$\frac{1}{7}$	$\frac{3}{7}$	$\frac{36}{7}$
	0	0	0	$\frac{19}{7}$	$\frac{1}{7}$	$\frac{82}{7}$

Tableau 2.13a

		↓				
	x_1	x_2	x_3	x_4	x_5	
x_3	1	−1	1	0	0	2
← x_4	(②)	1	0	1	0	4
x_5	−3	2	0	0	1	6
	−5	−3	0	0	0	0

Tableau 2.14a

			↓			
	x_1	x_2	x_3	x_4	x_5	
x_3	0	$-\frac{3}{2}$	1	$-\frac{1}{2}$	0	0
x_1	1	$\frac{1}{2}$	0	$\frac{1}{2}$	0	2
← x_5	0	(⑦/2)	0	$\frac{3}{2}$	1	12
	0	$-\frac{1}{2}$	0	$\frac{5}{2}$	0	10

Tableau 2.15a

	x_1	x_2	x_3	x_4	x_5	
x_3	0	0	1	$\frac{1}{7}$	$\frac{3}{7}$	$\frac{36}{7}$
x_1	1	0	0	$\frac{2}{7}$	$-\frac{1}{7}$	$\frac{2}{7}$
x_2	0	1	0	$\frac{3}{7}$	$\frac{2}{7}$	$\frac{24}{7}$
	0	0	0	$\frac{19}{7}$	$\frac{1}{7}$	$\frac{82}{7}$

The optimal solution is

$$x_1 = \tfrac{2}{7}, \qquad x_2 = \tfrac{24}{7},$$

with the optimal value of the objective function being

$$z = \tfrac{82}{7}.$$

The slack variables have values

$$x_3 = \tfrac{36}{7}, \quad x_4 = 0, \quad x_5 = 0.$$

What is happening geometrically? We start with the initial basic feasible solution as the origin $(0, 0)$, where $z = 0$. If we choose to replace x_3 with x_1, we move to the adjacent extreme point $(2, 0)$, where $z = 10$ (Tableau 2.14). Now we replace x_4 with x_2 and remain at $(2, 0)$ (Tableau 2.15). Finally we replace x_5 with x_3 and move to $(\frac{2}{7}, \frac{24}{7})$, where $z = \frac{82}{7}$. This is our optimal solution (Tableau 2.16).

Alternatively, because the θ-ratios are equal we could replace x_4 with x_1. The pivot step with this choice again moves us from $(0, 0)$ to $(2, 0)$, where $z = 10$ (Tableau 2.14a). However, at the next stage, x_3, which has value 0 and is the degenerate variable, is not a departing variable. Instead, x_5 is the departing variable, and we move immediately to the optimal solution (Tableau 2.15a) at $(\frac{2}{7}, \frac{24}{7})$. △

In general, in the case of degeneracy, an extreme point is the intersection of too many hyperplanes. For example, degeneracy occurs in R^2 when three or more lines intersect at a point, degeneracy occurs in R^3 when four or more planes intersect at a point, and so on.

Cycling

If no degenerate solution occurs in the course of the simplex method, then the value of z increases as we go from one basic feasible solution to an adjacent basic feasible solution. Since the number of basic feasible solutions is finite, the simplex method eventually stops. However, if we have a degenerate basic feasible solution and if a basic variable whose value is zero departs, then the value of z does not change. To see this, observe that z increases by a multiple of the value in the rightmost column of the pivotal row. But this value is zero, so that z does not increase. Therefore, after several steps of the simplex method we may return to a basic feasible solution that we already have encountered. In this case the simplex method is said to be **cycling** and will never terminate by finding an optimal solution or concluding that no bounded optimal solution exists. Cycling can only occur in the presence of degeneracy, but many linear programming problems that are degenerate do not cycle (see Example 1).

Examples of problems that cycle are difficult to construct and rarely occur in practice. However, Kotiah and Steinberg (see Further Reading)

have discovered a linear programming problem arising in the solution of a practical queuing model that does cycle. Also, Beale (see Further Reading) has constructed the following example of a smaller problem that cycles after a few steps.

EXAMPLE 2 (CYCLING). Consider the following linear programming problem in canonical form.

$$\begin{aligned}
&\text{Maximize} \quad z = 10x_1 - 57x_2 - 9x_3 - 24x_4 \\
&\text{subject to} \\
&\tfrac{1}{2}x_1 - \tfrac{11}{2}x_2 - \tfrac{5}{2}x_3 + 9x_4 + x_5 = 0 \\
&\tfrac{1}{2}x_1 - \tfrac{3}{2}x_2 - \tfrac{1}{2}x_3 + x_4 + x_6 = 0 \\
&x_1 + x_7 = 1 \\
&x_j \geq 0, \quad j = 1, \ldots, 7.
\end{aligned}$$

Using the simplex method we obtain the following sequence of tableaux.

Tableau 2.17

		x_1	x_2	x_3	x_4	x_5	x_6	x_7	
←	x_5	(1/2)	$-\frac{11}{2}$	$-\frac{5}{2}$	9	1	0	0	0
	x_6	$\frac{1}{2}$	$-\frac{3}{2}$	$-\frac{1}{2}$	1	0	1	0	0
	x_7	1	0	0	0	0	0	1	1
		−10	57	9	24	0	0	0	0

Tableau 2.18

		x_1	x_2	x_3	x_4	x_5	x_6	x_7	
	x_1	1	−11	−5	18	2	0	0	0
←	x_6	0	(4)	2	−8	−1	1	0	0
	x_7	0	11	5	−18	−2	0	1	1
		0	−53	−41	204	20	0	0	0

Tableau 2.19

		x_1	x_2	x_3	x_4	x_5	x_6	x_7	
←	x_1	1	0	(1/2)	−4	$-\frac{3}{4}$	$\frac{11}{4}$	0	0
	x_2	0	1	$\frac{1}{2}$	−2	$-\frac{1}{4}$	$\frac{1}{4}$	0	0
	x_7	0	0	$-\frac{1}{2}$	4	$\frac{3}{4}$	$-\frac{11}{4}$	1	1
		0	0	$-\frac{29}{2}$	98	$\frac{27}{4}$	$\frac{53}{4}$	0	0

Tableau 2.20

	x_1	x_2	x_3	x_4 ↓	x_5	x_6	x_7	
x_3	2	0	1	−8	$-\frac{3}{2}$	$\frac{11}{2}$	0	0
← x_2	−1	1	0	②	$\frac{1}{2}$	$-\frac{5}{2}$	0	0
x_7	1	0	0	0	0	0	1	1
	29	0	0	−18	−15	93	0	0

Tableau 2.21

	x_1	x_2	x_3	x_4	x_5 ↓	x_6	x_7	
← x_3	−2	4	1	0	($\frac{1}{2}$)	$-\frac{9}{2}$	0	0
x_4	$-\frac{1}{2}$	$\frac{1}{2}$	0	1	$\frac{1}{4}$	$-\frac{5}{4}$	0	0
x_7	1	0	0	0	0	0	1	1
	20	9	0	0	$-\frac{21}{2}$	$\frac{141}{2}$	0	0

Tableau 2.22

	x_1	x_2	x_3	x_4	x_5	x_6 ↓	x_7	
x_5	−4	8	2	0	1	−9	0	0
← x_4	$\frac{1}{2}$	$-\frac{3}{2}$	$-\frac{1}{2}$	1	0	①	0	0
x_7	1	0	0	0	0	0	1	1
	−22	93	21	0	0	−24	0	0

Tableau 2.23

	x_1	x_2	x_3	x_4	x_5	x_6	x_7	
x_5	$\frac{1}{2}$	$-\frac{11}{2}$	$-\frac{5}{2}$	9	1	0	0	0
x_6	$\frac{1}{2}$	$-\frac{3}{2}$	$-\frac{1}{2}$	1	0	1	0	0
x_7	1	0	0	0	0	0	1	1
	−10	57	9	24	0	0	0	0

Observe that Tableau 2.23 is identical to Tableau 2.17, and, thus, the simplex method has cycled. △

Computer programs designed for large linear programming problems provide several options for dealing with degeneracy and cycling. One option is to ignore degeneracy and to assume that cycling will not occur. Another option is to use Bland's Rule for choosing entering and departing variables to avoid cycling. This rule modifies Step 3 and 4 of the Simplex Method.

Bland's Rule

1. **Selecting the pivotal column.** Choose the column with the smallest subscript from among those columns with negative entries in the objective row instead of choosing the column with the most negative entry in the objective row.

2. **Selecting the pivotal row.** If two or more rows have equal θ-ratios, choose the row labeled by the basic variable with the smallest subscript, instead of making an arbitrary choice.

Bland showed that if these rules are used, then, in the event of degeneracy, cycling will not occur and the simplex method will terminate.

EXAMPLE 3. Referring to the tableaux from Example 2, note that Bland's rule affects only the choice of entering variable in Tableau 2.22. Applying the rule and rewriting Tableau 2.22 with the new choice of entering and departing variables, we obtain Tableau 2.23a.

Tableau 2.22

		x_1 ↓	x_2	x_3	x_4	x_5	x_6	x_7	
	x_5	-4	8	2	0	1	-9	0	0
←	x_4	⓵/2	$-\frac{3}{2}$	$-\frac{1}{2}$	1	0	1	0	0
	x_7	1	0	0	0	0	0	1	1
		-22	93	21	0	0	-24	0	0

Tableau 2.23a

		x_1	x_2	x_3 ↓	x_4	x_5	x_6	x_7	
	x_5	0	-4	-2	8	1	-1	0	0
	x_1	1	-3	-1	2	0	2	0	0
←	x_7	0	3	⓵	-2	0	-2	1	1
		0	27	-1	44	0	20	0	0

We perform the pivot step to obtain Tableau 2.24a, which represents an optimal solution. The cycling has been broken.

Tableau 2.24a

	x_1	x_2	x_3	x_4	x_5	x_6	x_7	
x_5	0	2	0	4	1	-5	2	2
x_1	1	0	0	0	0	0	1	1
x_3	0	3	1	-2	0	-2	1	1
	0	30	0	42	0	18	1	1

△

2.2 EXERCISES

In Exercises 1–6 solve the indicated linear programming problem, noting where degeneracies occur. Sketch the set of feasible solutions, indicating the order in which the extreme points are examined by the simplex algorithm.

1. Maximize $z = 6x_1 + 5x_2$
subject to
$$3x_1 - 2x_2 \leq 0$$
$$3x_1 + 2x_2 \leq 15$$
$$x_1 \geq 0, \quad x_2 \geq 0.$$

2. Maximize $z = 5x_1 + 4x_2$
subject to
$$x_1 + 2x_2 \leq 8$$
$$x_1 - 2x_2 \leq 4$$
$$3x_1 + 2x_2 \leq 12$$
$$x_1 \geq 0, \quad x_2 \geq 0.$$

3. Maximize $z = 3x_1 + 2x_2 + 5x_3$
subject to
$$2x_1 - x_2 + 4x_3 \leq 12$$
$$4x_1 + 3x_2 + 6x_3 \leq 18$$
$$x_1 \geq 0, \quad x_2 \geq 0, \quad x_3 \geq 0.$$

4. Maximize $z = 5x_1 + 8x_2 + x_3$
subject to
$$x_1 + x_2 + x_3 \leq 7$$
$$2x_1 + 3x_2 + 3x_3 \leq 12$$
$$3x_1 + 6x_2 + 5x_3 \leq 24$$
$$x_1 \geq 0, \quad x_2 \geq 0, \quad x_3 \geq 0.$$

5. Maximize $z = 6x_1 + 5x_2$
subject to
$$4x_1 + 3x_2 \leq 19$$
$$x_1 - x_2 \leq 3$$
$$x_1 - 2x_2 \leq 2$$
$$3x_1 + 4x_2 \leq 18$$
$$x_1 \geq 0, \quad x_2 \geq 0.$$

6. Maximize $z = 5x_1 + 3x_2$
subject to

$$\begin{aligned} 2x_1 + x_2 &\le 6 \\ 2x_1 - x_2 &\ge 0 \\ x_1 - x_2 &\le 0 \\ x_j &\ge 0, \quad j = 1,2. \end{aligned}$$

7. If a degeneracy arose in any of the exercises above, use all choices of the departing variable.

In Exercises 8 and 9,

(a) Show that cycling occurs when solving the problem using the simplex method.

(b) Use Brand's Rule to terminate cycling and obtain an optimal solution, if one exists.

8. Minimize $z = -x_1 + 7x_2 + x_3 + 2x_4$
subject to

$$\begin{aligned} x_1 + x_2 + x_3 + x_4 + x_5 &= 1 \\ \tfrac{1}{2}x_1 - \tfrac{11}{2}x_2 - \tfrac{5}{2}x_3 + 9x_4 + x_6 &= 0 \\ \tfrac{1}{2}x_1 - \tfrac{3}{2}x_2 - \tfrac{1}{2}x_3 + x_4 + x_7 &= 0 \\ x_j \ge 0, \quad j &= 1,\ldots,7 \end{aligned}$$

(due to K. T. Marshall and J. W. Suurballe).

9. Minimize $z = -\tfrac{2}{5}x_1 - \tfrac{2}{5}x_2 + \tfrac{9}{5}x_3$
subject to

$$\begin{aligned} \tfrac{3}{5}x_1 - \tfrac{32}{5}x_2 + \tfrac{24}{5}x_3 + x_4 &= 0 \\ \tfrac{1}{5}x_1 - \tfrac{9}{5}x_2 + \tfrac{3}{5}x_3 + x_5 &= 0 \\ \tfrac{2}{5}x_1 - \tfrac{8}{5}x_2 + \tfrac{1}{5}x_3 + x_6 &= 0 \\ x_2 + x_7 &= 1 \\ x_j \ge 0, \quad j &= 1,\ldots,7 \end{aligned}$$

(due to K. T. Marshall and J. W. Suurballe).

2.3 ARTIFICIAL VARIABLES

In the previous two sections we discussed the simplex algorithm as a method of solving certain types of linear programming problems. Recall that we restricted our discussion to those linear programming problems that could be put in standard form, and we considered only those problems in that form that had a *nonnegative right-hand side*. That is, we have

assumed that all our constraints were of the form

$$\sum_{j=1}^{n} a_{ij}x_j \leq b_i, \quad i = 1,2,\ldots,m, \tag{1}$$

where

$$b_i \geq 0.$$

The assumption that $\mathbf{b} \geq \mathbf{0}$ enabled us to easily find an initial basic feasible solution. For, when we introduced slack variables, we found that we could set the nonslack variables equal to zero and obtain the basic solution

$$\mathbf{x} = (0,0,\ldots,0,b_1,b_2,\ldots,b_m).$$

Furthermore, this solution was feasible since $b_i \geq 0$ for $i = 1,2,\ldots,m$.

Unfortunately, there are many linear programming problems in which not all the constraints can be transformed to the form of (1). For example, the constraint

$$2x + 3y \geq 4 \tag{2}$$

can be changed to

$$-2x - 3y \leq -4,$$

but then the right-hand side is negative. By adding a slack variable, we obtain

$$-2x - 3y + u = -4. \tag{3}$$

Setting the nonslack variables equal to zero gives us $u = -4$, which will not yield a feasible solution.

When we examine the procedure described above for finding an initial basic feasible solution, we see that it was *not* important that the procedure was applied to a problem in canonical form coming from a problem in standard form. It was important that the problem was in canonical form, that the right-hand side of $\mathbf{Ax} = \mathbf{b}$ was nonnegative, and that in each equation of the system of constraints there was a variable with coefficient $+1$ that appeared in no other equation. Then setting all but these "special" variables equal to zero, we again would have an initial basic feasible solution.

However, there may be equations in the canonical form of a problem in which no such "special" variable exists. In Equation (3) we can make the right-hand side positive so that the equation reads

$$2x + 3y - u = 4.$$

But now no coefficient is $+1$, and presumably x and y appear in other equations of the system of constraints, so that they cannot be chosen as "special " variables.

We now develop a procedure for introducing a variable that can serve as a basic variable for an initial basic feasible solution. We start with the general linear programming problem

$$\text{Maximize} \quad z = \mathbf{c}^{\mathrm{T}}\mathbf{x} \tag{4}$$

subject to

$$\left.\begin{array}{l} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n \ (\leq)(=)(\leq)\, b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n \ (\leq)(=)(\geq)\, b_2 \\ \quad\vdots \qquad\quad \vdots \qquad\qquad\quad \vdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n \ (\leq)(=)(\geq)\, b_m \end{array}\right\} \tag{5}$$

$$x_j \geq 0, \quad j = 1, 2, \ldots, n. \tag{6}$$

Note that we have assumed that the problem is a maximization problem. Any minimization problem can be converted to a maximization problem by multiplying the objective function by -1.

We now write each constraint in (5) so that its right-hand side is nonnegative. This can be done by multiplying those constraints with negative right-hand sides by -1. The constraints in (5) can now be rearranged and renumbered so that the $\leq$ constraints come first, then the $\geq$ constraints, and finally the $=$ constraints. Assume that there are r_1 $\leq$ constraints, r_2 $\geq$ constraints and r_3 $=$ constraints. Then we can assume that (5) is in the following form.

$$\left.\begin{array}{l} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n \leq b_1, \quad b_1 \geq 0 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n \leq b_2, \quad b_2 \geq 0 \\ \quad\vdots \qquad\quad \vdots \qquad\qquad\quad \vdots \qquad\quad \vdots \\ a_{r_1 1}x_1 + a_{r_1 2}x_2 + \cdots + a_{r_1 n}x_n \leq b_{r_1}, \quad b_{r_1} \geq 0 \end{array}\right\} \tag{7a}$$

$$\left.\begin{array}{l} a'_{11}x_1 + a'_{12}x_2 + \cdots + a'_{1n}x_n \geq b'_1, \quad b'_1 \geq 0 \\ a'_{21}x_1 + a'_{22}x_2 + \cdots + a'_{2n}x_n \geq b'_2, \quad b'_2 \geq 0 \\ \quad\vdots \qquad\quad \vdots \qquad\qquad\quad \vdots \qquad\quad \vdots \\ a'_{r_2 1}x_1 + a'_{r_2 2}x_2 + \cdots + a'_{r_2 n}x_n \geq b'_{r_2}, \quad b'_{r_2} \geq 0 \end{array}\right\} \tag{7b}$$

$$\left.\begin{array}{l} a''_{11}x_1 + a''_{12}x_2 + \cdots + a''_{1n}x_n = b''_1, \quad b''_1 \geq 0 \\ a''_{21}x_1 + a''_{22}x_2 + \cdots + a''_{2n}x_n = b''_2, \quad b''_2 \geq 0 \\ \quad\vdots \qquad\quad \vdots \qquad\qquad\quad \vdots \qquad\quad \vdots \\ a''_{r_3 1}x_1 + a''_{r_3 2}x_2 + \cdots + a''_{r_3 n}x_n = b''_{r_3}, \quad b''_{r_3} \geq 0 \end{array}\right\} \tag{7c}$$

Next we make each inequality in (7a) and (7b) into an equality by introducing a slack variable. Each slack variable must be nonnegative so

that in (7a) it is introduced with a + sign and in (7b) it is introduced with a − sign. We now write (7a) and (7b) as

$$\sum_{j=1}^{n} a_{ij}x_j + x_{n+i} = b_i, \quad b_i \geq 0, \quad i = 1,2,\ldots,r_1 \tag{8a}$$

$$\sum_{j=1}^{n} a'_{ij}x_j - x_{n+r_1+i} = b'_i, \quad b'_i \geq 0, \quad i = 1,2,\ldots,r_2. \tag{8b}$$

EXAMPLE 1. Consider the linear programming problem

$$\begin{aligned} &\text{Maximize} \quad z = 2x_1 + 5x_2 \\ &\text{subject to} \\ &\qquad 2x_1 + 3x_2 \leq 6 \\ &\qquad -2x_1 + x_2 \leq -2 \\ &\qquad x_1 - 6x_2 = -2 \\ &\qquad x_1 \geq 0, \quad x_2 \geq 0. \end{aligned}$$

We first rewrite the constraints so that the right-hand sides are non-negative. The problem becomes

$$\begin{aligned} &\text{Maximize} \quad z = 2x_1 + 5x_2 \\ &\text{subject to} \\ &\qquad 2x_1 + 3x_2 \leq 6 \\ &\qquad 2x_1 - x_2 \geq 2 \\ &\qquad -x_1 + 6x_2 = 2 \\ &\qquad x_1 \geq 0, \quad x_2 \geq 0. \end{aligned}$$

We now insert slack variables in each of the inequalities, obtaining an associated canonical form problem:

$$\begin{aligned} &\text{Maximize} \quad z = 2x_1 + 5x_2 \\ &\text{subject to} \\ &\left.\begin{aligned} 2x_1 + 3x_2 + x_3 &= 6 \\ 2x_1 - x_2 - x_4 &= 2 \\ -x_1 + 6x_2 &= 2 \end{aligned}\right\} \qquad (9) \\ &\qquad x_j \geq 0, \quad j = 1,2,3,4. \end{aligned}$$

However, we do not have a variable in each equation that can act as a basic variable for an initial basic feasible solution. In fact, in (9) both the second and third equations do not have such a variable. △

As Example 1 shows, we can convert the general linear programming problem in (4), (5), and (6) to an associated problem in canonical form with $\mathbf{b} \geq \mathbf{0}$. Thus, we may now assume that we have a problem in the following form:

$$\text{Maximize} \quad z = \sum_{j=1}^{s} c_j x_j \tag{10}$$

subject to

$$\sum_{j=1}^{s} a_{ij} x_j = b_i, \quad i = 1, 2, \ldots, m \tag{11}$$

$$x_j \geq 0, \quad j = 1, 2, \ldots, s \tag{12}$$

with $b_i \geq 0$, $i = 1, 2, \ldots, m$. The following method of finding an initial basic feasible solution is widely used in modern computer codes.

Two-Phase Method

To enable us to obtain an initial basic feasible solution, we introduce another variable into each equation in (11). We denote the variable for the ith equation by y_i. The variables y_i, $i = 1, 2, \ldots, m$, are called **artificial variables** and have no physical significance. Assuming the profit associated with each y_i to be zero, we obtain the problem

$$\text{Maximize} \quad z = \sum_{j=1}^{s} c_j x_j \tag{13}$$

subject to

$$\sum_{j=1}^{s} a_{ij} x_j + y_i = b_i, \quad i = 1, 2, \ldots, m \tag{14}$$

$$x_j \geq 0, \quad j = 1, 2, \ldots, s; \qquad y_i \geq 0, \quad i = 1, 2, \ldots, m \tag{15}$$

with $b_i \geq 0$, $i = 1, 2, \ldots, m$.

It is easy to see (Exercise 24) that the vector $\mathbf{x}$ in R^s is a feasible solution to the problem given by (10), (11), and (12) if and only if the vector

$$\begin{bmatrix} \mathbf{x} \\ \mathbf{0} \end{bmatrix}$$

in R^{s+m} is a feasible solution to the problem given by (13), (14), and (15). Note that it is easy to find an initial basic feasible solution to the latter problem, namely, $\mathbf{x} = \mathbf{0}$, $\mathbf{y} = \mathbf{b}$. We now develop a way to use the simplex algorithm to change this initial basic feasible solution into a basic feasible solution to the same problem in which $\mathbf{y} = \mathbf{0}$. Thus, we will have found a

basic feasible solution to the problem given by (10), (11), and (12). This procedure is Phase 1 of the **two-phase method.**

Phase 1

Since each y_i is constrainted to be nonnegative, one way of guaranteeing that each y_i is zero is to make the sum of the y_i's zero. Thus, we set up an auxiliary problem in which we minimize the sum of the y_i's subject to the constraints (14) and (15) and hope that this minimum value is zero. If this minimum value is not zero, then it must be positive, and at least one of the y_i's must be positive. Furthermore, the y_i's will never all be zero, since we have found the minimum value of their sum. Thus, in this case the original problem has no feasible solution.

We convert the canonical linear programming given by (10), (11), and (12) to the form involving artificial variables and introduce the new objective function. The resulting problem is

$$\text{Minimize} \quad z' = \sum_{i=1}^{m} y_i \tag{16}$$

subject to

$$\sum_{j=1}^{s} a_{ij}x_j + y_i = b_i, \quad i = 1,2,\ldots,m \tag{17}$$

$$x_j \geq 0, \quad j = 1,2,\ldots,s; \qquad y_i \geq 0, \quad i = 1,2,\ldots,m \tag{18}$$

with $b_i \geq 0$, $i = 1,2,\ldots,m$.

This problem has the initial basic feasible solution

$$\begin{bmatrix} 0 & 0 & \cdots & 0 & b_1 & b_2 & \cdots & b_m \end{bmatrix}^{\mathrm{T}},$$

obtained by setting

$$x_1 = 0, \quad x_2 = 0,\ldots,x_s = 0$$

as nonbasic variables and solving (17) for

$$y_1 = b_1, \quad y_2 = b_2,\ldots,y_m = b_m.$$

Writing the problem given by (16), (17), and (18) in matrix form, we find that the columns corresponding to $y_1, y_2,\ldots,y_m$ are the columns of an $m \times m$ identity matrix and, thus, are linearly independent. Therefore, this procedure yields a basic solution for the given problem.

To use the simplex method as it was developed earlier, we must multiply (16) by -1 to convert to a maximization problem and then write the result as

$$z + \sum_{i=1}^{m} y_i = 0, \tag{19}$$

where $z = -z'$. Recall that when the simplex method was first described, the initial basic variables were the slack variables, and these had zero

objective function coefficients. Consequently, the entries in the objective row in the columns labeled by the basic variables were zero initially and remained so after each pivoting step. This was necessary for the use of the optimality criterion. Therefore, we must eliminate y_i, $i = 1,2,\ldots,m$, from (19). We can do this by solving (17) for y_i.

Now

$$y_i = b_i - \sum_{j=1}^{s} a_{ij}x_j,$$

and, substituting into (19), we obtain

$$z + \sum_{i=1}^{m}\left(b_i - \sum_{j=1}^{s} a_{ij}x_j\right) = 0.$$

Rearranging, we can write the objective equation as

$$z - \sum_{j=1}^{s}\left(\sum_{i=1}^{m} a_{ij}\right)x_j = -\sum_{i=1}^{m} b_i. \tag{20}$$

We can now solve the problem as given by (20), (17), and (18) using the simplex method.

The reader can experiment with problems that are not in standard form using the SMPX courseware described in Appendix C. One must include artificial variables when entering the problem and must indicate to SMPX that these variables are artificial.

EXAMPLE 2. Consider the linear programming problem in canonical form

$$\begin{aligned}
&\text{Maximize} \quad z = x_1 - 2x_2 - 3x_3 - x_4 - x_5 + 2x_6\\
&\text{subject to}\\
&\quad x_1 + 2x_2 + 2x_3 + x_4 + x_5 = 12\\
&\quad x_1 + 2x_2 + x_3 + x_4 + 2x_5 + x_6 = 18\\
&\quad 3x_1 + 6x_2 + 2x_3 + x_4 + 3x_5 = 24\\
&\quad x_j \geq 0, \quad j = 1,2,\ldots,6.
\end{aligned}$$

Introducing the artificial variables y_1, y_2, and y_3, we can write the auxiliary problem as

$$\begin{aligned}
&\text{Minimize} \quad z' = y_1 + y_2 + y_3 \qquad (21)\\
&\text{subject to}\\
&\left.\begin{aligned}
x_1 + 2x_2 + 2x_3 + x_4 + x_5 + y_1 &= 12\\
x_1 + 2x_2 + x_3 + x_4 + 2x_5 + x_6 + y_2 &= 18\\
3x_1 + 6x_2 + 2x_3 + x_4 + 3x_5 + y_3 &= 24
\end{aligned}\right\} \qquad (22)\\
&x_j \geq 0, \quad j = 1,2,\ldots,6; \qquad y_i \geq 0, \quad i = 1,2,3 \qquad (23)
\end{aligned}$$

After conversion to a maximization problem, the objective function (21) can be written as

$$z' + y_1 + y_2 + y_3 = 0. \tag{24}$$

To eliminate y_1, y_2, and y_3 from (24) we add -1 times each of the constraints in (22) to (24), obtaining

$$z' - 5x_1 - 10x_2 - 5x_3 - 3x_4 - 6x_5 - x_6 = -54. \tag{25}$$

The initial basic feasible solution is

$$x_1 = x_2 = x_3 = x_4 = x_5 = x_6 = 0 \quad \text{(nonbasic variables)}$$
$$y_1 = 12, \quad y_2 = 18, \quad y_3 = 24 \quad \text{(basic variables)}.$$

Using (25), (22), and (23), we can write the initial tableau (Tableau 2.25).

Tableau 2.25

		x_1	x_2 ↓	x_3	x_4	x_5	x_6	y_1	y_2	y_3	
	y_1	1	2	2	1	1	0	1	0	0	12
	y_2	1	2	1	1	2	1	0	1	0	18
←	y_3	3	⑥	2	1	3	0	0	0	1	24
		−5	−10	−5	−3	−6	−1	0	0	0	−54

The most negative entry in the objective row is -10, so that x_2 is the entering variable and the second column is the pivotal column. The departing variable is determined by computing the minimum of the θ-ratios. We have

$$\min\{\tfrac{12}{2}, \tfrac{18}{2}, \tfrac{24}{6}\} = \tfrac{24}{6} = 4,$$

so that the row labeled by y_3 is the pivotal row and y_3 is the departing variable. The pivot is 6.

We obtain Tableau 2.26 from Tableau 2.25 by pivoting (verify). In Tableau 2.26 we choose x_3 as the entering variable and y_1 as the departing variable. We form Tableau 2.27 by pivoting.

There is a tie in determining the entering variable in Tableau 2.27. We choose x_6 as the entering variable and obtain Tableau 2.28.

Tableau 2.26

		x_1	x_2	x_3	x_4	x_5	x_6	y_1	y_2	y_3	
←	y_1	0	0	(4/3)	2/3	0	0	1	0	−1/3	4
	y_2	0	0	1/3	2/3	1	1	0	1	−1/3	10
	x_2	1/2	1	1/3	1/6	1/2	0	0	0	1/6	4
		0	0	−5/3	−4/3	−1	−1	0	0	5/3	−14

Tableau 2.27

		x_1	x_2	x_3	x_4	x_5	x_6	y_1	y_2	y_3	
	x_3	0	0	1	1/2	0	0	3/4	0	−1/4	3
←	y_2	0	0	0	1/2	1	(1)	−1/4	1	−1/4	9
	x_2	1/2	1	0	0	1/2	0	−1/4	0	1/4	3
		0	0	0	−1/2	−1	−1	5/4	0	5/4	−9

Tableau 2.28

		x_1	x_2	x_3	x_4	x_5	x_6	y_1	y_2	y_3	
	x_3	0	0	1	1/2	0	0	3/4	0	−1/4	3
	x_6	0	0	0	1/2	1	1	−1/4	1	−1/4	9
	x_2	1/2	1	0	0	1/2	0	−1/4	0	1/4	3
		0	0	0	0	0	0	1	1	1	0

At this point all the artificial variables are nonbasic variables and have value zero. Tableau 2.28 gives the basic feasible solution

$$x_2 = 3, \quad x_3 = 3, \quad x_6 = 9$$
$$x_1 = x_4 = x_5 = y_1 = y_2 = y_3 = 0$$

with objective function value $z = 0$.

The reader can verify that $\mathbf{x} = [0 \quad 3 \quad 3 \quad 0 \quad 0 \quad 9]^T$ is a basic feasible solution to the original problem without artificial variables. The introduction of the artificial variables gave a systematic way of finding such a basic feasible solution. △

If the solution to Phase 1 is a set of values for the variables that makes the objective function of the auxiliary problem equal to zero, then we may start Phase 2. There are two possibilities: (1) every artificial variable is a nonbasic variable in the final tableau of Phase 1, or (2) some artificial variables are still basic variables, with value 0, in the final tableau (see

Example 5). At this time we shall discuss only the first case. The second case is discussed in Section 3.3.

Phase 2

We assume that no artificial variable is a basic variable at the end of Phase 1 and the value of the objective function of the auxiliary problem is zero. The optimal solution obtained in Phase 1 is used to obtain an initial basic feasible solution for the original problem (10), (11), (12). By deleting the y_i's from the optimal solution, we obtain a basic feasible solution to the original problem because we have assumed that no artificial variables appear in the optimal solution. The initial tableau for Phase 2 is the final tableau of Phase 1, with the following modifications.

(a) Delete the columns labeled with artificial variables.

(b) Calculate a new objective row as follows. Start with the objective function of the original problem (10). For each of the basic variables in the final tableau of Phase 1, make the entry in the objective row in the column labeled by that basic variable equal to zero by adding a suitable multiple of the row labeled by that basic variable.

These two steps construct an initial tableau for Phase 2. We can now apply the simplex algorithm to this tableau.

EXAMPLE 2 (CONTINUED). We now form the initial tableau for Phase 2 from Tableau 2.28. We delete the columns labeled with y_1, y_2, and y_3 and then use the original objective function,

$$z = x_1 - 2x_2 - 3x_3 - x_4 - x_5 + 2x_6.$$

The objective row would be

$$\boxed{\begin{array}{ccccccc} -1 & 2 & 3 & 1 & 1 & -2 & 0 \end{array}} \tag{26}$$

But the entries in the second, third, and sixth columns must be zeroed for the optimality criterion to hold, since x_2, x_3, and x_6 are basic variables. We do this by adding -2 times the x_2 row to (26); also, we add -3 times the x_3 row and 2 times the x_6 row to (26). This calculation yields

-1	2	3	1	1	-2	0	[Equation (26)]
-1	-2	0	0	-1	0	-6	(-2 times x_2 row)
0	0	-3	$-\frac{3}{2}$	0	0	-9	(-3 times x_3 row)
0	0	0	1	2	2	18	(2 times x_6 row)
-2	0	0	$\frac{1}{2}$	2	0	3	(objective row for Phase 2).

We then have the initial tableau for Phase 2 (Tableau 2.29), in which, as usual, we have not included the z column.

We now continue with the simplex algorithm to find an optimal solution. The next tableau is Tableau 2.30 (verify).

Since the objective row in Tableau 2.30 has no negative entries, we have found an optimal solution,

$$x_1 = 6, \quad x_2 = 0, \quad x_3 = 3, \quad x_4 = 0, \quad x_5 = 0, \quad x_6 = 9,$$

which gives $z = 15$ as the value of the objective function. △

Tableau 2.29

	x_1	x_2	x_3	x_4	x_5	x_6	
x_3	0	0	1	$\frac{1}{2}$	0	0	3
x_6	0	0	0	$\frac{1}{2}$	1	1	9
← x_2	(1/2)	1	0	0	$\frac{1}{2}$	0	3
	−2	0	0	$\frac{1}{2}$	2	0	3

Tableau 2.30

	x_1	x_2	x_3	x_4	x_5	x_6	
x_3	0	0	1	$\frac{1}{2}$	0	0	3
x_6	0	0	0	$\frac{1}{2}$	1	1	9
x_1	1	2	0	0	1	0	6
	0	4	0	$\frac{1}{2}$	4	0	15

We had originally put an artificial variable in every equation. This method requires no decision steps but may cause more tableaux to be computed. If some of the equations have a variable that can be used as a basic variable, then it is not necessary to introduce an artificial variable into each of those equations.

EXAMPLE 3. In the problem discussed in Example 2, we see that x_6 can be used as a basic variable. That is, it appears in only one equation and, in that equation, has a coefficient of $+1$. Consequently, we need to introduce artificial variables only into the first and third equations. Doing this, we obtain the auxiliary problem

$$\text{Minimize} \quad z' = y_1 + y_2$$

$$\text{subject to}$$

$$\begin{aligned} x_1 + 2x_2 + 2x_3 + x_4 + x_5 + y_1 &= 12 \\ x_1 + 2x_2 + x_3 + x_4 + 2x_5 + x_6 &= 18 \\ 3x_1 + 6x_2 + 2x_3 + x_4 + 3x_5 + y_2 &= 12 \end{aligned}$$

$$x_j \geq 0, \quad j = 1, 2, \ldots, 6; \qquad y_1 \geq 0, \quad y_2 \geq 0.$$

As before, we must eliminate the basic variables from the objective function, rewritten as in (24) by adding -1 times each of the constraints that includes an artificial variable to the rewritten objective function. We obtain

$$z - 4x_1 - 8x_2 - 4x_3 - 2x_4 - 4x_5 = -36.$$

This now leads to the sequence of Tableaux 2.31, 2.32, and 2.33.

In Tableau 2.33 we have an optimal solution to the auxiliary problem in which all the artificial variables have value zero and are nonbasic variables. Thus, a basic feasible solution to the original problem is

$$[0 \quad 3 \quad 3 \quad 0 \quad 0 \quad 9]^T,$$

as we obtained in Example 2. △

The system of equations (11) that gives the constraints for the canonical form of a linear programming problem always must have a solution

Tableau 2.31

		x_1	x_2	x_3	x_4	x_5	x_6	y_1	y_2	
	y_1	1	2	2	1	1	0	1	0	12
	x_6	1	2	1	1	2	1	0	0	18
←	y_2	3	⑥	2	1	3	0	0	1	24
		−4	−8	−4	−2	−4	0	0	0	−36

Tableau 2.32

		x_1	x_2	x_3	x_4	x_5	x_6	y_1	y_2	
←	y_1	0	0	⑷/3	2/3	0	0	1	−1/3	4
	x_6	0	0	1/3	2/3	1	1	0	−1/3	10
	x_2	1/2	1	1/3	1/6	1/2	0	0	1/6	4
		0	0	−4/3	−2/3	0	0	0	4/3	−4

Tableau 2.33

	x_1	x_2	x_3	x_4	x_5	x_6	y_1	y_2	
x_3	0	0	1	1/2	0	0	3/4	−1/4	3
x_6	0	0	0	1/2	1	1	−1/4	−1/4	9
x_2	1/2	1	0	0	1/2	0	−1/4	1/4	3
	0	0	0	0	0	0	1	1	0

if $m \leq s$ and if there are m linearly independent columns in the coefficient matrix. But it is possible that none of the solutions satisfies the nonnegative criterion (12). When the artificial variables are added to the system of constraints (14), there is always a solution,

$$\begin{bmatrix} 0 & 0 & \cdots & 0 & b_1 & b_2 & \cdots & b_m \end{bmatrix}^T,$$

that satisfies the nonnegativity conditions (15). A solution to the system in (14) is a solution to the system in (11) if all the artificial variables have values equal to zero. We use the simplex algorithm to try to find a solution to (14) in which all the artificial variables are zero. The simplex algorithm is designed to find only solutions that satisfy the nonnegativity requirements. Thus, if one of the y_i is positive when the simplex algorithm reaches an optimal solution to the auxiliary problem, this indicates that there are no solutions to (11) that satisfy the nonnegativity constraints (12). In this case, the original problem (10), (11), (12) has no feasible solutions. The following example illustrates this situation.

EXAMPLE 4. Consider the general linear programming problem

$$\begin{aligned} &\text{Maximize} \quad z = 2x_1 + 5x_2 \\ &\text{subject to} \\ &\qquad 2x_1 + 3x_2 \leq 6 \\ &\qquad x_1 + x_2 \geq 4 \\ &\qquad x_1 \geq 0, \quad x_2 \geq 0. \end{aligned}$$

By inserting slack variables x_3 and x_4, we can write the problem in canonical form as

$$\begin{aligned} &\text{Maximize} \quad z = 2x_1 + 5x_2 \\ &\text{subject to} \\ &2x_1 + 3x_2 + x_3 = 6 \\ &x_1 + x_2 - x_4 = 4 \\ &x_j \geq 0, \quad j = 1,2,3,4. \end{aligned}$$

We insert an artificial variable y into the second equation; x_3 can serve as the basic variable in the first equation. The auxiliary problem then has the form

$$\begin{aligned} &\text{Minimize} \quad z' = y \\ &\text{subject to} \\ &2x_1 + 3x_2 + x_3 = 6 \\ &x_1 + x_2 - x_4 + y = 4 \\ &x_j \geq 0, \quad j = 1,2,3,4; \quad y \geq 0. \end{aligned}$$

After adding -1 times the second constraint to the rewritten objective function, we obtain

$$z' - x_1 - x_2 + x_4 = -4.$$

We then calculate the sequence of Tableaux 2.34, 2.35, and 2.36.

Tableau 2.34

	x_1	x_2	x_3	x_4	y	
x_3	2	③	1	0	0	6
y	1	1	0	-1	1	4
	-1	-1	0	1	0	-4

Tableau 2.35

	x_1	x_2	x_3	x_4	y	
x_2	⅔	1	⅓	0	0	2
y	⅓	0	$-\frac{1}{3}$	-1	1	2
	$-\frac{1}{3}$	0	⅓	1	0	-2

Tableau 2.36

	x_1	x_2	x_3	x_4	y	
x_1	1	$\frac{3}{2}$	$\frac{1}{2}$	0	0	3
y	0	$-\frac{1}{2}$	$-\frac{1}{2}$	-1	1	1
	0	$\frac{1}{2}$	$\frac{1}{2}$	1	0	-1

Tableau 2.36 represents an optimal solution to Phase 1 in which the artificial variable y has the value 1. This means that the given problem has no feasible solution. When we look at the graphs of the constraints (Figure 2.4), we see that there is no intersection between the two half-spaces. The set of feasible solutions is empty. △

We have already pointed out that an artificial variable can appear in an optimal solution to the auxiliary problem with a value of zero. In this case the given problem has a feasible solution. The following example illustrates these ideas. We can complete Phase 1 in this section; in Section 3.3 we will develop tools for handling Phase 2.

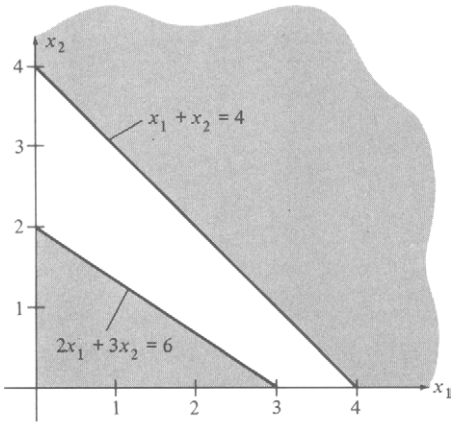


FIGURE 2.4

EXAMPLE 5. Consider the linear programming problem in canonical form

$$\text{Maximize} \quad z = x_1 + 2x_2 + x_3$$

subject to

$$\begin{aligned} 3x_1 + \ x_2 - x_3 &= 15 \\ 8x_1 + 4x_2 - x_3 &= 50 \\ 2x_1 + 2x_2 + x_3 &= 20 \\ x_1 \geq 0, \quad x_2 \geq 0, \quad x_3 &\geq 0. \end{aligned}$$

In this problem we must introduce an artificial variable in each of the constraint equations. We obtain the auxiliary problem

$$\text{Maximize} \quad z' = y_1 + y_2 + y_3$$

subject to

$$\begin{aligned} 3x_1 + \ x_2 - x_3 + y_1 \qquad\quad &= 15 \\ 8x_1 + 4x_2 - x_3 \qquad + y_2 \qquad &= 50 \\ 2x_1 + 2x_2 + x_3 \qquad\quad + y_3 &= 20 \\ x_j \geq 0, \quad j = 1,2,3; \qquad y_i \geq 0, \quad i &= 1,2,3 \end{aligned}$$

Rewriting the objective function in the same manner as before, we obtain the initial tableau (Tableau 2.37) for the auxiliary problem. At the end of Phase 1 we obtain Tableau 2.38 which represents an optimal solution to the auxiliary problem with $y_2 = 0$ as a basic variable. △

Tableau 2.37

		x_1	x_2	x_3	y_1	y_2	y_3	
←	y_1	③	1	−1	1	0	0	15
	y_2	8	4	−1	0	1	0	50
	y_3	2	2	1	0	0	1	20
		−13	−7	1	0	0	0	−85

Tableau 2.38

	x_1	x_2	x_3	y_1	y_2	y_3	
x_1	1	$\frac{3}{5}$	0	$\frac{1}{5}$	0	$\frac{1}{5}$	7
y_2	0	0	0	−2	1	−1	0
x_3	0	$\frac{4}{5}$	1	$-\frac{2}{5}$	0	$\frac{3}{5}$	6
	0	0	0	3	0	2	0

Figure 2.5 gives a flowchart and Figure 2.6, a structure diagram that summarizes the two-phase method.

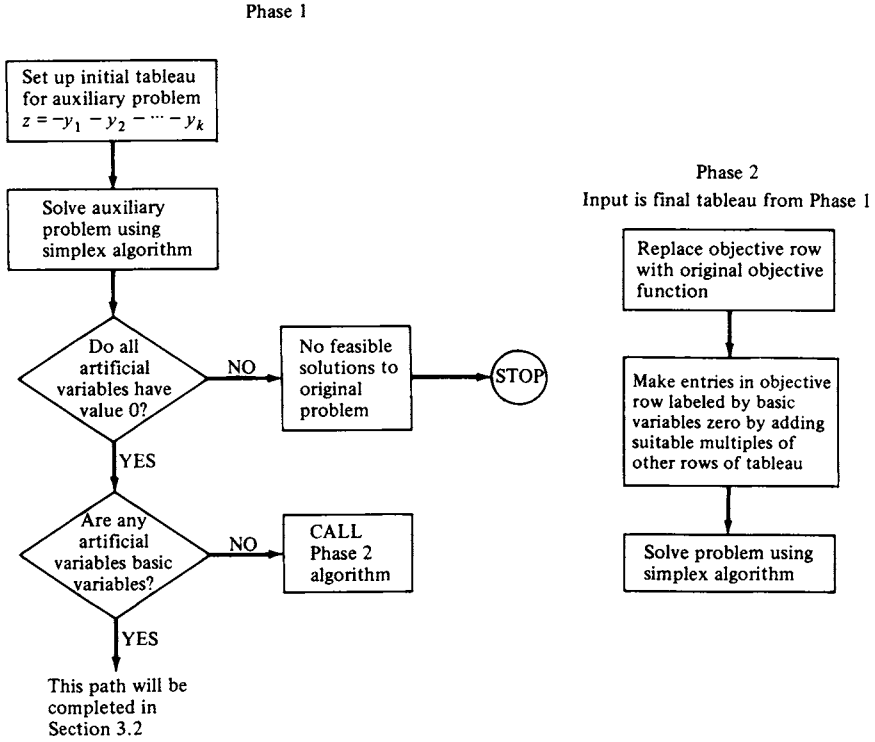


FIGURE 2.5 Flowchart of the two-phase algorithm.

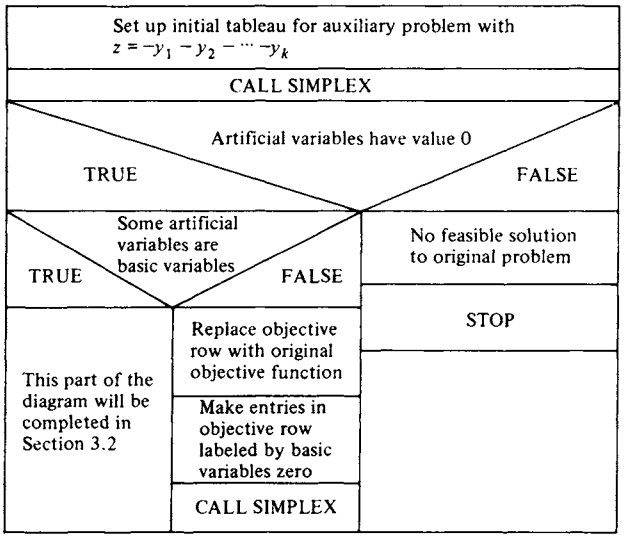


FIGURE 2.6 Structure diagram of the two-phase algorithm.

Big *M* Method (Optional)

The following method of solving linear programming problems that require artificial variables is attributed to Charnes. Historically it precedes the two-phase method; it has been replaced by the latter in computer codes due to the greater efficiency of the two-phase method. It is still of interest for theoretical computations.

Instead of introducing an auxiliary problem, the **big *M* method** ensures that the artificial variables are zero in the final solution by assigning to each y_i a **penalty cost**, $-M$, where M is a very large positive number. This means that we use an objective function of the form

$$z = \sum_{j=1}^{s} c_j x_j - \sum_{i=1}^{m} M y_i.$$

If any y_i is positive, the $-M$ serves to decrease z drastically.

We convert the canonical linear programming problem given by (10), (11), and (12) to the form involving artificial variables. We obtain

$$\text{Maximize} \quad z = \sum_{j=1}^{s} c_j x_j - \sum_{i=1}^{m} M y_i \tag{27}$$

subject to

$$\sum_{j=1}^{s} a_{ij} x_j + y_i = b_i, \quad i = 1, 2, \ldots, m \tag{28}$$

$$x_j \geq 0, \quad j = 1, 2, \ldots, s; \qquad y_i \geq 0, \quad i = 1, 2, \ldots, m \tag{29}$$

with $b_i \geq 0$, $i = 1, 2, \ldots, m$.

This problem has an initial basic feasible solution,

$$[0 \quad 0 \quad \cdots \quad 0 \quad b_1 \quad b_2 \quad \cdots \quad b_m]^T,$$

obtained by setting

$$x_1 = 0, \quad x_2 = 0, \ldots, x_s = 0$$

as nonbasic variables and solving (28) for

$$y_1 = b_1, \quad y_2 = b_2, \ldots, y_m = b_m.$$

To use the simplex method as it was developed earlier, we must write (27) as

$$z - \sum_{j=1}^{s} c_j x_j + M \sum_{i=1}^{m} y_i = 0 \tag{30}$$

and eliminate the y_i from this equation. This is done by adding $-M$ times each of the constraints in (28) to (30). We obtain

$$z - \sum_{j=1}^{s} c_j x_j + M \sum_{i=1}^{m} \left(b_i - \sum_{j=1}^{s} a_{ij} x_j \right) = 0.$$

Rearranging, we can write the previous equation as

$$z - \sum_{j=1}^{s} \left(c_j + M \sum_{i=1}^{m} a_{ij} \right) x_j = -M \sum_{i=1}^{m} b_i. \tag{31}$$

We can now solve the problem as given by (31), (28), and (29) using the simplex method.

EXAMPLE 6. Consider the linear programming problem in Example 2. Since x_6 appears in only one constraint, and there with a coefficient of $+1$, we may use x_6 as a basic variable. Thus, we introduce artificial variables into only the first and third equations, obtaining the problem

$$\text{Maximize} \quad z = x_1 - 2x_2 - 3x_3 - x_4 - x_5 + 2x_6 - My_1 - My_2$$

subject to

$$\begin{aligned} x_1 + 2x_2 + 2x_3 + x_4 + x_5 \qquad + y_1 \qquad &= 12 \\ x_1 + 2x_2 + x_3 + x_4 + 2x_5 + x_6 \qquad &= 18 \\ 3x_1 + 6x_2 + 2x_3 + x_4 + 3x_5 \qquad + y_2 &= 24 \\ x_j \geq 0, \quad j = 1, 2, \ldots, 6; \qquad y_1 \geq 0, \quad y_2 \geq 0. \end{aligned}$$

We rewrite the objective function as

$$z - x_1 + 2x_2 + \cdots - 2x_6 + My_1 + My_2$$

and eliminate the basic variables x_6, y_1, and y_2 from this equation. This is done by adding

$$\begin{aligned}(-M) &\times \text{first constraint}\\ 2 &\times \text{second constraint}\\ (-M) &\times \text{third constraint.}\end{aligned}$$

These operations lead to the equations

$$\begin{aligned}z - x_1 + 2x_2 + 3x_3 + x_4 + x_5 - 2x_6 + My_1 + My_2 &= 0\\ -Mx_1 - 2Mx_2 - 2Mx_3 - Mx_4 - Mx_5 - My_1 &= -12M\\ 2x_1 + 4x_2 + 2x_3 + 2x_4 + 4x_5 + 2x_6 &= 36\\ -3Mx_1 - 6Mx_2 - 2Mx_3 - Mx_4 - 3Mx_5 - My_2 &= -24M.\end{aligned}$$

The result of adding these equations is

$$z + (1 - 4M)x_1 + (6 - 8M)x_2 + (5 - 4M)x_3 + (3 - 2M)x_4 + (5 - 4M)x_5 = 36 - 36M,$$

from which we obtain the initial tableau (Tableau 2.39).

Tableau 2.39

		x_1	x_2	x_3	x_4	x_5	x_6	y_1	y_2	
	y_1	1	2	2	1	1	0	1	0	12
	x_6	1	2	1	1	2	1	0	0	18
←	y_2	3	⑥	2	1	3	0	0	1	24
		$1-4M$	$6-8M$	$5-4M$	$3-2M$	$5-4M$	0	0	0	$36-36M$

(↓ x_2)

Since M is a large positive number, the most negative entry in the objective row is $6 - 8M$, so that x_2 is the entering variable. The departing variable, obtained as usual, is y_2. Pivoting, we obtain Tableau 2.40. Using the same reasoning, we obtain Tableaux 2.41 and 2.42.

Since the objective row has no negative entries, we have found an optimal solution:

$$x_1 = 6, \quad x_2 = 0, \quad x_3 = 3, \quad x_4 = 0, \quad x_5 = 0, \quad x_6 = 9$$
$$y_1 = 0, \quad y_2 = 0,$$

which gives $z = 15$ as the value of the objective function. This solution coincides with the one obtained in Example 2. △

Tableau 2.40

		x_1	x_2	x_3 ↓	x_4	x_5	x_6	y_1	y_2	
←	y_1	0	0	$\frac{4}{3}$	$\frac{2}{3}$	0	0	1	$-\frac{1}{3}$	4
	x_6	0	0	$\frac{1}{3}$	$\frac{2}{3}$	1	1	0	$-\frac{1}{3}$	10
	x_2	$\frac{1}{2}$	1	$\frac{1}{3}$	$\frac{1}{6}$	$\frac{1}{2}$	0	0	$\frac{1}{6}$	4
		-2	0	$3-\frac{4}{3}M$	$2-\frac{2}{3}M$	2	0	0	$-1+\frac{4}{3}M$	$12-4M$

Tableau 2.41

		x_1 ↓	x_2	x_3	x_4	x_5	x_6	y_1	y_2	
	x_3	0	0	1	$\frac{1}{2}$	0	0	$\frac{3}{4}$	$-\frac{1}{4}$	3
	x_6	0	0	0	$\frac{1}{2}$	1	1	$-\frac{1}{4}$	$-\frac{1}{4}$	9
←	x_2	$\frac{1}{2}$	1	0	0	$\frac{1}{2}$	0	$-\frac{1}{4}$	$\frac{1}{4}$	3
		-2	0	0	$\frac{1}{2}$	2	0	$-\frac{9}{4}+M$	$-\frac{1}{4}+M$	3

Tableau 2.42

	x_1	x_2	x_3	x_4	x_5	x_6	y_1	y_2	
x_3	0	0	1	$\frac{1}{2}$	0	0	$\frac{3}{4}$	$-\frac{1}{4}$	3
x_6	0	0	0	$\frac{1}{2}$	1	1	$-\frac{1}{4}$	$-\frac{1}{4}$	9
x_1	1	2	0	0	1	0	$-\frac{1}{2}$	$\frac{1}{2}$	6
	0	4	0	$\frac{1}{2}$	4	0	$-\frac{13}{4}+M$	$\frac{3}{4}+M$	15

2.3 EXERCISES

In Exercises 1–4 set up the initial simplex tableau (a) for solving the problem using the two-phase method and (b) for solving the problem using the big M method.

1. Maximize $z = x_1 + 3x_3$
   subject to
   $$x_1 + 2x_2 + 7x_3 = 4$$
   $$x_1 + 3x_2 + x_3 = 5$$
   $$x_1 \geq 0, \quad x_2 \geq 0, \quad x_3 \geq 0.$$

2. Maximize $z = x_1 + 2x_2 + x_4$
   subject to
   $$x_1 + 3x_2 - x_3 + x_4 \leq 5$$
   $$x_1 + 7x_2 + x_3 \geq 4$$
   $$4x_1 + 2x_2 + x_4 = 3$$
   $$x_j \geq 0, \quad j = 1,2,3,4.$$

3. Minimize $z = 3x_1 - 2x_2$
subject to

$$\begin{aligned} x_1 + \ x_2 + 2x_3 &\geq 7 \\ 2x_1 + \ x_2 + \ x_3 &\geq 4 \\ x_1 \geq 0, \quad x_2 &\geq 0, \quad x_3 \geq 0. \end{aligned}$$

4. Minimize $z = x_1 + 2x_2 + 7x_3 - x_4$
subject to

$$\begin{aligned} 3x_1 + \ x_2 - 2x_3 - x_4 &= 2 \\ 2x_1 + 4x_2 + 7x_3 \qquad &\geq 3 \\ x_j \geq 0, \quad j &= 1,2,3,4. \end{aligned}$$

In Exercises 5 and 6 carry out Phase 1 for the given problems.

5. Maximize $z = 3x_1 - 4x_3$
subject to

$$\begin{aligned} 2x_1 + x_2 + 3x_3 &\geq 5 \\ x_1 - x_2 + \ x_3 &\geq 1 \\ x_1 \geq 0, \quad x_2 &\geq 0, \quad x_3 \geq 0. \end{aligned}$$

6. Maximize $z = x_1 + x_2 + 2x_4$
subject to

$$\begin{aligned} 3x_1 + \ x_2 + 3x_3 + 2x_4 &= 10 \\ x_1 - 3x_2 + 2x_3 \qquad &\leq 7 \\ x_1 + 2x_2 + 3x_3 + \ x_4 &\geq 4 \\ x_j \geq 0, \quad j &= 1,2,3,4. \end{aligned}$$

In Exercises 7–9 we give the final tableau for Phase 1 of the two-phase method along with the original objective function. In these tableaux we use $y_1, y_2, \ldots$ to denote artificial variables. (a) Form the initial tableau for Phase 2 using the given information and (b) apply the simplex method to the tableau in (a) to find an optimal solution to the given problem.

7. Maximize $z = 2x_1 + x_2 + x_3$

	x_1	x_2	x_3	x_4	x_5	
x_2	-1	1	$\frac{3}{10}$	0	$-\frac{1}{2}$	$\frac{3}{5}$
x_4	$\frac{1}{2}$	0	$\frac{1}{5}$	1	$-\frac{1}{10}$	$\frac{7}{10}$
	0	0	0	0	0	0

8. Maximize $z = x_2 + 3x_3 + x_4$

	x_1	x_2	x_3	x_4	x_5	x_6	x_7	
x_7	$-\frac{3}{2}$	0	-2	-1	0	$-\frac{3}{4}$	1	0
x_2	0	1	1	3	0	$\frac{1}{2}$	0	2
x_5	1	0	3	0	1	$-\frac{1}{2}$	0	4
	0	0	0	0	0	0	0	0

9. Maximize $z = 2x_1 + x_2 + x_4$

	x_1	x_2	x_3	x_4	x_5	x_6	y_1	y_2	
x_2	$\frac{3}{8}$	1	1	0	0	$\frac{3}{4}$	0	0	1
x_5	$-\frac{5}{8}$	0	1	1	1	$\frac{7}{4}$	0	0	2
y_1	-1	0	0	-2	0	$-\frac{5}{4}$	1	0	$\frac{3}{2}$
y_2	0	0	$-\frac{1}{2}$	1	0	$-\frac{1}{4}$	0	1	0
	1	0	$\frac{1}{2}$	1	0	$\frac{3}{2}$	0	0	$-\frac{3}{2}$

In Exercises 10–23 solve the indicated linear programming problem using either the two-phase method or the big M method.

10. Example 2, Section 1.1.
11. Example 5, Section 1.1.
12. Example 6, Section 1.1.
13. Example 7(c), Section 1.1.
14. Example 7(d), Section 1.1.
15. Exercise 1, Section 1.1.
16. Exercise 11, Section 1.1.
17. Exercise 12, Section 1.4.
18. Exercise 3, Section 1.5.
19. Exercise 8, Section 1.1.
20. Maximize $z = 2x_1 + 5x_2 - x_3$
subject to

$$-4x_1 + 2x_2 + 6x_3 = 4$$
$$6x_1 + 9x_2 + 12x_3 = 3$$
$$x_1 \geq 0, \quad x_2 \geq 0, \quad x_3 \geq 0.$$

21. Maximize $z = 3x_1 - x_2 + 2x_3 + 4x_4$
subject to

$$\begin{aligned} x_2 + 7x_3 + 2x_4 &\geq 3 \\ x_1 + 2x_2 + x_3 &= 9 \\ 2x_1 + 3x_2 + x_3 - 4x_4 &\leq 7 \\ x_j \geq 0, \quad j &= 1,2,3,4. \end{aligned}$$

22. Maximize $z = 2x_1 - x_2 + x_3 - x_4 + x_5$
subject to

$$\begin{aligned} x_1 + x_2 - x_3 + x_4 + x_5 &= 3 \\ 2x_1 - x_2 + x_3 - 2x_4 &= 2 \\ 3x_1 - x_3 + 3x_4 &\geq 2 \\ x_j \geq 0, \quad j &= 1,2,3,4. \end{aligned}$$

23. Maximize $z = 3x_1 + x_2 - x_3 + 2x_4 - x_5 + 2x_6$
subject to

$$\begin{aligned} 2x_1 + x_2 - x_3 + x_6 &= 3 \\ 3x_1 + 2x_3 + x_4 + 2x_5 &= 4 \\ x_2 - 3x_3 + x_5 &= 2 \\ x_j \geq 0, \quad j &= 1,2,\ldots,6. \end{aligned}$$

24. Show that the vector $\mathbf{x}$ in R^s is a feasible solution to the problem in canonical form given by (10), (11), and (12) if and only if the vector

$$\begin{bmatrix} \mathbf{x} \\ \mathbf{0} \end{bmatrix}$$

in R^{s+m} is a feasible solution to the auxiliary problem given by (13), (14), and (15).

25. Explain why the coefficients of M in the objective row of Tableau 2.40 are the same as the nonzero entries in the objective row of Tableau 2.32.

Further Reading

Beale, E. M. L. "Cycling in the Dual Simplex Algorithm." *Naval Res. Logistics Q.* **2** (1955), 269–276.

Kotiah, Thoddi C. T., and Steinberg, D. I. "Occurrences in Cycling and Other Phenomena Arising in a Class of Linear Programming Models." *Commun. ACM* **20** (1977), 107–112.

Kotiah, Thoddi C. T., and Steinberg, D. I. "On the Possibility of Cycling with the Simplex Method." *Operations Res.* **26** (1978), 374–375.

Marshall, K. T., and Suurballe, J. W. "A Note on Cycling in the Simplex Method." *Naval Res. Logistics Q.* **16** (1969), 121–137.

3

Further Topics in Linear Programming

THIS CHAPTER COVERS several topics in linear programming that have important computational consequences. The idea of duality, which is introduced in the second section, is particularly useful in modeling, because it provides economic interpretations of the solution to a linear programming problem. We present a brief discussion of sensitivity analysis, another tool that is useful in interpreting the solution to a linear programming problem. We discuss several variants of the simplex algorithm, including the one that is used in most computer codes. Finally, we deal with computational considerations from the viewpoint of the user of a packaged linear programming system.

3.1 DUALITY

In this section we shall show how to associate a minimization problem with each linear programming problem in standard form. There are some very interesting interpretations of the associated problem that we will

discuss. Generally, a problem in standard form can be thought of as a manufacturing problem, one in which scarce resources are allocated in a way that maximizes profit. The associated minimization problem is one that seeks to minimize cost.

Consider the pair of linear programming problems

$$\left.\begin{array}{l}\text{Maximize} \quad z = \mathbf{c}^{\mathrm{T}}\mathbf{x} \\ \text{subject to} \\ \qquad \mathbf{A}\mathbf{x} \leq \mathbf{b} \\ \qquad \mathbf{x} \geq \mathbf{0}\end{array}\right\} \tag{1}$$

and

$$\left.\begin{array}{l}\text{Minimize} \quad z' = \mathbf{b}^{\mathrm{T}}\mathbf{w} \\ \text{subject to} \\ \qquad \mathbf{A}^{\mathrm{T}}\mathbf{w} \geq \mathbf{c} \\ \qquad \mathbf{w} \geq \mathbf{0},\end{array}\right\} \tag{2}$$

where $\mathbf{A}$ is an $m \times n$ matrix, $\mathbf{c}$ and $\mathbf{x}$ are $n \times 1$ column vectors, and $\mathbf{b}$ and $\mathbf{w}$ are $m \times 1$ column vectors.

These problems are called **dual problems**. The problem given by (1) is called the **primal problem**; the problem given by (2) is called the **dual problem.**

EXAMPLE 1. If the primal problem is

$$\text{Maximize} \quad z = \begin{bmatrix} 2 & 3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

subject to

$$\begin{bmatrix} 3 & 2 \\ -1 & 2 \\ 4 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \leq \begin{bmatrix} 2 \\ 5 \\ 1 \end{bmatrix}$$

$$x_1 \geq 0, \quad x_2 \geq 0,$$

then the dual problem is

$$\text{Minimize} \quad z' = \begin{bmatrix} 2 & 5 & 1 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix}$$

subject to

$$\begin{bmatrix} 3 & -1 & 4 \\ 2 & 2 & 1 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix} \geq \begin{bmatrix} 2 \\ 3 \end{bmatrix}$$

$$w_1 \geq 0, \quad w_2 \geq 0, \quad w_3 \geq 0. \qquad \Delta$$

Observe that, in forming the dual problem, the coefficients of the ith constraint of the primal problem became the coefficients of the variable w_i in the constraints of the dual problem. Conversely, the coefficients of x_j became the coefficients of the jth constraint in the dual problem. Also, the coefficients of the objective function of the primal problem became the right-hand sides of the constraints of the dual problem, and conversely.

THEOREM 3.1. *Given a primal problem as in (1), the dual of its dual problem is again the primal problem.*

Proof. The dual problem as given by (2) is

$$\left.\begin{aligned} &\text{Minimize} \quad z' = \mathbf{b}^{\mathrm{T}}\mathbf{w} \\ &\text{subject to} \\ &\qquad \mathbf{A}^{\mathrm{T}}\mathbf{w} \geq \mathbf{c} \\ &\qquad \mathbf{w} \geq \mathbf{0}. \end{aligned}\right\} \tag{3}$$

We can rewrite (3) as

$$\left.\begin{aligned} &\text{Maximize} \quad z' = -\mathbf{b}^{\mathrm{T}}\mathbf{w} \\ &\text{subject to} \\ &\qquad -\mathbf{A}^{\mathrm{T}}\mathbf{w} \leq -\mathbf{c} \\ &\qquad \mathbf{w} \geq \mathbf{0}. \end{aligned}\right\} \tag{4}$$

Now the dual problem to (4) is

$$\begin{aligned} &\text{Minimize} \quad z = -\mathbf{c}^{\mathrm{T}}\mathbf{x} \\ &\text{subject to} \\ &\qquad (-\mathbf{A}^{\mathrm{T}})^{\mathrm{T}}\mathbf{x} \geq -\mathbf{b} \\ &\qquad \mathbf{x} \geq \mathbf{0}. \end{aligned}$$

This problem can be rewritten as

$$\begin{aligned} &\text{Maximize} \quad z = \mathbf{c}^{\mathrm{T}}\mathbf{x} \\ &\text{subject to} \\ &\qquad \mathbf{A}\mathbf{x} \leq \mathbf{b} \\ &\qquad \mathbf{x} \geq \mathbf{0}, \end{aligned}$$

which is the primal problem. △

THEOREM 3.2. *The linear programming problem in canonical form given by*

$$\begin{aligned} &\text{Maximize} \quad z = \mathbf{c}^{\mathrm{T}}\mathbf{x} \\ &\text{subject to} \\ &\qquad \mathbf{A}\mathbf{x} = \mathbf{b} \\ &\qquad \mathbf{x} \geq \mathbf{0} \end{aligned}$$

has for its dual the linear programming problem

$$\begin{aligned} &\text{Minimize} \quad z' = \mathbf{b}^T\mathbf{w} \\ &\text{subject to} \\ &\mathbf{A}^T\mathbf{w} \geq \mathbf{c} \\ &\mathbf{w} \quad \text{unrestricted.} \end{aligned}$$

Proof. The primal problem can be written as

$$\begin{aligned} &\text{Maximize} \quad z = \mathbf{c}^T\mathbf{x} \\ &\text{subject to} \\ &\mathbf{A}\mathbf{x} \leq \mathbf{b} \\ &-\mathbf{A}\mathbf{x} \leq -\mathbf{b} \\ &\mathbf{x} \geq 0 \end{aligned}$$

by using the method of converting equalities that we described in Section 1.1. In matrix form the primal problem is

$$\begin{aligned} &\text{Maximize} \quad z = \mathbf{c}^T\mathbf{x} \\ &\text{subject to} \\ &\begin{bmatrix} \mathbf{A} \\ -\mathbf{A} \end{bmatrix}\mathbf{x} \leq \begin{bmatrix} \mathbf{b} \\ -\mathbf{b} \end{bmatrix} \\ &\mathbf{x} \geq \mathbf{0}. \end{aligned}$$

The dual problem is then

$$\begin{aligned} &\text{Minimize} \quad z' = \begin{bmatrix} \mathbf{b}^T & -\mathbf{b}^T \end{bmatrix}\begin{bmatrix} \mathbf{u} \\ \mathbf{v} \end{bmatrix} \\ &\text{subject to} \\ &\begin{bmatrix} \mathbf{A}^T & -\mathbf{A}^T \end{bmatrix}\begin{bmatrix} \mathbf{u} \\ \mathbf{v} \end{bmatrix} \geq \mathbf{c} \\ &\mathbf{u} \geq \mathbf{0}, \quad \mathbf{v} \geq \mathbf{0}. \end{aligned}$$

When we multiply out the matrices, we have

$$\begin{aligned} &\text{Minimize} \quad z' = \mathbf{b}^T\mathbf{u} - \mathbf{b}^T\mathbf{v} = \mathbf{b}^T(\mathbf{u} - \mathbf{v}) \\ &\text{subject to} \\ &\mathbf{A}^T\mathbf{u} - \mathbf{A}^T\mathbf{v} = \mathbf{A}^T(\mathbf{u} - \mathbf{v}) \geq \mathbf{c} \\ &\mathbf{u} \geq \mathbf{0}, \quad \mathbf{v} \geq \mathbf{0}. \end{aligned}$$

If we let $\mathbf{w} = \mathbf{u} - \mathbf{v}$, then the dual problem has the form

$$\left.\begin{array}{l}\text{Minimize} \quad z' = \mathbf{b}^T\mathbf{w} \\ \text{subject to} \\ \qquad \mathbf{A}^T\mathbf{w} \geq \mathbf{c} \\ \qquad \mathbf{w} \text{ unrestricted}\end{array}\right\} \tag{5}$$

because any vector may be written as the difference of two nonnegative vectors. △

THEOREM 3.3. *The linear programming problem*

$$\left.\begin{array}{l}\text{Maximize} \quad z = \mathbf{c}^T\mathbf{x} \\ \text{subject to} \\ \qquad \mathbf{A}\mathbf{x} \leq \mathbf{b} \\ \qquad \mathbf{x} \text{ unrestricted},\end{array}\right\} \tag{6}$$

has as its dual problem,

$$\begin{array}{l}\text{Minimize} \quad z' = \mathbf{b}^T\mathbf{w} \\ \text{subject to} \\ \qquad \mathbf{A}^T\mathbf{w} = \mathbf{c} \\ \qquad \mathbf{w} \geq \mathbf{0}.\end{array}$$

Proof. We can rewrite the given problem as

$$\begin{array}{l}\text{Minimize} \quad z = -\mathbf{c}^T\mathbf{x} \\ \text{subject to} \\ \qquad -\mathbf{A}\mathbf{x} \geq -\mathbf{b} \\ \qquad \mathbf{x} \text{ unrestricted}.\end{array}$$

Comparing this statement of the problem with (5), we see that it is the dual of

$$\begin{array}{l}\text{Maximize} \quad z' = -\mathbf{b}^T\mathbf{w} \\ \text{subject to} \\ \qquad -\mathbf{A}^T\mathbf{w} = -\mathbf{c} \\ \qquad \mathbf{w} \geq \mathbf{0}.\end{array}$$

This last problem statement can be written as

$$\left.\begin{array}{l}\text{Minimize} \quad z' = \mathbf{b}^T\mathbf{w} \\ \text{subject to} \\ \qquad \mathbf{A}^T\mathbf{w} = \mathbf{c} \\ \qquad \mathbf{w} \geq \mathbf{0}.\end{array}\right\} \tag{7}$$

We have shown that the dual of problem (7) is problem (6). Therefore, the dual of the dual of problem (7) is the dual of problem (6). Applying

TABLE 3.1

Primal problem	*Dual problem*
Maximization	Minimization
Coefficients of objective function	Right-hand sides of constraints
Coefficients of ith constraint	Coefficients of ith variable, one in each constraint
ith constraint is an inequality $\leq$	ith variable is ≥ 0
ith constraint is an equality	ith variable is unrestricted
jth variable is unrestricted	jth constraint is an equality
jth variable is ≥ 0	jth constraint is an inequality $\geq$
Number of variables	Number of constraints

Theorem 3.1 it follows that problem (7) is the dual of problem (6), as we were to show. △

We summarize the relationships between the primal and dual problems in Table 3.1. Remember that Theorem 3.1 allows us to also read the table from right to left. That is, if the headings "Primal problem" and "Dual problem" are interchanged, the resulting table remains valid. For example, Table 3.1 shows that if the jth constraint in a minimization problem is a $\geq$ inequality, then the jth variable of the dual problem is ≥ 0. Note that the table shows how to find the dual of a maximization problem with $\leq$ and $=$ constraints and of a minimization problem with $\geq$ and $=$ constraints. If we have a maximization problem with a $\geq$ constraint, this constraint must be converted to $\leq$ (by multiplying by -1) before the dual problem can be constructed. The same procedure must be used on a minimization problem with a $\leq$ constraint.

EXAMPLE 2. If the primal problem is

$$\begin{aligned} &\text{Maximize} \quad z = 3x_1 + 2x_2 + x_3 \\ &\text{subject to} \\ &\qquad x_1 + 2x_2 - x_3 \leq 4 \\ &\qquad 2x_1 - x_2 + x_3 = 8 \\ &\qquad x_1 - x_2 \leq 6 \\ &x_1 \geq 0, \quad x_2 \geq 0, \quad x_3 \text{ unrestricted,} \end{aligned}$$

then the dual problem is

$$\begin{aligned} &\text{Minimize} \quad z' = 4w_1 + 8w_2 + 6w_3 \\ &\text{subject to} \\ &\qquad w_1 + 2w_2 + w_3 \geq 3 \\ &\qquad 2w_1 - w_2 - w_3 \geq 2 \\ &\qquad -w_1 + w_2 = 1 \\ &w_1 \geq 0, \quad w_3 \geq 0, \quad w_2 \text{ unrestricted.} \end{aligned}$$

△

EXAMPLE 3. If the primal problem is

$$\begin{aligned}
&\text{Minimize} \quad z = 2x_1 - 3x_2 + x_4 \\
&\text{subject to} \\
&\quad x_1 + 2x_2 + x_3 \leq 7 \\
&\quad x_1 + 4x_2 - x_4 = 5 \\
&\quad x_2 + x_3 + 5x_4 \geq 3 \\
&x_1 \geq 0, \quad x_2 \geq 0, \quad x_3 \geq 0, \quad x_4 \geq 0,
\end{aligned}$$

then the dual problem is

$$\begin{aligned}
&\text{Maximize} \quad z' = -7w_1 + 5w_2 + 3w_3 \\
&\text{subject to} \\
&\quad -w_1 + w_2 \leq 2 \\
&\quad -2w_1 + 4w_2 + w_3 \leq -3 \\
&\quad -w_1 + w_3 \leq 0 \\
&\quad -w_2 + 5w_3 \leq 1 \\
&w_1 \geq 0, \quad w_3 \geq 0, \quad w_2 \text{ unrestricted.}
\end{aligned}$$

To find this dual problem, we write the first constraint of the primal problem as

$$-x_1 - 2x_2 - x_3 \geq -7.$$

An alternate method would be to change the primal problem to a maximization problem and multiply the third constraint by -1. △

Economic Interpretation of the Dual Problem

The dual problem can have various economic interpretations depending upon the point of view one takes. We shall describe two possible interpretations here. As we discover more relationships between the primal and dual problems, we will be able to present some additional interpretations.

In Example 1 of Section 1.1 we have a model of a sawmill. It is

$$\begin{aligned}
&\text{Maximize} \quad z = 120x_1 + 100x_2 \\
&\text{subject to} \\
&\quad 2x_1 + 2x_2 \leq 8 \\
&\quad 5x_1 + 3x_2 \leq 15 \\
&x_1 \geq 0, \quad x_2 \geq 0.
\end{aligned}$$

This is a simple example of a typical manufacturing problem that is in standard form (1). The first constraint in the sawmill problem deals with

the number of hours for which the saw is available. The second constraint deals with the number of hours for which the plane is available. In general, in the ith constraint of (1),

$$\sum_{j=1}^{n} a_{ij}x_j \leq b_i,$$

we may think of b_i as the total supply of the ith resource, or raw material, or input. For the first constraint of the sawmill problem it was sawing time. The coefficient a_{ij} in the general problem represents the amount of the ith input required per unit of the jth product, or output. For example, $a_{21} = 5$ in the sawmill example represents the 5 hr of planing time required for each 1000 board feet of finish-grade lumber. The variable x_j is the unknown amount of the jth output that is to be produced. The coefficient c_j in the objective function represents the profit, or value, derived from one unit of the jth output. The optimal solution $x_1 = \frac{3}{2}$, $x_2 = \frac{5}{2}$ maximizes the total value of all outputs

$$z = \sum_{j=1}^{n} c_j x_j.$$

The dual problem to the sawmill example is

$$\begin{aligned} &\text{Minimize} \quad z' = 8w_1 + 15w_2 \\ &\text{subject to} \\ &\qquad 2w_1 + 5w_2 \geq 120 \\ &\qquad 2w_1 + 3w_2 \geq 100 \\ &\qquad w_1 \geq 0, \quad w_2 \geq 0. \end{aligned}$$

The coefficients of the first constraint are the amounts of each input that are needed to make one unit (1000 board feet) of the first output. That is, to make 1000 board feet of finish-grade lumber we need 2 hr of sawing and 5 hr of planing. The right-hand side of the first constraint is the profit, or value of one unit of the first output. Likewise, the second constraint of the dual problem of the sawmill example says that to make 1000 board feet of construction-grade lumber we need 2 hr of sawing and 3 hr of planing, and the value of this amount of lumber is \$100. Solving the dual problem we discover (verify) that $w_1 = 35$ and $w_2 = 10$.

The dual problem of the general linear programming problem in standard form (1) is

$$\begin{aligned} &\text{Minimize} \quad z' = \mathbf{b}^{\mathrm{T}}\mathbf{w} \\ &\text{subject to} \\ &\qquad \mathbf{A}^{\mathrm{T}}\mathbf{w} \geq \mathbf{c} \\ &\qquad \mathbf{w} \geq \mathbf{0}. \end{aligned}$$

The jth constraint of this problem is

$$\sum_{i=1}^{m} a_{ij} w_i \geq c_j.$$

As above, the coefficient a_{ij} represents the amount of input i per unit of output j, and the right-hand side is the value per unit of output j. This means that the units of the dual variable w_i are "value per unit of input i." The dual variables act as prices, or costs, or values of one unit of each of the inputs. They are called by several names, including **accounting prices, fictitious prices, shadow prices**, and **imputed values**.

At an optimal solution to the primal problem, profit, which is equal to $\mathbf{c}^T\mathbf{x}$, is also equal to $\mathbf{b}^T\mathbf{w}$, as we will show in Section 3.2. Thus, increasing the ith input b_i by one unit increases $\mathbf{b}^T\mathbf{w}$, and hence profit, by w_i units. Hence, at an optimal solution to the dual problem, w_i represents the contribution to profit of one unit of the ith input. The values of the dual variables are not directly related to the actual costs of the inputs. Just because the optimal solution to the dual of the sawmill problem is $w_1 = 35$, $w_2 = 10$ does not mean that the cost of the saw is \$35 per hour and the cost of the plane is \$10 per hour. The actual costs are hidden in whatever computations were done to figure out that the profits were \$120 per 1000 board feet of finish-grade and \$100 per board feet of construction-grade lumber. In the sense that the dual variables do not represent actual costs, they are fictitious prices.

The left-hand side of the jth constraint of the dual problem gives the total value of the inputs used in making one unit of the jth output. This constraint says that this value must be at least as much as the profit of one unit of the jth output. But at an optimal solution the value of the left-hand side represents the total contribution to profit of one unit of the jth output, and it is reasonable to expect to operate when this contribution to profit is at least as much as the actual profit. If this were not the case, the manufacturer would be well advised to use the available inputs in a better way.

We see then that the dual problem seeks shadow prices for each of the inputs that minimize their total price, subject to the restriction that these prices, or values, yield a corresponding value for each unit of output that is at least the profit for a unit of output.

Another description of the dual variables comes from the fact that, as we will show in Section 3.2, at an optimal solution to the dual problem,

$$\text{Profit} = \mathbf{b}^T\mathbf{w}.$$

To increase this profit the manufacturer must increase the availability of at least one of the resources. If b_i is increased by one unit, the profit will increase by w_i. Thus, w_i represents the **marginal value** of the ith input. In

the same way, w_i is the loss incurred if one unit of the ith resource is not used. Thus it can be considered as a **replacement value** of the ith resource for insurance purposes. In fact, an insurance company would want to use the dual problem in case of a claim for lost resources; it wants to pay out as little as possible to settle the claim.

It is also interesting to look at an interpretation of the dual of the diet problem that was given in Example 2 of Section 1.1. The model of the diet problem is

$$\begin{aligned} &\text{Minimize} \quad z = 20x_1 + 25x_2 \\ &\text{subject to} \\ &\qquad 2x_1 + 3x_2 \geq 18 \\ &\qquad x_1 + 3x_2 \geq 12 \\ &\qquad 4x_1 + 3x_2 \geq 24 \\ &\qquad x_1 \geq 0, \quad x_2 \geq 0. \end{aligned}$$

The dual problem is

$$\begin{aligned} &\text{Maximize} \quad z' = 18w_1 + 12w_2 + 24w_3 \\ &\text{subject to} \\ &\qquad 2w_1 + w_2 + 4w_3 \leq 20 \\ &\qquad 3w_1 + 3w_2 + 3w_3 \leq 25 \\ &\qquad w_1 \geq 0, \quad w_2 \geq 0, \quad w_3 \geq 0. \end{aligned}$$

To discuss the dual problem we introduce some notation. Let N_1, N_2, and N_3 denote the nutrients fat, carbohydrates, and protein, respectively. It is also convenient to denote foods A and B by F_1 and F_2, respectively. Now introduce a manufacturer that makes artificial foods P_1, P_2, and P_3 with the property that for each $i = 1$, 2, or 3, one unit of P_i provides one unit of nutrient N_i. Assume that the manufacturer sets w_i as the price of P_i ($i = 1$, 2, or 3). Recall that in the original statement of the problem, a_{ij} is the number of units of nutrient N_i in 1 oz of food F_j. For example, $a_{12} = 3$ is the number of units of fat (N_1) in 1 oz of food F_2, and $a_{31} = 4$ is the number of units of protein (N_3) in 1 oz of food F_1. The artificial food manufacturer will set its prices so that

$$2w_1 + w_2 + 4w_3 \leq 20$$

and

$$3w_1 + 3w_2 + 3w_3 \leq 25.$$

That is, it will set the prices on the foods P_i so that when these foods are taken in the proportions necessary to give the same nutrition as foods F_1 and F_2, the cost of the substitute for F_1 is no greater than the cost of F_1 itself and the cost of the substitute for F_2 is no greater than the cost of F_2

itself. Thus, the nutritionist will always find it at least as economical to buy the three artificial foods. Since we require 18 units of fat (from P_1), 12 units of carbohydrate (from P_2), and 24 units of protein (from P_3), the manufacturer's revenue is

$$z' = 18w_1 + 12w_2 + 24w_3.$$

It seeks to maximize this revenue.

Thus, the fictitious prices w_1, w_2, and w_3 of the nutrients are those prices that the artificial food manufacturer should charge to maximize the revenue, yet still be able to compete with the producer of foods F_1 and F_2. Therefore, these fictitious prices represent competitive prices.

3.1 EXERCISES

In Exercises 1–6 find the dual of the given linear programming problem.

1. Minimize $z = 3x_1 + 4x_2$
   subject to
   $$\begin{aligned} x_1 + 4x_2 &\geq 8 \\ 2x_1 + 3x_2 &\geq 12 \\ 2x_1 + x_2 &\geq 6 \\ x_1 \geq 0, \quad x_2 &\geq 0. \end{aligned}$$

2. Minimize $z = 6x_1 + 6x_2 + 8x_3 + 9x_4$
   subject to
   $$\begin{aligned} x_1 + 2x_2 + x_3 + x_4 &\geq 3 \\ 2x_1 + x_2 + 4x_3 + 9x_4 &\geq 8 \\ x_1 \geq 0, \quad x_2 \geq 0, \quad x_3 \geq 0, \quad x_4 &\geq 0. \end{aligned}$$

3. Maximize $z = 3x_1 + 2x_2 + 5x_3 + 7x_4$
   subject to
   $$\begin{aligned} 3x_1 + 2x_2 + x_3 &\leq 8 \\ 5x_1 + x_2 + 2x_3 + 4x_4 &= 7 \\ 4x_1 + x_3 - 2x_4 &\leq 12 \\ x_1 \geq 0, \quad x_2 \geq 0, \quad x_3 \geq 0, \quad x_4 &\geq 0. \end{aligned}$$

4. Maximize $z = 2x_1 + x_2 + 3x_3 + 4x_4$
   subject to
   $$\begin{aligned} 4x_1 + 2x_2 + 5x_3 + 5x_4 &\leq 10 \\ 4x_1 + 2x_2 + 5x_3 + 5x_4 &\geq 5 \\ 3x_1 + 5x_2 + 4x_3 + x_4 &\geq 8 \\ 3x_1 + 5x_2 + 4x_3 + x_4 &\leq 15 \\ x_1 + x_2 + x_3 + x_4 &= 20 \\ x_1 \geq 0, \quad x_2 \geq 0, \quad x_3 \geq 0, \quad x_4 &\geq 0. \end{aligned}$$

5. Maximize $z = 3x_1 + x_2 + 4x_3$
subject to
$$3x_1 + 3x_2 + x_3 \leq 18$$
$$2x_1 + 2x_2 + 4x_3 = 12$$
$$x_1 \geq 0, \quad x_3 \geq 0.$$

6. Minimize $z = 5x_1 + 2x_2 + 6x_3$
subject to
$$4x_1 + 2x_2 + x_3 \geq 12$$
$$3x_1 + 2x_2 + 3x_3 \leq 6$$
$$x_1 \geq 0, \quad x_2 \geq 0.$$

7. The text suggested an alternate method for solving Example 3.
   (a) Use this method to construct the dual problem.
   (b) Verify that the resulting problem is the same as the one given in the text. (*Hint:* Use the fact that w_2 is unrestricted.)

In Exercises 8–11 formulate the dual problem for the given linear programming problem.

8. Exercise 2, Section 1.1
9. Exercise 4, Section 1.1. Give an economic interpretation of the dual problem.
10. Exercise 9, Section 1.1. Give an economic interpretation of the dual problem.
11. Using the definition of the dual of a problem in standard form, find the dual of the linear programming problem

Maximize $z = \mathbf{c}^T\mathbf{x} + \mathbf{d}^T\mathbf{x}'$
subject to
$$\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{x}' \leq \mathbf{b}$$
$$\mathbf{x} \geq \mathbf{0}, \quad \mathbf{x}' \text{ unrestricted}$$

(*Hint:* Write $\mathbf{x}' = \mathbf{u} - \mathbf{v}$, $\mathbf{u} \geq \mathbf{0}$, $\mathbf{v} \geq \mathbf{0}$, and express the given problem as a standard linear programming problem in matrix notation.)

3.2 THE DUALITY THEOREM

In his work on game theory, John von Neumann, one of the most versatile mathematicians of all time, proved a duality theorem for games. In October of 1947 George Dantzig, one of the pioneers of linear programming and the developer of the simplex algorithm, went to see von Neumann in Princeton. After hearing the basic ideas in linear programming, von Neumann indicated to Dantzig his Duality Theorem for games and also conjectured and proved an equivalent result for linear programming. However, this proof was not published, and the first careful proof of

the Duality Theorem, now recognized as the fundamental theorem of linear programming, was published in 1950 by A. W. Tucker and his students David Gale and Harold W. Kuhn. The Duality Theorem establishes conditions for a feasible solution to a linear programming problem to be an optimal solution.

To present and prove the Duality Theorem we first need to develop some tools for expressing the solution to a linear programming problem in terms of the entries in any tableau that has been constructed while using the simplex method to solve the problem.

Suppose that we are given a general linear programming problem and suppose that we have introduced slack and artificial variables to convert the problem to canonical form. That is, we take our problem as

$$\left.\begin{aligned} &\text{Maximize} \quad z = \mathbf{c}^{\mathrm{T}}\mathbf{x} \\ &\text{subject to} \\ &\qquad \mathbf{A}\mathbf{x} = \mathbf{b} \\ &\qquad \mathbf{x} \geq \mathbf{0}, \end{aligned}\right\} \tag{1}$$

where $\mathbf{A}$ is an $m \times s$ matrix that contains an $m \times m$ identity submatrix; $\mathbf{b} \geq \mathbf{0}$ is an $m \times 1$ matrix; and $\mathbf{c}$ is an $s \times 1$ matrix. Remember that if x_j is a slack variable, then $c_j = 0$. Also, we use the two-phase method for the artificial variables so that, if x_j is an artificial variable, then $c_j = 0$.

We now examine a tableau constructed by the simplex algorithm during the solution of our problem. This tableau represents a basic feasible solution. Let i_1 be the subscript of the first basic variable in this solution; let i_2 be the subscript of the second basic variable. Continuing in this manner, we end with i_m being the subscript of the mth basic variable. Let N denote the set of indices of the nonbasic variables. We also let $\mathbf{A}_j$ denote the jth column of $\mathbf{A}$. Using this notation, we may write the second equality of (1) as

$$\sum_{r=1}^{m} x_{i_r}\mathbf{A}_{i_r} + \sum_{j \in N} x_j\mathbf{A}_j = \mathbf{b}. \tag{2}$$

Recall that the nonbasic variables are set equal to zero, so that (2) may be simplified to

$$x_{i_1}\mathbf{A}_{i_1} + x_{i_2}\mathbf{A}_{i_2} + \cdots + x_{i_m}\mathbf{A}_{i_m} = \mathbf{b}. \tag{3}$$

We make an $m \times m$ matrix out of the m columns $\mathbf{A}_{i_1}, \mathbf{A}_{i_2}, \ldots, \mathbf{A}_{i_m}$ of $\mathbf{A}$ corresponding to basic variables and denote this matrix by $\mathbf{B}$. We introduce notation for the basic feasible solution expressed as a vector and the

corresponding cost vector by letting

$$\mathbf{x}_B = \begin{bmatrix} x_{i_1} \\ x_{i_2} \\ \vdots \\ x_{i_m} \end{bmatrix} \quad \text{and} \quad \mathbf{c}_B = \begin{bmatrix} c_{i_1} \\ c_{i_2} \\ \vdots \\ c_{i_m} \end{bmatrix}.$$

Then (3) may be written as

$$\mathbf{B}\mathbf{x}_B = \mathbf{b}. \tag{4}$$

Using Theorem 0.9, the columns of $\mathbf{B}$ are linearly independent, so $\mathbf{B}$ must be a nonsingular matrix. We may write (4) as

$$\mathbf{x}_B = \mathbf{B}^{-1}\mathbf{b}. \tag{5}$$

That is, the m-tuple of basic variables has the value $\mathbf{B}^{-1}\mathbf{b}$ in any tableau.

We also note that the m columns of $\mathbf{B}$ considered as m-tuples form a basis for R^m. Thus, the jth column of our initial tableau, $\mathbf{A}_j$, $j = 1, 2, \ldots, s$, can be written as a linear combination of the columns $\mathbf{A}_{i_1}, \mathbf{A}_{i_2}, \ldots, \mathbf{A}_{i_m}$, where the indices $i_1, i_2, \ldots, i_m$ are the indices of the basic variables of our current tableau in the order of the labels on the left-hand side.

We have

$$\mathbf{A}_j = t_{1j}\mathbf{A}_{i_1} + t_{2j}\mathbf{A}_{i_2} + \cdots + t_{mj}\mathbf{A}_{i_m}, \quad j = 1, 2, \ldots, s. \tag{6}$$

It can be shown that the vector of coefficients in (6),

$$\mathbf{t}_j = \begin{bmatrix} t_{1j} \\ t_{2j} \\ \vdots \\ t_{mj} \end{bmatrix},$$

which is the coordinate vector of $\mathbf{A}_j$ with respect to the basis

$$\{\mathbf{A}_{i_1}, \mathbf{A}_{i_2}, \ldots, \mathbf{A}_{i_m}\}$$

of R^m, is also the jth column of our current tableau. In the same way as in (4), we may write (6) as

$$\mathbf{B}\mathbf{t}_j = \mathbf{A}_j$$

or

$$\mathbf{t}_j = \mathbf{B}^{-1}\mathbf{A}_j. \tag{7}$$

Equation (7) says that any column of a tableau can be found from the corresponding column of the initial tableau by multiplying on the left by $\mathbf{B}^{-1}$.

Using the notation that has been developed, we define

$$z_j = \mathbf{c}_B^T \mathbf{t}_j \tag{8}$$

or, using (7),

$$z_j = \mathbf{c}_B^T \mathbf{B}^{-1} \mathbf{A}_j. \tag{9}$$

From this definition we see that for the rth basic variable x_{i_r} we have

$$z_{i_r} = \begin{bmatrix} c_{i_1} & c_{i_2} & \cdots & c_{i_m} \end{bmatrix} \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \leftarrow r\text{th entry}$$

$$= c_{i_r}.$$

The objective function for this problem takes on a particularly simple form when we use the definition of z_j. Recall that

$$z = \mathbf{c}^T \mathbf{x} = \sum_{j=1}^{s} c_j x_j. \tag{10}$$

We separate the sum on the right-hand side of (10) into two parts: one part contains the basic variables and the other part contains the nonbasic variables. We write

$$z = \sum_{r=1}^{m} c_{i_r} x_{i_r} + \sum_{j \in N} c_j x_j. \tag{11}$$

In order to apply the optimality criterion to our tableau, we must modify (11) so that the coefficients of the basic variables are zero. To make this modification we add $(-c_{i_1}) \times$ first row of the current tableau $+(-c_{i_2}) \times$ second row $+ \cdots +(-c_{i_m}) \times m$th row to (11). In symbols, the rth row of the current tableau can be expressed as

$$\sum_{j=1}^{s} t_{rj} x_j = \mathbf{x}_{Br},$$

where $\mathbf{x}_{Br}$ is the rth component of the vector $\mathbf{x}_B$.

Adding the appropriate multiples of the rows of the tableau to (11), we obtain

$$z - \sum_{r=1}^{m} c_{i_r} \mathbf{x}_{Br} = \sum_{j \in N} c_j x_j - \sum_{r=1}^{m} c_{i_r} \sum_{j \in N} t_{r_j} x_j.$$

Simplifying, we have

$$z - \mathbf{c}_B^T \mathbf{x}_B = \sum_{j \in N} (c_j - \mathbf{c}_B^T \mathbf{t}_j) x_j$$

or by (8)

$$\sum_{j \in N} (z_j - c_j) x_j + z = \mathbf{c}_B^T \mathbf{x}_B.$$

Since $z_{i_r} - c_{i_r} = 0$, we obtain

$$\sum_{j=1}^{s} (z_j - c_j) x_j + z = \mathbf{c}_B^T \mathbf{x}_B.$$

This equation shows that the entries in the objective row of a tableau are simply

$$z_j - c_j = \mathbf{c}_B^T \mathbf{t}_j - c_j. \tag{12}$$

We may restate the optimality criterion for the simplex method: a tableau represents an optimal solution if and only if, for all j, $z_j - c_j \geq 0$.

EXAMPLE 1. Consider the linear programming problem in canonical form from Section 2.1, Example 2.

$$\begin{aligned}
&\text{Maximize} \quad z = 8x_1 + 9x_2 + 5x_3 \\
&\text{subject to} \\
&x_1 + x_2 + 2x_3 + x_4 = 2 \\
&2x_1 + 3x_2 + 4x_3 + x_5 = 3 \\
&6x_1 + 6x_2 + 2x_3 + x_6 = 8 \\
&x_j \geq 0, \quad j = 1, 2, \ldots, 6,
\end{aligned}$$

where

$$\mathbf{A} = \begin{bmatrix} 1 & 1 & 2 & 1 & 0 & 0 \\ 2 & 3 & 4 & 0 & 1 & 0 \\ 6 & 6 & 2 & 0 & 0 & 1 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} 2 \\ 3 \\ 8 \end{bmatrix}, \quad \text{and} \quad \mathbf{c} = \begin{bmatrix} 8 \\ 9 \\ 5 \\ 0 \\ 0 \\ 0 \end{bmatrix}.$$

After the first iteration of the simplex method we obtained Tableau 3.2 as a tableau representing the basic feasible solution in which $i_1 = 4$, $i_2 = 2$, and $i_3 = 6$, and thus

$$\mathbf{x}_B = \begin{bmatrix} x_4 \\ x_2 \\ x_6 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 2 \end{bmatrix}.$$

Tableau 3.1

	x_1	x_2	x_3	x_4	x_5	x_6	
x_4	$\frac{1}{3}$	0	$\frac{2}{3}$	1	$-\frac{1}{3}$	0	1
x_2	$\frac{2}{3}$	1	$\frac{4}{3}$	0	$\frac{1}{3}$	0	1
x_6	2	0	−6	0	−2	1	2
	−2	0	7	0	3	0	9

We also have $\mathbf{c}_B^T = [c_4 \quad c_2 \quad c_6] = [0 \quad 9 \quad 0]$. The problem in matrix form

$$\mathbf{Ax} = \mathbf{b}$$

can be written as

$$\mathbf{A}_1 x_1 + \mathbf{A}_2 x_2 + \cdots + \mathbf{A}_6 x_6 = \mathbf{b}$$

or

$$\begin{bmatrix} 1 \\ 2 \\ 6 \end{bmatrix} x_1 + \begin{bmatrix} 1 \\ 3 \\ 6 \end{bmatrix} x_2 + \cdots + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} x_6 = \begin{bmatrix} 2 \\ 3 \\ 8 \end{bmatrix}.$$

Since the tableau represents x_4, x_2, and x_6 as basic variables in this order, we may rearrange the previous equation as

$$\begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} x_4 + \begin{bmatrix} 1 \\ 3 \\ 6 \end{bmatrix} x_2 + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} x_6 + \begin{bmatrix} 1 \\ 2 \\ 6 \end{bmatrix} x_1 + \begin{bmatrix} 2 \\ 4 \\ 2 \end{bmatrix} x_3 + \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} x_5 = \begin{bmatrix} 2 \\ 3 \\ 8 \end{bmatrix},$$

the entries being the coefficients of the basic variables in the objective function. Thus,

$$\mathbf{B} = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 3 & 0 \\ 0 & 6 & 1 \end{bmatrix}$$

and

$$\mathbf{B}^{-1} = \begin{bmatrix} 1 & -\frac{1}{3} & 0 \\ 0 & \frac{1}{3} & 0 \\ 0 & -2 & 1 \end{bmatrix}.$$

Then

$$\mathbf{x}_B = \mathbf{B}^{-1}\mathbf{b} = \begin{bmatrix} 1 & -\frac{1}{3} & 0 \\ 0 & \frac{1}{3} & 0 \\ 0 & -2 & 1 \end{bmatrix} \begin{bmatrix} 2 \\ 3 \\ 8 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 2 \end{bmatrix},$$

which agrees with the right-hand side of the tableau. Furthermore, $\mathbf{t}_1 = \mathbf{B}^{-1}\mathbf{A}_1$ or

$$\begin{bmatrix} \frac{1}{3} \\ \frac{2}{3} \\ 2 \end{bmatrix} = \begin{bmatrix} 1 & -\frac{1}{3} & 0 \\ 0 & \frac{1}{3} & 0 \\ 0 & -2 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 6 \end{bmatrix}.$$

The rest of the columns can be checked in the same manner. Finally, the entries in the objective row are $z_j - c_j$, where $z_j = \mathbf{c}_B^T \mathbf{t}_j$. We have

$$z_1 = [0 \quad 9 \quad 0]\begin{bmatrix} \frac{1}{3} \\ \frac{2}{3} \\ 2 \end{bmatrix} = 6,$$

and the first entry in the objective row is

$$z_1 - c_1 = 6 - 8 = -2.$$

The other entries can be checked in the same manner. △

Relations between the Solutions to the Primal and Dual Problems

We saw in Chapter 2 that there are three possible outcomes when attempting to solve a linear programming problem.

1. No feasible solutions exist.
2. There is a finite optimal solution.
3. Feasible solutions exist, but the objective function is unbounded.

Since the dual problem is a linear programming problem, attempting to solve it also leads to these three possible outcomes. Consequently in considering relationships between solutions to the primal and dual problems there are nine alternatives for the pair of solutions. We now present theorems that show which of these alternatives actually can occur.

THEOREM 3.4 (WEAK DUALITY THEOREM). *If* $\mathbf{x}_0$ *is a feasible solution to the primal problem*

$$\left.\begin{array}{l} \text{Maximize} \quad z = \mathbf{c}^T\mathbf{x} \\ \text{subject to} \\ \qquad \mathbf{A}\mathbf{x} \leq \mathbf{b} \\ \qquad \mathbf{x} \geq \mathbf{0}, \end{array}\right\} \tag{13}$$

and if $\mathbf{w}_0$ *is a feasible solution to the dual problem*

$$\left.\begin{array}{l} \text{Minimize} \quad z' = \mathbf{b}^T\mathbf{w} \\ \text{subject to} \\ \qquad \mathbf{A}^T\mathbf{w} \geq \mathbf{c} \\ \qquad \mathbf{w} \geq \mathbf{0}, \end{array}\right\} \tag{14}$$

then

$$\mathbf{c}^T\mathbf{x}_0 \leq \mathbf{b}^T\mathbf{w}_0 \tag{15}$$

That is, the value of the objective function of the dual problem is always greater than or equal to the value of the objective function of the primal problem.

Proof. Since $\mathbf{x}_0$ is a feasible solution to (13), we have

$$\mathbf{A}\mathbf{x}_0 \leq \mathbf{b}. \tag{16}$$

It follows from (16) that

$$\mathbf{w}_0^T\mathbf{A}\mathbf{x}_0 \leq \mathbf{w}_0^T\mathbf{b} = \mathbf{b}^T\mathbf{w}_0 \tag{17}$$

since $\mathbf{w}_0 \geq \mathbf{0}$. The equality in (17) comes from the fact that $\mathbf{w}_0^T\mathbf{b}$ is a 1×1 matrix and consequently is equal to its transpose.

Since $\mathbf{w}_0$ is a feasible solution to (14), we have

$$\mathbf{A}^T\mathbf{w}_0 \geq \mathbf{c}$$

or, taking transposes,

$$\mathbf{w}_0^T\mathbf{A} \geq \mathbf{c}^T.$$

Again we can multiply by $\mathbf{x}_0$, which is nonnegative, without changing the inequality. We get

$$\mathbf{w}_0^T\mathbf{A}\mathbf{x}_0 \geq \mathbf{c}^T\mathbf{x}_0. \tag{18}$$

Combining inequalities (17) and (18) gives the desired result. △

An important application of Theorem 3.4 arises in the case in which the primal problem has feasible solutions but the objective function is unbounded. This means that, given any number N, we can find a feasible solution to the primal problem for which the value of the objective function is greater than N. Consequently the objective function of the dual problem (using Theorem 3.4) is greater than N for any feasible solution $\mathbf{w}_0$ to the dual problem. This means that there are no feasible solutions to the dual problem. For if $\mathbf{w}$ were such a solution, the value of the dual objective function would be $\mathbf{b}^T\mathbf{w}$. But N can be chosen greater than this value, and we have

$$\mathbf{b}^T\mathbf{w} < N < \mathbf{b}^T\mathbf{w},$$

the second inequality coming from the argument above. This impossible pair of inequalities means that $\mathbf{w}$ cannot exist. The discussion above has proved part (a) of the following theorem.

THEOREM 3.5. *(a) If the primal problem has feasible solutions but the objective function is unbounded, then the dual problem has no feasible solutions.*

(b) If the dual problem has feasible solutions but the objective function is unbounded, then the primal problem has no feasible solutions.

Proof. (b) Combining Theorem 3.1 and part (a) we obtain the desired result. △

We now give a condition that a feasible solution to the primal problem be an optimal solution.

THEOREM 3.6. *If* $\mathbf{x}_0$ *and* $\mathbf{w}_0$ *are feasible solutions to the primal and dual problems (13) and (14), respectively, and if* $\mathbf{c}^T\mathbf{x}_0 = \mathbf{b}^T\mathbf{w}_0$, *then both* $\mathbf{x}_0$ *and* $\mathbf{w}_0$ *are optimal solutions to their respective problems.*

Proof. Suppose $\mathbf{x}_1$ is any feasible solution to the primal problem. Then, from (15),

$$\mathbf{c}^T\mathbf{x}_1 \leq \mathbf{b}^T\mathbf{w}_0 = \mathbf{c}^T\mathbf{x}_0.$$

Hence, $\mathbf{x}_0$ is an optimal solution. Similarly, if $\mathbf{w}_1$ is any feasible solution to the dual problem, then, from (15),

$$\mathbf{b}^T\mathbf{w}_0 = \mathbf{c}^T\mathbf{x}_0 \leq \mathbf{b}^T\mathbf{w}_1,$$

and we see that $\mathbf{w}_0$ is an optimal solution to the dual problem. △

THEOREM 3.7 (DUALITY THEOREM).

(a) If either the primal or dual problem has a feasible solution with a finite optimal objective value, then the other problem has a feasible solution with the same objective value.

(b) If the primal (13) and dual (14) problems have feasible solutions, then

(i) the primal problem has an optimal solution—say, $\mathbf{x}_0$*;*
(ii) the dual problem has an optimal solution—say, $\mathbf{w}_0$*; and*
(iii) $\mathbf{c}^T\mathbf{x}_0 = \mathbf{b}^T\mathbf{w}_0$.

Proof. (a) We convert the primal problem given in (13) to canonical form by introducing the vector of slack variables $\mathbf{x}'$ and writing,

$$\text{Maximize} \quad z = [\mathbf{c} \mid \mathbf{0}]^T \begin{bmatrix} \mathbf{x} \\ \mathbf{x}' \end{bmatrix}$$

subject to

$$[\mathbf{A} \mid \mathbf{I}] \begin{bmatrix} \mathbf{x} \\ \mathbf{x}' \end{bmatrix} = \mathbf{b}$$

$$\mathbf{x} \geq \mathbf{0}$$
$$\mathbf{x}' \geq \mathbf{0}.$$

Let $\hat{\mathbf{x}}$ be an optimal solution to this problem. Then there is a corresponding invertible matrix $\mathbf{B}$ that gives the values of the basic variables of $\hat{\mathbf{x}}$ as $\hat{\mathbf{x}}_B = \mathbf{B}^{-1}\mathbf{b}$ [see Equation (5)]. The objective function value at the optimal solution is then

$$z = \mathbf{c}_B^T\hat{\mathbf{x}}_B = \mathbf{c}_B^T\mathbf{B}^{-1}\mathbf{b}.$$

We let

$$\mathbf{w} = (\mathbf{B}^{-1})^T\mathbf{c}_B$$

and thus

$$\mathbf{w}^T = \mathbf{c}_B^T\mathbf{B}^{-1},$$

and by substitution we obtain $z = \mathbf{w}^T\mathbf{b}$. Since z is a number,

$$z = \mathbf{w}^T\mathbf{b} = (\mathbf{w}^T\mathbf{b})^T = \mathbf{b}^T\mathbf{w}.$$

Thus, the objective function value for the dual problem, namely, $\mathbf{b}^T\mathbf{w}$, agrees with the objective function value for the primal problem. We now show that $\mathbf{w}$ is a feasible solution to the dual problem given by (14).

From (7) we have

$$\mathbf{t}_j = \mathbf{c}_B^T\mathbf{B}^{-1}[\mathbf{A}\,|\,\mathbf{I}]_j$$

and from (8) we have

$$z_j = \mathbf{c}_B^T\mathbf{t}_j = \mathbf{c}_B^T\mathbf{B}^{-1}[\mathbf{A}\,|\,\mathbf{I}]_j.$$

The optimality criterion of the simplex algorithm implies that

$$z_j - c_j \geq 0,$$

which can be written in vector form as

$$\mathbf{z} - [\mathbf{c}^T\,|\,\mathbf{0}] = \mathbf{c}_B^T\mathbf{B}^{-1}[\mathbf{A}\,|\,\mathbf{I}] - [\mathbf{c}^T\,|\,\mathbf{0}] \geq \mathbf{0}.$$

Multiplying in the previous equation and separating the partitioned matrix, we have

$$\mathbf{c}_B^T\mathbf{B}^{-1}\mathbf{A} \geq \mathbf{c}^T$$

and

$$\mathbf{c}_B^T\mathbf{B}^{-1} \geq \mathbf{0}.$$

Using the definition of $\mathbf{w}$, these inequalities can be written as

$$\mathbf{w}^T\mathbf{A} \geq \mathbf{c}^T$$
$$\mathbf{w} \geq \mathbf{0}$$

or

$$\mathbf{A}^T\mathbf{w} \geq \mathbf{c}$$
$$\mathbf{w} \geq \mathbf{0}.$$

Hence, $\mathbf{w}$ is a feasible solution to the dual problem given by (14) and yields the same value for the objective function of the dual problem as does the optimal solution $\mathbf{x}$ for the objective function of the primal problem.

(b) Let $\mathbf{x}$ be a feasible solution to (13) and $\mathbf{w}$ be a feasible solution to (14). By Theorem 3.4

$$\mathbf{c}^T\mathbf{x} \leq \mathbf{b}^T\mathbf{w}.$$

Since the objective function of the primal problem is bounded by $\mathbf{b}^T\mathbf{w}$ and the set of feasible solutions is not empty, there is a finite optimal solution

$\mathbf{x}_0$. By part (a) there is a feasible solution $\mathbf{w}_0$ to (14) so

$$\mathbf{b}^T\mathbf{w}_0 = \mathbf{c}^T\mathbf{x}_0.$$

Theorem 3.6 then implies that both $\mathbf{x}_0$ and $\mathbf{w}_0$ are optimal. △

The optimality criterion for the simplex method tells us that the jth variable of the primal problem is a candidate for an entering variable if $z_j - c_j < 0$. We have already noted that w_i is a value, or cost, attached to each unit of resource i. Then z_j represents the total value assigned to one unit of the jth product. In economics this value is called an **imputed value** because it is not directly determined by the marketplace; it is a value assigned using some other basis than an exchange of money. The jth variable is a candidate for an entering variable if $z_j < c_j$; that is, x_j may enter if the imputed value z_j of the jth product is less than the profit c_j per unit of the jth product.

We summarize the results of the Weak Duality Theorem and the Duality Theorem in the following table. The column and row labels have the following meanings: none, no feasible solution; finite, optimal solution with finite objective function value; and unbounded, feasible solutions with unbounded objective function value.

TABLE 3.2

	Primal		
Dual	*None*	*Finite*	*Unbounded*
None		Impossible (Theorem 3.7)	
Finite	Impossible (Theorem 3.7)		Impossible (Theorem 3.5)
Unbounded		Impossible (Theorem 3.5)	Impossible (Theorem 3.5)

We next present examples to show that the missing entries in Table 3.2 are all possible.

EXAMPLE 2. Consider the linear programming problem

$$\left.\begin{aligned} &\text{Maximize} \quad z = 2x_1 + x_2 \\ &\text{subject to} \\ &\qquad 3x_1 - 2x_2 \le 6 \\ &\qquad x_1 - 2x_2 \le 1 \\ &\qquad x_1 \ge 0, \quad x_2 \ge 0. \end{aligned}\right\} \tag{19}$$

The set of feasible solutions is shown in Figure 3.1a. It is evident that this problem has an unbounded optimal solution. For setting $x_1 = 0$ allows x_2 to be as large as we please. In this case $z = x_2$ is also as large as we please.

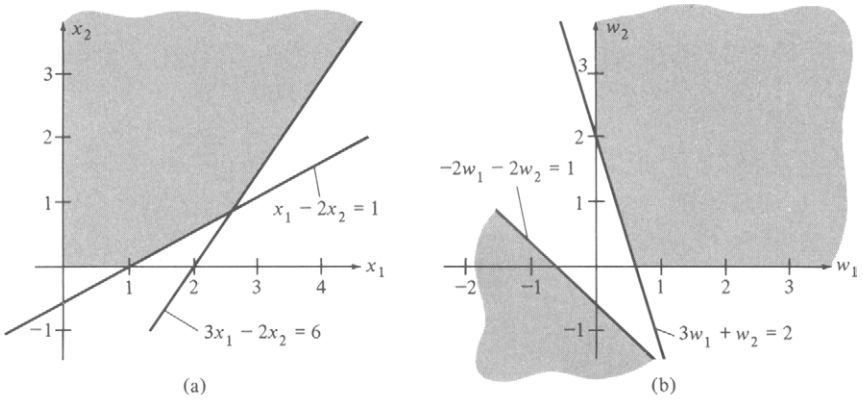


FIGURE 3.1

The dual problem to (19) is

$$\begin{aligned} &\text{Minimize} \quad z' = 6w_1 + w_2 \\ &\text{subject to} \\ &\qquad 3w_1 + \ w_2 \geq 2 \\ &\qquad -2w_1 - 2w_2 \geq 1 \\ &\qquad w_1 \geq 0, \quad w_2 \geq 0. \end{aligned}$$

The constraints are shown in Figure 3.1b. There are no feasible solutions to the problem, since the second constraint can never hold for nonnegative values of w_1 and w_2. △

It is also possible, as the next example shows, that neither the primal problem nor its dual will have a feasible solution.

EXAMPLE 3. Consider the linear programming problem

$$\begin{aligned} &\text{Maximize} \quad z = 3x_1 + 2x_2 \\ &\text{subject to} \\ &\qquad 2x_1 - 2x_2 \leq -1 \\ &\qquad -2x_1 + 2x_2 \leq -4 \\ &\qquad x_1 \geq 0, \quad x_2 \geq 0. \end{aligned}$$

Its dual problem is

$$\begin{aligned} &\text{Minimize} \quad z' = -w_1 - 4w_2 \\ &\text{subject to} \\ &\qquad 2w_1 - 2w_2 \geq 3 \\ &\qquad -2w_1 + 2w_2 \geq 2 \\ &\qquad w_1 \geq 0, \quad w_2 \geq 0. \end{aligned}$$

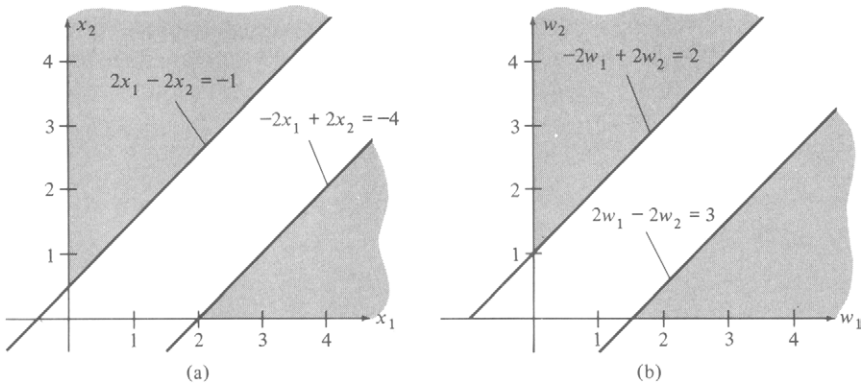


FIGURE 3.2

The graphs of the constraints of the primal problem are shown in Figure 3.2a, and the graphs for the dual problem are shown in Figure 3.2b. Neither of these problems has a feasible solution. △

Complementary Slackness

In addition to the relations between an optimal solution to the primal problem and an optimal solution to the corresponding dual problem, which we have already discussed, we can obtain information about which constraints may be "acting" on the solution to the primal problem. Specifically, we can show that if an optimal solution to the primal problem makes a constraint into a strict inequality, then the corresponding dual variable must be zero.

We consider a linear programming problem in standard form (13) as the primal problem. Its dual problem is given in (14). For these problems we state and prove the theorem on complementary slackness.

THEOREM 3.8. *For any pair of optimal solutions to the primal problem and its dual, we have*:

(*a*) *For* $i = 1, 2, \ldots, m$, *the product of the* i*th slack variable for the primal problem and the* i*th dual variable is zero. That is,* $x_{n+i}w_i = 0$, $i = 1, 2, \ldots, m$, *where* x_{n+i} *is the* i*th slack variable for the primal problem.*

(*b*) *For* $j = 1, 2, \ldots, n$, *the product of the* j*th slack variable for the dual problem and the* j*th variable for the primal problem is zero.*

Another way of stating the theorem is to say that, if the ith slack variable of the primal problem is not zero, then the ith dual variable must be zero. Likewise, if the jth slack variable for the dual problem is not zero, then the jth primal variable must be zero. Note that it is possible for both the slack variable and its corresponding dual variable to be zero.

Proof. We add slack variables to the primal problem and write

$$\mathbf{Ax} + \mathbf{Ix}' = \mathbf{b},$$

where

$$\mathbf{x}' = \begin{bmatrix} x_{n+1} \\ x_{n+2} \\ \vdots \\ x_{n+m} \end{bmatrix}$$

is the vector of slack variables. Then for any vector of dual variables

$$\mathbf{w} = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{bmatrix}$$

we have

$$\mathbf{w}^{\mathrm{T}}\mathbf{Ax} + \mathbf{w}^{\mathrm{T}}\mathbf{Ix}' = \mathbf{w}^{\mathrm{T}}\mathbf{b}.$$

This equality is an equality between numbers so that it is preserved when the transpose of each side is taken. We obtain

$$\mathbf{x}^{\mathrm{T}}\mathbf{A}^{\mathrm{T}}\mathbf{w} + \mathbf{x}'^{\mathrm{T}}\mathbf{w} = \mathbf{b}^{\mathrm{T}}\mathbf{w}. \tag{20}$$

At the optimal solutions $\begin{bmatrix} \mathbf{x}_0 \\ \mathbf{x}'_0 \end{bmatrix}$ and $\mathbf{w}_0$, we have, from the duality theorem,

$$\mathbf{c}^{\mathrm{T}}\mathbf{x}_0 = \mathbf{b}^{\mathrm{T}}\mathbf{w}_0. \tag{21}$$

Using (20), we may rewrite (21) as

$$\mathbf{c}^{\mathrm{T}}\mathbf{x}_0 = \mathbf{x}_0^{\mathrm{T}}\mathbf{A}^{\mathrm{T}}\mathbf{w}_0 + \mathbf{x}_0'^{\mathrm{T}}\mathbf{w}_0. \tag{22}$$

In the dual problem, $\mathbf{A}^{\mathrm{T}}\mathbf{w}_0 \geq \mathbf{c}$; hence, we may write (22) as

$$\mathbf{c}^{\mathrm{T}}\mathbf{x}_0 \geq \mathbf{x}_0^{\mathrm{T}}\mathbf{c} + \mathbf{x}_0'^{\mathrm{T}}\mathbf{w}_0. \tag{23}$$

Now $\mathbf{c}^{\mathrm{T}}\mathbf{x}_0$ is a number, so that $\mathbf{c}^{\mathrm{T}}\mathbf{x}_0 = (\mathbf{c}^{\mathrm{T}}\mathbf{x}_0)^{\mathrm{T}} = \mathbf{x}_0^{\mathrm{T}}\mathbf{c}$, implying $0 \geq \mathbf{x}'^{\mathrm{T}}_0\mathbf{w}_0$. Since $\mathbf{x}'_0 \geq \mathbf{0}$ and $\mathbf{w}_0 \geq \mathbf{0}$ imply that $\mathbf{x}_0'^{\mathrm{T}}\mathbf{w}_0 \geq 0$, we have equality in (23), and $\mathbf{x}'_0\mathbf{w}_0 = 0$. That is,

$$x_{n+1}w_1 + x_{n+2}w_2 + \cdots + x_{n+m}w_m = 0.$$

and each term is nonnegative. Therefore, for each $i = 1, 2, \ldots, m$, we have $x_{n+i}w_i = 0$. The proof of part (b) is similar. △

EXAMPLE 4. Consider the linear programming problem

$$\text{Maximize} \quad z = 2x_1 + x_2$$

subject to

$$x_1 + 2x_2 \leq 8$$
$$3x_1 + 4x_2 \leq 18$$
$$x_1 \geq 0, \quad x_2 \geq 0.$$

The dual of this problem is

$$\text{Minimize} \quad z' = 8w_1 + 18w_2$$

subject to

$$w_1 + 3w_2 \geq 2$$
$$2w_1 + 4w_2 \geq 1$$
$$w_1 \geq 0, \quad w_2 \geq 0.$$

The feasible regions for the primal and dual problems are shown in Figures 3.3a and 3.3b, respectively.

An optimal solution to the primal problem is (verify)

$$x_1 = 6, \quad x_2 = 0, \quad \text{and} \quad z = 12.$$

Since there is slack in the first constraint, the principle of complementary slackness says that the first dual variable must be zero in an optimal solution to the dual problem. Thus, without evaluating the objective function at the extreme points, we see that

$$w_1 = 0, \quad w_2 = \tfrac{2}{3}$$

must be an optimal solution to the dual problem. Furthermore, the value of the dual objective function at this extreme point must be 12. If there

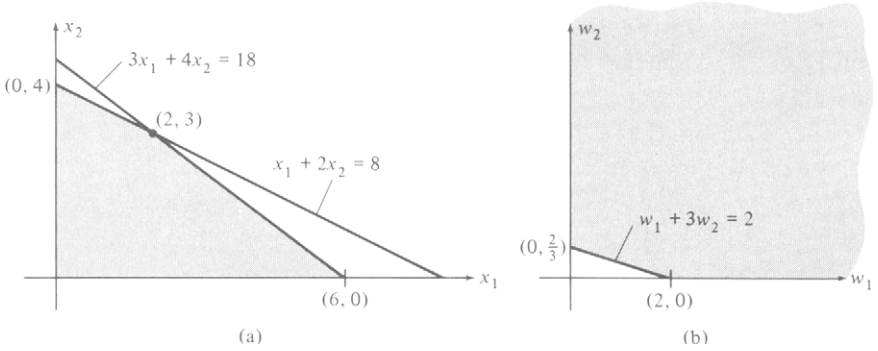


FIGURE 3.3

were several points for which $w_1 = 0$, a point at which the dual objective function has value 12 would be an optimal solution to the dual problem by the Duality Theorem (Theorem 3.7). △

The economic interpretation of complementary slackness is related to the understanding of the dual variables as marginal costs or shadow prices. Suppose that in an optimal solution to the primal problem the ith slack variable is nonzero. This means that there is more of the ith input available than is needed for this optimal solution. The value of the ith slack variable is exactly the excess of the ith input. But there is no need for any of this excess of the ith input; its marginal value is zero. The theorem on complementary slackness tells us that, if the ith primal slack variable is positive, then the ith dual variable, which can be thought of as the marginal value of the ith input, is zero. On the other hand, if in an optimal solution to the dual problem the ith dual variable is nonzero, its value can be thought of as the marginal value of the ith input. For in this case the ith primal slack variable is zero, indicating that all the ith input has been used and that it is desirable to have more. Its marginal value is positive.

The dual variables can also be viewed as a measure of the contribution of each resource to the maximum profit. Recall that at optimal solutions $z = z'$ or

$$\mathbf{b}^T\mathbf{w} = \mathbf{c}^T\mathbf{x} = \text{maximum profit.} \tag{24}$$

The constraints of the primal problem in which there is slack at an optimal solution correspond to dual variables that have value zero by complementary slackness. Thus, the corresponding resources do not contribute to the maximum profit. Considering the constraints of the primal problem in which there is no slack at an optimal solution, the values of the corresponding dual variables are nonzero and do indeed affect the maximum profit. We see from Equation (23) that the values of the nonzero dual variables divide the profit proportionately among the corresponding resources.

For example, in the sawmill problem, the saw may be assigned a value of \$35 per hour ($w_1 = 35$) and the plane may be assigned a value of \$10 per hour ($w_2 = 10$). The maximum profit of \$430 is attributable to the saw and the plane at the respective rates of \$35 per hour and \$10 per hour. That is, to make this profit, the saw is $3\frac{1}{2}$ times more valuable than the plane. In this sense the dual variables are accounting costs and would be useful for cost-accounting procedures.

We summarize the discussion and examples of this section by completing Table 3.3 by citing an example of the possible combinations of results for the primal and dual problem pairs. This summary appears in Table 3.3.

TABLE 3.3

	Primal		
Dual	*None*	*Finite*	*Unbounded*
None	Possible (Example 3)	Impossible (Theorem 3.7)	Possible (Example 2)
Finite	Impossible (Theorem 3.7)	Possible (Example 4)	Impossible (Theorem 3.5)
Unbounded	Possible (Example 2)	Impossible (Theorem 3.5)	Impossible (Theorem 3.5)

3.2 EXERCISES

In Exercises 1–4, properties of solutions to a given linear programming problem are specified. Describe in as much detail as possible the solutions to the dual of the given problem. (*Hint:* You may need to use Theorems 3.5–3.8.)

1. An optimal solution is $[0.27 \quad 1.83 \quad 0.94 \quad 0.5 \quad 0 \quad 0]^T$ with objective function value 117.81.
2. An optimal solution to the dual problem is $[0 \quad 3 \quad 15 \quad 0 \quad 5]^T$ with dual objective function value 125.
3. There are no feasible solutions to the problem.
4. There are solutions to the dual problem with arbitrarily large dual objective function values.
5. Suppose for the linear programming problem

$$\begin{aligned} &\text{Maximize} \quad z = \mathbf{c}^T\mathbf{x} \\ &\text{subject to} \\ &\qquad \mathbf{Ax} \leq \mathbf{b} \\ &\qquad \mathbf{x} \geq \mathbf{0}, \end{aligned}$$

we know that $\mathbf{b} = [12 \quad 21 \quad 8 \quad 2 \quad 5]^T$. Assume that $\mathbf{w} = [0 \quad 4 \quad 5 \quad 0 \quad 3]^T$ is an optimal solution to the dual of the given problem. Calculate the optimal value of the objective function for the given problem.

In Exercises 6 and 7, solve the given linear programming problem using the simplex method. Also, set up and solve the dual of the given problem. Finally, verify that your solutions satisfy part (b) (iii) of the Duality Theorem.

6. Minimize $z = 4x + 6y$
subject to

$$\begin{aligned} x + 3y &\geq 5 \\ 2x + y &\geq 3 \\ x \geq 0, \quad y &\geq 0. \end{aligned}$$

7. Maximize $z = 8x_1 + 9x_2 + 5x_3$
subject to

$$\begin{aligned} x_1 + x_2 + 2x_3 &\le 2 \\ 2x_1 + 3x_2 + 4x_3 &\le 3 \\ 3x_1 + 3x_2 + x_3 &\le 4 \\ x_j &\ge 0, \quad j = 1,2,3 \end{aligned}$$

8. A health food store packages a nut sampler consisting of walnuts, pecans, and almonds. Suppose that each ounce of walnuts contains 12 units of protein and 3 units of iron and costs 12 cents, that each ounce of pecans contains 1 unit of protein and 3 units of iron and costs 9 cents, and that each ounce of almonds contains 2 units of protein and 1 unit of iron and costs 6 cents. If each package of the nut sampler is to contain at least 24 units of protein and at least 18 units of iron, how many ounces of each type of nut should be used to minimize the cost of the sampler? (*Hint:* Set up and solve the dual problem. Then use the principle of complementary slackness to solve the given problem.)
9. Show without using the Simplex Method that $x_1 = \frac{5}{26}$, $x_2 = \frac{5}{2}$, $x_3 = \frac{27}{26}$ is an optimal solution to the following linear programming problem:

Maximize $z = 9x_1 + 14x_2 + 7x_3$

subject to

$$\begin{aligned} 2x_1 + x_2 + 3x_3 &\le 6 \\ 5x_1 + 4x_2 + x_3 &\le 12 \\ 2x_2 &\le 5 \\ x_1, \quad x_2, \quad x_3 &\text{ unrestricted.} \end{aligned}$$

(*Hint:* Formulate the dual of this problem and then find a feasible solution.)
10. Consider the linear programming problem

Maximize $z = 3x_1 + 4x_2$

subject to

$$\begin{aligned} x_1 + 2x_2 &\le 10 \\ x_1 + x_2 &\le 8 \\ 3x_1 + 5x_2 &\le 26 \\ x_1 \ge 0, \quad x_2 &\ge 0. \end{aligned}$$

By using the principle of complementary slackness, show that $w_1 = 0$ in any optimal solution to the dual problem.

11. Suppose that $x_1 = 2,\ x_2 = 0,\ x_3 = 4$ is an optimal solution to the linear programming problem

$$\begin{aligned} &\text{Maximize} \quad z = 4x_1 + 2x_2 + 3x_3 \\ &\text{subject to} \\ &\qquad 2x_1 + 3x_2 + x_3 \le 12 \\ &\qquad x_1 + 4x_2 + 2x_3 \le 10 \\ &\qquad 3x_1 + x_2 + x_3 \le 10 \\ &\qquad x_1 \ge 0, \quad x_2 \ge 0, \quad x_3 \ge 0. \end{aligned}$$

Using the principle of complementary slackness and the duality theorem (Theorem 3.7), find an optimal solution to the dual problem. What value will the objective function of the dual problem have at this optimal solution?

3.3 COMPUTATIONAL RELATIONS BETWEEN THE PRIMAL AND DUAL PROBLEMS

We now modify the form of a tableau and the steps in the pivoting process to take advantage of the new information we have. We add an additional row and column to a tableau. The row is written above the column labels and contains the vector $\mathbf{c}^{\mathrm{T}} = [c_1 \quad c_2 \quad \dots \quad c_s]$. The column is written to the left of the column denoting the basic variables and contains the vector

$$\mathbf{c}_{\mathrm{B}} = \begin{bmatrix} c_{i_1} \\ c_{i_2} \\ \vdots \\ c_{i_m} \end{bmatrix}.$$

The entries of $\mathbf{c}_{\mathrm{B}}$ can be determined by copying the values of c_j from the new top row corresponding to the basic variables of the tableau.

The pivoting step can be changed to compute the entries of the objective row by using (12) of Section 3.2 rather than by using elementary row operations. Recall that the entries of the objective row are $z_j - c_j$. We now rework an example using these new ideas.

EXAMPLE 1. Consider the linear programming problem from Examples 2 and 3 in Section 2.3.

$$\begin{aligned} &\text{Maximize} \quad z = x_1 - 2x_2 - 3x_3 - x_4 - x_5 + 2x_6 \\ &\text{subject to} \\ &x_1 + 2x_2 + 2x_3 + x_4 + x_5 + x_7 = 12 \\ &x_1 + 2x_2 + x_3 + x_4 + 2x_5 + x_6 = 18 \\ &3x_1 + 6x_2 + 2x_3 + x_4 + 3x_5 + x_8 = 24 \\ &\qquad x_j \ge 0, \quad j = 1, 2, \dots, 8. \end{aligned}$$

We have

$$\mathbf{A} = \begin{bmatrix} 1 & 2 & 2 & 1 & 1 & 0 & 1 & 0 \\ 1 & 2 & 1 & 1 & 2 & 1 & 0 & 0 \\ 3 & 6 & 2 & 1 & 3 & 0 & 0 & 1 \end{bmatrix},$$

$$\mathbf{b} = \begin{bmatrix} 12 \\ 18 \\ 24 \end{bmatrix}, \quad \text{and} \quad \mathbf{c} = \begin{bmatrix} 1 \\ -2 \\ -3 \\ -1 \\ -1 \\ 2 \\ 0 \\ 0 \end{bmatrix}.$$

Note that this problem is in the form of Equation (1) in Section 3.2. The matrix **A** contains the identity matrix by taking columns 7, 6, and 8 in this order. We have denoted the artificial variables by x_7 and x_8 rather than by y_1 and y_2 to be consistent with (1).

The initial tableau for the auxiliary problem of the two-phase method is shown in Tableau 3.2a, for which the objective row has not been filled in. Recall from Section 2.3 that we had to eliminate the initial basic variables x_7 and x_8 from the objective function by substitution. This substitution procedure is replaced by the procedure of computing the objective row as $z_j - c_j$. We have

$$\mathbf{c}_B^T = [\,-1 \quad 0 \quad -1\,]$$

since $i_1 = 7$, $i_2 = 6$, and $i_3 = 8$. The entry in column 1 of the objective row is, by (11),

$$z_1 - c_1 = [\,-1 \quad 0 \quad -1\,]\begin{bmatrix} 1 \\ 1 \\ 3 \end{bmatrix} - 0 = -4.$$

Tableau 3.2a

$\mathbf{c}_B$		0 x_1	0 x_2	0 x_3	0 x_4	0 x_5	0 x_6	−1 x_7	−1 x_8	$\mathbf{x}_B$
−1	x_7	1	2	2	1	1	0	1	0	12
0	x_6	1	2	1	1	2	1	0	0	18
−1	x_8	3	6	2	1	3	0	0	1	24

In the same way we compute the other entries obtaining the full initial tableau (Tableau 3.2b). Note that it is the same as Tableau 2.23 (Section 2.3).

Tableau 3.2b

			↓								
	$\mathbf{c}_B$		0 x_1	0 x_2	0 x_3	0 x_4	0 x_5	0 x_6	−1 x_7	−1 x_8	$\mathbf{x}_B$
	−1	x_7	1	2	2	1	1	0	1	0	12
	0	x_6	1	2	1	1	2	1	0	0	18
←	−1	x_8	3	⑥	2	1	3	0	0	1	24
			−4	−8	−4	−2	−4	0	0	0	−36

The value of the last entry in the objective row is computed as

$$z = \mathbf{c}_B^T \mathbf{x}_B,$$

where $\mathbf{x}_B$ is the last column of the tableau. For this tableau,

$$\mathbf{B} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \text{and} \quad \mathbf{B}^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

since

$$i_1 = 7 \quad \text{and} \quad \mathbf{A}_{i_1} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}; \qquad i_2 = 6 \quad \text{and} \quad \mathbf{A}_{i_2} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix};$$

$$i_3 = 8 \quad \text{and} \quad \mathbf{A}_{i_3} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}.$$

The entering and departing variables are determined as they were previously. We find x_2 is the entering variable and x_8 is the departing variable. In the next tableau (Tableau 3.3), x_7, x_6, and x_2 will be the basic variables, in this order, so that $i_1 = 7$, $i_2 = 6$, and $i_3 = 2$. Thus,

$$\mathbf{c}_B = \begin{bmatrix} -1 \\ 0 \\ 0 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} 1 & 0 & 2 \\ 0 & 1 & 2 \\ 0 & 0 & 6 \end{bmatrix},$$

and

$$\mathbf{B}^{-1} = \begin{bmatrix} 1 & 0 & -\frac{1}{3} \\ 0 & 1 & -\frac{1}{3} \\ 0 & 0 & \frac{1}{6} \end{bmatrix}.$$

Tableau 3.3

	$\mathbf{c}_B$		0 x_1	0 x_2	0 x_3	0 x_4	0 x_5	0 x_6	−1 x_7	−1 x_8	$\mathbf{x}_B$
←	−1	x_7	0	0	$\textcircled{\frac{4}{3}}$	$\frac{2}{3}$	0	0	1	$-\frac{1}{3}$	4
	0	x_6	0	0	$\frac{1}{3}$	$\frac{2}{3}$	1	1	0	$-\frac{1}{3}$	10
	0	x_2	$\frac{1}{2}$	1	$\frac{1}{3}$	$\frac{1}{6}$	$\frac{1}{2}$	0	0	$\frac{1}{6}$	4
			0	0	$-\frac{4}{3}$	$-\frac{2}{3}$	0	0	0	$\frac{4}{3}$	−4

We may verify that

$$\mathbf{x}_B = \mathbf{B}^{-1}\mathbf{b} = \begin{bmatrix} 1 & 0 & -\frac{1}{3} \\ 0 & 1 & -\frac{1}{3} \\ 0 & 0 & \frac{1}{6} \end{bmatrix} \begin{bmatrix} 12 \\ 18 \\ 24 \end{bmatrix} = \begin{bmatrix} 4 \\ 10 \\ 4 \end{bmatrix}.$$

We also know that $t_j = \mathbf{B}^{-1}\mathbf{A}_j$, $j = 1, 2, \ldots, 8$, and we may verify that, for example,

$$t_3 = \mathbf{B}^{-1}\mathbf{A}_3 = \begin{bmatrix} 1 & 0 & -\frac{1}{3} \\ 0 & 1 & -\frac{1}{3} \\ 0 & 0 & \frac{1}{6} \end{bmatrix} \begin{bmatrix} 2 \\ 1 \\ 2 \end{bmatrix} = \begin{bmatrix} \frac{4}{3} \\ \frac{1}{3} \\ \frac{1}{3} \end{bmatrix}.$$

The objective row is computed after the usual procedure for pivoting is applied to find the other rows. The entries in the objective row are found by

$$z_j - c_j = \mathbf{c}_B^T \mathbf{t}_j - c_j.$$

For example, the entry in the third column of Tableau 3.3 is

$$z_3 - c_3 = \mathbf{c}_B^T \mathbf{t}_3 - c_3 = [-1 \quad 0 \quad 0] \begin{bmatrix} \frac{4}{3} \\ \frac{1}{3} \\ \frac{1}{3} \end{bmatrix} - 0 = -\frac{4}{3}.$$

We see that Tableau 3.3 is the same as Tableau 2.32 of Section 2.3. We continue the simplex algorithm by determining the entering and departing variables for Tableau 3.3 and pivoting to form Tableau 3.4.

Tableau 3.4

c_B		0 x_1	0 x_2	0 x_3	0 x_4	0 x_5	0 x_6	−1 x_7	−1 x_8	x_B
0	x_3	0	0	1	$\frac{1}{2}$	0	0	$\frac{3}{4}$	$-\frac{1}{4}$	3
0	x_6	0	0	0	$\frac{1}{2}$	1	1	$-\frac{1}{4}$	$-\frac{1}{4}$	9
0	x_2	$\frac{1}{2}$	1	0	0	$\frac{1}{2}$	0	$-\frac{1}{4}$	$\frac{1}{4}$	3
		0	0	0	0	0	0	1	1	0

Tableau 3.4 gives an optimal solution to the auxiliary problem. The next step is to form the initial tableau for Phase 2, as described in Section 2.3. The columns corresponding to the artificial variables x_7 and x_8 are deleted and the entries in the objective row of this tableau are computed as $z_j - c_j$. Remember to put the coefficients of the original objective function at the top of the tableau. Carrying out these steps, we obtain Tableau 3.5.

Tableau 3.5

	c_B		↓ 1 x_1	−2 x_2	−3 x_3	−1 x_4	−1 x_5	2 x_6	x_B
	−3	x_3	0	0	1	$\frac{1}{2}$	0	0	3
	2	x_6	0	0	0	$\frac{1}{2}$	1	1	9
←	−2	x_2	(1/2)	1	0	0	$\frac{1}{2}$	0	3
			−2	0	0	$\frac{1}{2}$	2	0	3

Pivoting, we obtain Tableau 3.6, which yields an optimal solution to the given problem. The reader should check that he or she understands how the entries were determined.

Tableau 3.6

c_B		−2 x_1	−3 x_2	−1 x_3	−1 x_4	2 x_5	0 x_6	x_B
−3	x_3	0	0	1	$\frac{1}{2}$	0	0	3
2	x_6	0	0	0	$\frac{1}{2}$	1	1	9
1	x_1	1	2	0	0	1	0	6
		0	4	0	$\frac{1}{2}$	4	0	15

△

We have now defined enough tools to complete our discussion of artificial variables. Recall that we had taken a problem in canonical form

$$\begin{aligned} &\text{Maximize} \quad z = \mathbf{c}^{\mathrm{T}}\mathbf{x} \\ &\text{subject to} \\ &\qquad \mathbf{A}\mathbf{x} = \mathbf{b} \\ &\qquad \mathbf{x} \geq \mathbf{0}, \end{aligned}$$

where $\mathbf{b} \geq \mathbf{0}$. We introduced artificial variables into each of the constraint equations. For Phase 1 we used a different objective function, namely,

$$\text{Minimize} \quad z = y_1 + y_2 + \cdots + y_m$$

where y_i, $i = 1, 2, \ldots, m$ are the artificial variables. Suppose that at the end of Phase 1, the minimum of this objective function is zero but that there are artificial variables which remain basic (at value zero) in the final optimal tableau. We now proceed as follows.

Phase 2

The initial tableau for Phase 2 is the final tableau of Phase 1 with the following modifications.

(a) Delete the columns from the final tableau of Phase 1 that are labeled with the nonbasic artificial variables.

(b) Replace the row above the column labels with the coefficients of the original objective function, assigning 0 as a cost for each of the basic artificial variables.

(c) Form the vector $\mathbf{c}_{\mathrm{B}}$ from the new row of objective function coefficients.

(d) Calculate the entries in the new objective row as $z_j - c_j = \mathbf{c}_{\mathrm{B}}^{\mathrm{T}}\mathbf{t}_j - c_j$.

As we proceed with the simplex method for Phase 2 we must ensure that the remaining artificial variables do not take on positive values. This would happen when one of these variables remained basic and the pivotal elimination gave a positive entry in $\mathbf{x}_{\mathrm{B}}$ for the position labeled by the artificial variable. Suppose that x_k is to be the entering variable and that the rows labeled by artificial variables are $i_1, i_2, \ldots, i_p$. Denote the kth column of the current tableau by

$$\mathbf{t}_k = \begin{bmatrix} t_{1k} \\ t_{2k} \\ \vdots \\ t_{mk} \end{bmatrix}.$$

It can be shown (Exercise 18) that if

$$t_{i_1 k} \geq 0, \quad t_{i_2 k} \geq 0, \quad \ldots, \quad t_{i_p k} \geq 0,$$

then none of the artificial variables that are basic will take on a positive value. If, however, we have

$$t_{i,k} < 0$$

for some r, $r = 1, 2, \ldots, p$, then the usual simplex procedure could cause the artificial variable that labels row i_r to take on a positive value. Consequently, we must modify the usual simplex procedure. The new procedure for selecting a departing variable is as follows. If at least one of the entries in the entering variable column corresponding to a row labeled by an artificial variable is negative, choose one of these artificial variables as the departing variable. Otherwise, use the usual simplex procedure to obtain a finite optimal solution or to discover that such a solution does not exist.

EXAMPLE 2. In Section 2.3 we showed that the problem given in Example 5 ended with an artificial variable in the basis. We rework this example using the two-phase method. The original problem in canonical form is

$$\begin{aligned} &\text{Maximize} \quad z = x_1 + 2x_2 + x_3 \\ &\text{subject to} \\ &\qquad 3x_1 + \ x_2 - x_3 = 15 \\ &\qquad 8x_1 + 4x_2 - x_3 = 50 \\ &\qquad 2x_1 + 2x_2 + x_3 = 20 \\ &\quad x_1 \geq 0, \quad x_2 \geq 0, \quad x_3 \geq 0. \end{aligned}$$

The new problem for Phase 1 is

$$\begin{aligned} &\text{Minimize} \quad z = y_1 + y_2 + y_3 \\ &\text{subject to} \\ &3x_1 + \ x_2 - x_3 + y_1 \qquad\quad = 15 \\ &8x_1 + 4x_2 - x_3 \qquad + y_2 \qquad = 50 \\ &2x_1 + 2x_2 + x_3 \qquad\quad + y_3 = 20 \\ &x_j \geq 0, \quad j = 1,2,3; \qquad y_i \geq 0, \quad i = 1,2,3. \end{aligned}$$

We rewrite the objective function as

$$\text{Maximize} \quad z = -y_1 - y_2 - y_3$$

to have a maximization problem.

We now have the following sequence of tableaux (Tableaux 3.7–3.10) for Phase 1. The objective row of the initial tableau is computed by using $z_j - c_j$. Since the initial basic variables are y_1, y_2, and y_3, we have

$$\mathbf{c}_B = \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix}$$

and

$$z_1 - c_1 = [-1 \quad -1 \quad -1]\begin{bmatrix}3\\8\\2\end{bmatrix} - 0 = -13.$$

Similarly,

$$z_2 - c_2 = [-1 \quad -1 \quad -1]\begin{bmatrix}1\\4\\2\end{bmatrix} - 0 = -7$$

and

$$z_3 - c_3 = 1.$$

For the basic variables y_1, y_2, and y_3 we have

$$z_4 - c_4 = [-1 \quad -1 \quad -1]\begin{bmatrix}1\\0\\0\end{bmatrix} - (-1) = 0$$

and, similarly, $z_5 - c_5 = 0$ and $z_6 - c_6 = 0$.

The value of the objective function is

$$[-1 \quad -1 \quad -1]\begin{bmatrix}15\\50\\20\end{bmatrix} = -85.$$

The results of these computations are shown in Tableaux 3.7–3.10.

Tableau 3.7

			↓						
	c_B		0 x_1	0 x_2	0 x_3	−1 y_1	−1 y_2	−1 y_3	x_B
←	−1	y_1	③	1	−1	1	0	0	15
	−1	y_2	8	4	−1	0	1	0	50
	−1	y_3	2	2	1	0	0	1	20
			−13	−7	1	0	0	0	−85

Tableau 3.8

					↓				
	c_B		0 x_1	0 x_2	0 x_3	−1 y_1	−1 y_2	−1 y_3	x_B
	0	x_1	1	$\frac{1}{3}$	$-\frac{1}{3}$	$\frac{1}{3}$	0	0	5
←	−1	y_2	0	$\frac{4}{3}$	⑤/3	$-\frac{8}{3}$	1	0	10
	−1	y_3	0	$\frac{4}{3}$	$\frac{5}{3}$	$-\frac{2}{3}$	0	1	10
			0	$-\frac{8}{3}$	$-\frac{10}{3}$	$\frac{13}{3}$	0	0	−20

Tableau 3.9

					↓			
c_B		0 x_1	0 x_2	0 x_3	−1 y_1	−1 y_2	−1 y_3	x_B
0	x_1	1	$\frac{3}{5}$	0	$-\frac{1}{5}$	$\frac{1}{5}$	0	7
0	x_3	0	$\frac{4}{5}$	1	$-\frac{8}{5}$	$\frac{3}{5}$	0	6
← −1	y_3	0	0	0	②	−1	1	0
		0	0	0	−1	2	0	0

Tableau 3.10

c_B		0 x_1	0 x_2	0 x_3	−1 y_1	−1 y_2	−1 y_3	x_B
0	x_1	1	$\frac{3}{5}$	0	0	$\frac{1}{10}$	$\frac{1}{10}$	7
0	x_3	0	$\frac{4}{5}$	1	0	$-\frac{1}{5}$	$\frac{4}{5}$	6
−1	y_1	0	0	0	1	$-\frac{1}{2}$	$\frac{1}{2}$	0
		0	0	0	0	$\frac{3}{2}$	$\frac{1}{2}$	0

Thus, Phase 1 has an optimal solution with $x_1 = 7$, $x_3 = 6$, $y_1 = 0$ and value 0 for the objective function. The artificial variable y_1 appears as a basic variable in the optimal solution.

The initial tableau for Phase 2 is shown in Tableau 3.11. The columns corresponding to y_2 and y_3 have been deleted and a cost of 0 has been assigned to y_1. The objective row has been filled in, using $z_j - c_j$.

Tableau 3.11

				↓		
c_B		1 x_1	2 x_2	1 x_3	0 y_1	x_B
1	x_1	1	$\frac{3}{5}$	0	0	7
← 1	x_3	0	⑷/5	1	0	6
0	y_1	0	0	0	1	0
		0	$-\frac{3}{5}$	0	0	13

We now apply the simplex method to Tableau 3.11. In selecting the departing variable we make sure the entry in the row labeled by y_1 and the pivotal column is nonnegative. If it is not, we will choose y_1 as the departing variable. We get Tableau 3.12.

Tableau 3.12

$\mathbf{c}_B$		1 x_1	2 x_2	1 x_3	0 y_1	$\mathbf{x}_B$
1	x_1	1	0	$-\frac{3}{4}$	0	$\frac{5}{2}$
2	x_2	0	1	$\frac{5}{4}$	0	$\frac{15}{2}$
0	y_1	0	0	0	1	0
		0	0	$\frac{3}{4}$	0	$\frac{35}{2}$

An optimal solution to the original problem is therefore

$$x_1 = \tfrac{5}{2}$$
$$x_2 = \tfrac{15}{2}$$
$$x_3 = 0,$$

which gives $\frac{35}{2}$ as the optimal value of the objective function. △

Solution to Dual Problem from Final Tableau of Primal Problem

One of our objectives in this section is to describe how to find an optimal solution to the dual problem from the final tableau for the primal problem. We discuss the easiest case first, namely, when the primal problem is in canonical form,

$$\begin{aligned} &\text{Maximize} \quad z = \mathbf{c}^T\mathbf{x} \\ &\text{subject to} \\ &\qquad \mathbf{A}\mathbf{x} = \mathbf{b} \\ &\qquad \mathbf{x} \geq \mathbf{0}, \end{aligned}$$

where $\mathbf{A}$ contains the identity matrix and $\mathbf{b} \geq \mathbf{0}$. In particular this situation arises when the primal problem is given in standard form and has been converted to canonical form by adding slack variables. There are two easily given descriptions for finding an optimal solution to the dual of the problem given above. The dual problem is

$$\begin{aligned} &\text{Minimize} \quad z' = \mathbf{b}^T\mathbf{w} \\ &\text{subject to} \\ &\qquad \mathbf{A}^T\mathbf{w} \geq \mathbf{c} \\ &\qquad \mathbf{w} \text{ unrestricted}. \end{aligned}$$

An optimal solution

$$\mathbf{w} = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_m \end{bmatrix}$$

to it is given by

$$\mathbf{w}^{\mathrm{T}} = \mathbf{c}_{\mathrm{B}}^{\mathrm{T}}\mathbf{B}^{-1}, \tag{1}$$

where $\mathbf{B}$ is the matrix consisting of certain columns of the initial tableau. The columns that are used in $\mathbf{B}$ correspond to the basic variables of the final tableau of the primal problem. We can find $\mathbf{B}^{-1}$ from the *final* tableau as follows. From our assumptions about $\mathbf{A}$ and $\mathbf{b}$ we may infer that the columns labeled by the initial basic variables in the initial tableau form the $m \times m$ identity matrix when they are properly ordered. It can be shown that the columns in the final tableau with the same labels as the initial basic variables and arranged in the same order give $\mathbf{B}^{-1}$.

It can also be shown that an optimal solution to the dual problem is given by

$$w_j = c_{i_j} + (z_{i_j} - c_{i_j}), \tag{2}$$

where the subscript i_j ranges over the indices of the *initial basic variables*. Of course, c_{i_j} is the entry above the label of the i_jth column and $z_{i_j} - c_{i_j}$ is the corresponding entry in the objective row. This second description shows that if an initial basic variable is a slack or artificial variable, the value of the corresponding dual variable is the entry in the i_jth column of the objective row of the final tableau of the primal problem. This fact follows, since $c_{i_j} = 0$ for any slack or artificial variable.

EXAMPLE 3. Consider as our primal problem the linear programming problem

$$\text{Maximize} \quad z = 8x_1 + 9x_2 + 4x_3$$

subject to

$$\begin{aligned} x_1 + x_2 + 2x_3 &\leq 2 \\ 2x_1 + 3x_2 + 4x_3 &\leq 3 \\ 7x_1 + 6x_2 + 2x_3 &\leq 8 \\ x_1 \geq 0, \quad x_2 \geq 0, \quad x_3 &\geq 0. \end{aligned}$$

Introducing the slack variables x_4, x_5, and x_6, our primal problem becomes

$$\begin{aligned}
&\text{Maximize} \quad z = 8x_1 + 9x_2 + 4x_3 \\
&\text{subject to} \\
&x_1 + x_2 + 2x_3 + x_4 = 2 \\
&2x_1 + 3x_2 + 4x_3 + x_5 = 3 \\
&7x_1 + 6x_2 + 2x_3 + x_6 = 8 \\
&x_j \geq 0, \quad j = 1,2,\ldots,6.
\end{aligned}$$

Solving this problem by the simplex method we are led to the sequence of Tableaux 3.13–3.15.

The dual problem in this example is

$$\begin{aligned}
&\text{Minimize} \quad z' = 2w_1 + 3w_2 + 8w_3 \\
&\text{subject to} \\
&w_1 + 2w_2 + 7w_3 \geq 8 \\
&w_1 + 3w_2 + 6w_3 \geq 9 \\
&2w_1 + 4w_2 + 2w_3 \geq 4 \\
&w_1 \geq 0, \quad w_2 \geq 0, \quad w_3 \geq 0.
\end{aligned}$$

The solution to this problem is found from Tableau 3.15. In Tableau 3.15 the basic variables are x_4, x_2, and x_1, in that order. Therefore, reading from Tableau 3.13, we find

$$\mathbf{A}_4 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \quad \mathbf{A}_2 = \begin{bmatrix} 1 \\ 3 \\ 6 \end{bmatrix}, \quad \mathbf{A}_1 = \begin{bmatrix} 1 \\ 2 \\ 7 \end{bmatrix},$$

and

$$\mathbf{B} = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 3 & 2 \\ 0 & 6 & 7 \end{bmatrix}.$$

Since the initial basic variables are x_4, x_5, and x_6, in that order, we find the columns of $\mathbf{B}^{-1}$ under the labels x_4, x_5, and x_6 in Tableau 3.15. Thus,

$$\mathbf{B}^{-1} = \begin{bmatrix} 1 & -\frac{1}{9} & -\frac{1}{9} \\ 0 & \frac{7}{9} & -\frac{2}{9} \\ 0 & -\frac{2}{3} & \frac{1}{3} \end{bmatrix}.$$

Tableau 3.13

	$\mathbf{c}_B$		8 x_1	9 x_2	4 x_3	0 x_4	0 x_5	0 x_6	$\mathbf{x}_B$
	0	x_4	1	1	2	1	0	0	2
←	0	x_5	2	③	4	0	1	0	3
	0	x_6	7	6	2	0	0	1	8
			−8	−9	−4	0	0	0	0

Tableau 3.14

	$\mathbf{c}_B$		8 x_1	9 x_2	4 x_3	0 x_4	0 x_5	0 x_6	$\mathbf{x}_B$
	0	x_4	$\frac{1}{3}$	0	$\frac{2}{3}$	1	$-\frac{1}{3}$	0	1
	9	x_2	$\frac{2}{3}$	1	$\frac{4}{3}$	0	$\frac{1}{3}$	0	1
←	0	x_6	③	0	−6	0	−2	1	2
			−2	0	8	0	3	0	9

Tableau 3.15

$\mathbf{c}_B$		8 x_1	9 x_2	4 x_3	0 x_4	0 x_5	0 x_6	$\mathbf{x}_B$
0	x_4	0	0	$\frac{4}{3}$	1	$-\frac{1}{9}$	$-\frac{1}{9}$	$\frac{7}{9}$
9	x_2	0	1	$\frac{8}{3}$	0	$\frac{7}{9}$	$-\frac{2}{9}$	$\frac{5}{9}$
8	x_1	1	0	−2	0	$-\frac{2}{3}$	$\frac{1}{3}$	$\frac{2}{3}$
		0	0	4	0	$\frac{5}{3}$	$\frac{2}{3}$	$\frac{31}{3}$

Then an optimal solution to the dual problem is, by (1),

$$\mathbf{w}^T = \mathbf{c}_B^T\mathbf{B}^{-1} = [0 \quad 9 \quad 8]\begin{bmatrix} 1 & -\frac{1}{9} & -\frac{1}{9} \\ 0 & \frac{7}{9} & -\frac{2}{9} \\ 0 & -\frac{2}{3} & \frac{1}{3} \end{bmatrix} = \begin{bmatrix} 0 & \frac{5}{3} & \frac{2}{3} \end{bmatrix}.$$

If (2) is used, an optimal value of w_1 is

$$c_4 - (z_4 - c_4) = 0 + 0 = 0$$

since x_4 was the first initial basic variable. Likewise, an optimal value of w_2 comes from the second initial basic variable x_5 and is given as

$$w_2 = c_5 + (z_5 - c_5) = 0 + \tfrac{5}{3} = \tfrac{5}{3}.$$

Finally,

$$w_3 = c_6 + (z_6 - c_6) = 0 + \tfrac{2}{3} = \tfrac{2}{3}.$$

Thus, this solution to the dual problem yields the value $\frac{31}{3}$ for the dual objective function, which checks with the value of the primal objective function. Theorem 3.6 assures us that these solutions are indeed optimal, since they yield the same value for the objective functions. △

Now let us consider finding a solution to the dual of an arbitrary general linear programming problem. As we discussed in earlier sections, such a problem can always be converted to one in the following form:

$$\left.\begin{aligned} &\text{Maximize} \quad z = \mathbf{c}^{\mathrm{T}}\mathbf{x} \\ &\text{subject to} \\ &\qquad \mathbf{A}^{(1)}\mathbf{x} \le \mathbf{b}^{(1)} \\ &\qquad \mathbf{A}^{(2)}\mathbf{x} \ge \mathbf{b}^{(2)} \\ &\qquad \mathbf{A}^{(3)}\mathbf{x} = \mathbf{b}^{(3)} \\ &\qquad \mathbf{x} \ge \mathbf{0} \\ &\mathbf{b}^{(1)} \ge \mathbf{0}, \quad \mathbf{b}^{(2)} > \mathbf{0}, \quad \mathbf{b}^{(3)} \ge \mathbf{0}. \end{aligned}\right\} \tag{3}$$

This conversion does not change the set of feasible solutions but may change the objective function of the original problem by multiplying it by -1. A problem in the form above is ready to be solved by the simplex method after slack variables and artificial variables are introduced.

After multiplying the second set of constraints of (3) by -1, we may set up the dual problem. It is

$$\left.\begin{aligned} &\text{Minimize} \quad z' = \begin{bmatrix} \mathbf{b}^{(1)\mathrm{T}} & -\mathbf{b}^{(2)\mathrm{T}} & \mathbf{b}^{(3)\mathrm{T}} \end{bmatrix} \begin{bmatrix} \mathbf{w}^{(1)} \\ \mathbf{w}^{(2)} \\ \mathbf{w}^{(3)} \end{bmatrix} \\ &\text{subject to} \\ &\qquad \begin{bmatrix} \mathbf{A}^{(1)\mathrm{T}} & -\mathbf{A}^{(2)\mathrm{T}} & \mathbf{A}^{(3)\mathrm{T}} \end{bmatrix} \begin{bmatrix} \mathbf{w}^{(1)} \\ \mathbf{w}^{(2)} \\ \mathbf{w}^{(3)} \end{bmatrix} \ge \mathbf{c} \\ &\mathbf{w}^{(1)} \ge \mathbf{0}, \quad \mathbf{w}^{(2)} \ge \mathbf{0}, \quad \mathbf{w}^{(3)} \text{ unrestricted}. \end{aligned}\right\} \tag{4}$$

We seek the solution to (4) using the final tableau from the solution to (3). If the two-phase method is used, the columns labeled by artificial variables must not be discarded at the end of Phase 1. They must be kept throughout Phase 2 and must be modified accordingly by each pivoting step. However, the entries in the objective row labeled by artificial variables must *not* be used when checking the optimality criterion. This means

that there may be negative numbers in the objective row when an optimal solution is reached, but these numbers will only be in columns labeled by artificial variables.

Assuming that all the columns labeled by artificial variables are available, an optimal solution to the dual problem (4) can be obtained from (1) or (2). If (1) is used, then $\mathbf{B}^{-1}$ is automatically available, as described in Example 3, in the final tableau of Phase 2 from the columns labeled by the initial basic variables. We then compute

$$\hat{\mathbf{w}}^{\mathrm{T}} = \mathbf{c}_{\mathrm{B}}^{\mathrm{T}}\mathbf{B}^{-1},$$

but $\hat{\mathbf{w}}$ is *not* an optimal solution to the dual problem (4). Because the second set of constraints in (3) was multiplied by -1 to find the dual problem but was not changed to use the simplex algorithm, those entries in $\hat{\mathbf{w}}$ corresponding to the second set of constraints in (3) must be multiplied by -1. The vector thus formed is $\mathbf{w}$, an optimal solution to the dual problem (4).

If (2) is used, we must distinguish between the two-phase and the big M methods. If the two-phase method is used, then the cost of an artificial variable is 0, so that

$$\hat{w}_i = 0 + (z_i - 0) = z_i.$$

With the big M method the cost of an artificial variable c_i is $-M$, and $z_i - c_i$ will be of the form $k + M$, so that

$$\hat{w}_i = -M + (k + M) = k.$$

In either case we must, as above, multiply each of the $\hat{w}_i$ by -1, where i runs through the indices of the second set of constraints in (3). The set of values $w_1, w_2, \ldots, w_m$ thus obtained is an optimal solution to the dual problem.

EXAMPLE 4. Consider as our primal problem the linear programming problem

$$\text{Maximize} \quad z = 3x_1 + 2x_2 + 5x_3$$

subject to

$$\begin{aligned} x_1 + 3x_2 + 2x_3 &\leq 15 \\ 2x_2 - x_3 &\geq 5 \\ 2x_1 + x_2 - 5x_3 &= 10 \\ x_1 \geq 0, \quad x_2 &\geq 0, \quad x_3 \geq 0. \end{aligned}$$

After multiplying the second constraint of the primal problem by -1, we find that the dual of the resulting problem is given as

$$\text{Minimize} \quad z' = 15w_1 - 5w_2 + 10w_3$$

subject to

$$\begin{aligned} w_1 \qquad\quad + 2w_3 &\geq 3 \\ 3w_1 - 2w_2 + w_3 &\geq 2 \\ 2w_1 + w_2 - 5w_3 &\geq 5 \end{aligned}$$

$$w_1 \geq 0, \quad w_2 \geq 0, \quad w_3 \text{ unrestricted.}$$

Introducing the slack variables x_4 and x_5 and the artificial variables y_1 and y_2, we can formulate the auxiliary problem to the primal problem as

$$\text{Minimize} \quad z = y_1 + y_2$$

subject to

$$\begin{aligned} x_1 + 3x_2 + 2x_3 + x_4 &= 15 \\ 2x_2 - x_3 - x_5 + y_1 &= 5 \\ 2x_1 + x_2 - 5x_3 + y_2 &= 10 \end{aligned}$$

$$x_i \geq 0, \quad i = 1,2,3,4; \qquad y_1 \geq 0, \quad y_2 \geq 0.$$

From this statement of the primal problem we can construct our initial tableau (Tableau 3.16).

The initial basic variables are x_4, y_1, and y_2. At the end of Phase 1 we have Tableau 3.17.

Tableau 3.16

c_B		0 x_1	0 x_2	0 x_3	0 x_4	0 x_5	−1 y_1	−1 y_2	x_B
0	x_4	1	3	2	1	0	0	0	15
−1	y_1	0	2	−1	0	−1	1	0	5
−1	y_2	2	1	−5	0	0	0	1	10
		−2	−3	6	0	1	0	0	−15

Tableau 3.17

c_B		0 x_1	0 x_2	0 x_3	0 x_4	0 x_5	−1 y_1	−1 y_2	x_B
0	x_4	0	0	$\frac{23}{4}$	1	$\frac{5}{4}$	$-\frac{5}{4}$	$-\frac{1}{2}$	$\frac{15}{4}$
0	x_2	0	1	$-\frac{1}{2}$	0	$-\frac{1}{2}$	$\frac{1}{2}$	0	$\frac{5}{2}$
0	x_1	1	0	$-\frac{9}{4}$	0	$\frac{1}{4}$	$-\frac{1}{4}$	$\frac{1}{2}$	$\frac{15}{4}$
		0	0	0	0	0	1	1	0

Converting to Phase 2 but keeping the columns labeled with artificial variables, we construct Tableau 3.18.

Tableau 3.18

			3	2	5	0	0	0	0	
	$\mathbf{c}_B$		x_1	x_2	x_3	x_4	x_5	y_1	y_2	$\mathbf{x}_B$
←	0	x_4	0	0	$\frac{23}{4}$	1	$\frac{5}{4}$	$-\frac{5}{4}$	$-\frac{1}{2}$	$\frac{15}{4}$
	2	x_2	0	1	$-\frac{1}{2}$	0	$-\frac{1}{2}$	$\frac{1}{2}$	0	$\frac{5}{2}$
	3	x_1	1	0	$-\frac{9}{4}$	0	$\frac{1}{4}$	$-\frac{1}{4}$	$\frac{1}{2}$	$\frac{15}{4}$
			0	0	$-\frac{51}{4}$	0	$-\frac{1}{4}$	$\frac{1}{4}$	$\frac{3}{2}$	$\frac{65}{4}$

Now we perform another iteration of the simplex method, including the results of the pivoting step on the columns labeled by artificial variables but ignoring these columns when applying the optimality criterion. We obtain the final tableau (Tableau 3.19). Using (1) to obtain an optimal solution to the dual problem, we first compute

$$\hat{\mathbf{w}}^T = \mathbf{c}_B^T \mathbf{B}^{-1}$$

or

$$\hat{\mathbf{w}}^T = [5 \quad 2 \quad 3] \begin{bmatrix} \frac{4}{23} & -\frac{5}{23} & -\frac{2}{23} \\ \frac{2}{23} & \frac{9}{23} & -\frac{1}{23} \\ \frac{9}{23} & -\frac{17}{23} & \frac{7}{23} \end{bmatrix}$$

$$= \begin{bmatrix} \frac{51}{23} & -\frac{58}{23} & \frac{9}{23} \end{bmatrix}.$$

As before,

$$w_1 = \hat{w}_1, \quad w_2 = -\hat{w}_2, \quad \text{and} \quad w_3 = \hat{w}_3.$$

Tableau 3.19

		3	2	5	0	0	0	0	
$\mathbf{c}_B$		x_1	x_2	x_3	x_4	x_5	y_1	y_2	$\mathbf{x}_B$
5	x_3	0	0	1	$\frac{4}{23}$	$\frac{5}{23}$	$-\frac{5}{23}$	$-\frac{2}{23}$	$\frac{15}{23}$
2	x_2	0	1	0	$\frac{2}{23}$	$-\frac{9}{23}$	$\frac{9}{23}$	$-\frac{1}{23}$	$\frac{65}{23}$
3	x_1	1	0	0	$\frac{9}{23}$	$\frac{17}{23}$	$-\frac{17}{23}$	$\frac{7}{23}$	$\frac{120}{23}$
		0	0	0	$\frac{51}{23}$	$\frac{58}{23}$	$-\frac{58}{23}$	$\frac{9}{23}$	$\frac{565}{23}$

Thus, an optimal solution to the dual problem is

$$\mathbf{w}^T = \begin{bmatrix} \frac{51}{23} & \frac{58}{23} & \frac{9}{23} \end{bmatrix}$$

and the value of the dual objective function is

$$z' = 15(\tfrac{51}{23}) - 5(\tfrac{58}{23}) + 10(\tfrac{9}{23}) = \tfrac{565}{23}.$$

We see from Tableau 3.18 that an optimal solution to the primal problem is

$$x_1 = \tfrac{120}{23}, \quad x_2 = \tfrac{65}{23}, \quad x_3 = \tfrac{15}{23}$$

and the value of the objective function is $\frac{565}{23}$, which is the same as the value for the dual problem.

We can also solve the primal problem using the big M method, obtaining the final tableau (Tableau 3.20). Using (2) we can obtain an optimal solution to the dual problem from Tableau 3.20 as follows. Since the first initial basic variable is x_4, we then find

$$w_1 = \hat{w}_1 = c_4 + (z_4 - c_4) = 0 + \tfrac{51}{23} = \tfrac{51}{23}.$$

The second initial basic variable is y_1, so that

$$\hat{w}_2 = -M + (M - \tfrac{58}{23}) = -\tfrac{58}{23}.$$

But since the second constraint was multiplied by -1 in forming the dual problem, we must multiply $\hat{w}_2$ by -1 to obtain the value of the dual variable at the optimum. We have

$$w_2 = -\hat{w}_2 = \tfrac{58}{23}.$$

Proceeding as in the case of the first dual variable, we find

$$w_3 = \hat{w}_3 = -M + M + \tfrac{9}{23} = \tfrac{9}{23}.$$

Substituting these values into the objective function, we get

$$z' = 15(\tfrac{51}{23}) - 5(\tfrac{58}{23}) + 90(\tfrac{9}{23}) = \tfrac{565}{23},$$

which is the same as the value for the primal problem.

Tableau 3.20

		3	2	5	0	0	$-M$	$-M$	
$\mathbf{c}_B$		x_1	x_2	x_3	x_4	x_5	y_1	y_2	$\mathbf{x}_B$
5	x_3	0	0	1	$\frac{4}{23}$	$\frac{5}{23}$	$-\frac{5}{23}$	$-\frac{2}{23}$	$\frac{15}{23}$
2	x_2	0	1	0	$\frac{2}{23}$	$-\frac{9}{23}$	$\frac{9}{23}$	$-\frac{1}{23}$	$\frac{65}{23}$
3	x_1	1	0	0	$\frac{9}{23}$	$\frac{17}{23}$	$-\frac{17}{23}$	$\frac{7}{23}$	$\frac{120}{23}$
		0	0	0	$\frac{51}{23}$	$\frac{58}{23}$	$M - \frac{58}{23}$	$M + \frac{9}{23}$	$\frac{565}{23}$

△

3.3 EXERCISES

In Exercises 1 and 2 fill in *all* the missing entries in the given simplex tableaux.

1.

$\mathbf{c}_B$		-1 x_1	x_2	x_3	0 x_4	0 x_5	5 x_6	$\mathbf{x}_B$
2	x_2	-3		-1	-2			2
	x_6	2		0	-1			3
		6		7	6			1
				4				

2.

$\mathbf{c}_B$	4 x_1	$\frac{5}{3}$ x_2	$\frac{4}{3}$ x_3	3 x_4	-1 x_5	0 x_6	$-\frac{2}{3}$ x_7	$\mathbf{x}_B$
	0	$\frac{4}{3}$	$\frac{2}{3}$	0	1	0	$-\frac{1}{3}$	4
	0	$\frac{1}{3}$	$\frac{2}{3}$	1	0	1	$-\frac{1}{3}$	10
	1	$\frac{1}{3}$	$\frac{1}{6}$	$\frac{1}{2}$	0	0	$\frac{1}{6}$	4

In Exercises 3 and 4 we give the original objective function of a linear programming problem and the final tableau at the end of Phase 1. Find the initial tableau for Phase 2 and solve the resulting linear programming problem.

3. Maximize $z = 2x_1 + 3x_2 + 5x_3 + x_4$.

$\mathbf{c}_B$		x_1	x_2	x_3	x_4	x_5	x_6	x_7	y_1	y_2	$\mathbf{x}_B$
0	x_1	1	$\frac{1}{2}$	1	0	3	0	-1	0	0	$\frac{1}{2}$
0	x_4	0	1	$-\frac{1}{2}$	1	-2	0	-1	0	0	4
0	x_6	0	2	$-\frac{2}{3}$	0	$-\frac{1}{2}$	1	0	0	0	$\frac{3}{2}$
-1	y_1	0	$-\frac{3}{2}$	-2	0	-3	0	0	1	0	0
-1	y_2	0	-4	1	0	2	0	-2	0	1	0
		0	$\frac{11}{2}$	1	0	1	0	2	0	0	0

4. Maximize $z = 3x_1 + x_2 + 3x_3$.

$\mathbf{c}_B$		x_1	x_2	x_3	x_4	x_5	x_6	y_1	$\mathbf{x}_B$
0	x_1	1	1	0	2	0	$\frac{2}{3}$	0	2
0	x_3	0	-1	1	-1	0	-1	0	$\frac{5}{2}$
0	x_5	0	2	0	3	1	1	0	$\frac{2}{3}$
-1	y_1	0	$-\frac{3}{2}$	0	$-\frac{1}{2}$	0	-2	1	0
		0	$\frac{3}{2}$	0	$\frac{1}{2}$	0	2	0	0

5. Verify the entries in Tableaux 3.4 and 3.5.

In Exercises 6–12 solve the given linear programming problem calculating $z_j - c_j$ as described in this section and using the new format for the tableaux.

6. Maximize $z = 2x_1 + x_2 + 3x_3$
subject to

$$\begin{aligned} 2x_1 - x_2 + 3x_3 &\leq 6 \\ x_1 + 3x_2 + 5x_3 &\leq 10 \\ 2x_1 + x_3 &\leq 7 \\ x_1 \geq 0, \quad x_2 \geq 0, \quad x_3 &\geq 0. \end{aligned}$$

7. Maximize $z = x_1 + x_2 + x_3 + x_4$
subject to

$$\begin{aligned} x_1 + 2x_2 - x_3 + 3x_4 &\leq 12 \\ x_1 + 3x_2 + x_3 + 2x_4 &\leq 8 \\ 2x_1 - 3x_2 - x_3 + 2x_4 &\leq 7 \\ x_1 \geq 0, \quad x_2 \geq 0, \quad x_3 \geq 0, \quad x_4 &\geq 0. \end{aligned}$$

8. Minimize $z = 8x_1 + 6x_2 + 11x_3$
subject to

$$\begin{aligned} 5x_1 + x_2 + 3x_3 &\leq 4 \\ 5x_1 + x_2 + 3x_3 &\geq 2 \\ 2x_1 + 4x_2 + 7x_3 &\leq 5 \\ 2x_1 + 4x_2 + 7x_3 &\geq 3 \\ x_1 + x_2 + x_3 &= 1 \\ x_1 \geq 0, \quad x_2 \geq 0, \quad x_3 &\geq 0. \end{aligned}$$

(Exercise 14 will require the use of the columns corresponding to artificial variables.)

9. Minimize $z = 4x_1 + x_2 + x_3 + 3x_4$
subject to

$$\begin{aligned} 2x_1 + x_2 + 3x_3 + x_4 &\geq 12 \\ 3x_1 + 2x_2 + 4x_3 &= 5 \\ 2x_1 - x_2 + 2x_3 + 3x_4 &= 8 \\ 3x_1 + 4x_2 + 3x_3 + x_4 &\geq 16 \\ x_1 \geq 0, \quad x_2 \geq 0, \quad x_3 \geq 0, \quad x_4 &\geq 0. \end{aligned}$$

10. Maximize $z = 2x_1 + x_2 + x_3 + x_4$
subject to

$$\begin{aligned} x_1 + 2x_2 + x_3 + 2x_4 &\leq 7 \\ x_1 + 2x_2 + x_3 + 2x_4 &\geq 3 \\ 2x_1 + 3x_2 - x_3 - 4x_4 &\leq 10 \\ x_1 + x_2 + x_3 + x_4 &= 1 \\ x_1 \geq 0, \quad x_2 \geq 0, \quad x_3 \geq 0, \quad x_4 &\geq 0. \end{aligned}$$

11. Maximize $z = 3x_1 + x_2 + 3x_3 + x_4$
subject to

$$\begin{aligned} x_1 + x_2 + 4x_3 + x_4 &\leq 6 \\ 2x_1 + 6x_3 + 2x_4 &\geq 8 \\ 20x_1 + 2x_2 + 47x_3 + 11x_4 &\leq 48 \\ x_1 \geq 0, \quad x_2 \geq 0, \quad x_3 \geq 0, \quad x_4 &\geq 0 \end{aligned}$$

12. Maximize $z = 2x_1 + x_2 + 3x_4$
subject to

$$\begin{aligned} 9x_1 + 14x_2 - 6x_3 - 6x_4 &\leq 2 \\ x_1 + x_2 - x_3 - x_4 &\geq -1 \\ -20x_1 - 5x_2 + 5x_3 + 13x_4 &= 11 \\ 5x_1 + 10x_2 - 2x_3 + 14x_4 &= 6 \\ x_1 \geq 0, \quad x_2 \geq 0, \quad x_3 \geq 0, \quad x_4 &\geq 0. \end{aligned}$$

13. For each of the linear programming problems in Exercises 6 and 9 find the matrices $\mathbf{B}$ and $\mathbf{B}^{-1}$ from each of the tableaux that arose in the solution of the problem.
14. For each of the primal linear programming problems in Exercises 6 and 8 find an optimal solution to the dual problem using the final tableau determined in solving the primal problem.
15. Solve Example 5 in Section 1.1.
16. Solve Project 2 in Section 1.1.
17. Solve Project 3 in Section 1.1.
18. Verify the remarks preceding Example 2 regarding Phase 2.

3.4 THE DUAL SIMPLEX METHOD

In modeling applied problems as linear programming problems, it frequently becomes necessary to add more constraints to the model. These constraints are generally used to make the model more accurately represent the real problem, and their need becomes evident when the researcher compares the solution to the linear programming problem with the situation being modeled. However, adding one or more constraints may cause the existing solution to become infeasible. In this case the dual simplex method, discussed in this section, can be used to restore feasibility without having to resolve the entire new problem.

When we use the simplex algorithm on a primal problem we begin with a feasible but nonoptimal solution. Each iteration of the simplex algorithm finds a feasible solution that is closer to optimality, and this procedure continues until an optimal solution is reached. In the meantime, what is happening to the dual problem? Let us examine the sawmill problem in this context.

EXAMPLE 1. The primal problem in standard form for the model is

$$\begin{aligned} &\text{Maximize} \quad z = 120x + 100y \\ &\text{subject to} \\ &\qquad 2x + 2y \le 8 \\ &\qquad 5x + 3y \le 15 \\ &\qquad x \ge 0, \quad y \ge 0. \end{aligned}$$

The dual problem is

$$\begin{aligned} &\text{Minimize} \quad z' = 8s + 15t \\ &\text{subject to} \\ &\qquad 2s + 5t \ge 120 \\ &\qquad 2s + 3t \ge 100 \\ &\qquad s \ge 0, \quad t \ge 0. \end{aligned}$$

The initial tableau for the primal problem, after adding the necessary slack variables, is as follows.

Tableau 3.21

c_B		120 x	100 y	0 u	0 v	x_B
0	u	2	2	1	0	8
0	v	5		0	1	15
		−120	−100	0	0	0

From this tableau we see that

$$\mathbf{c}_B = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \quad \text{and} \quad \mathbf{B}^{-1} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix},$$

and we may compute from the formula $\mathbf{w}^T = \mathbf{c}_B^T \mathbf{B}^{-1}$ that

$$\mathbf{w}^T = [s \quad t] = [0 \quad 0]\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = [0 \quad 0].$$

Note that this "solution" to the dual problem satisfies the nonnegativity conditions but neither of the constraints.

We now pivot in Tableau 3.21 and obtain Tableau 3.22.

Tableau 3.22

		120	100	0	0	
$\mathbf{c}_B$		x	y	u	v	$\mathbf{x}_B$
0	u	0	$\frac{4}{5}$	1	$-\frac{2}{5}$	2
120	x	1	$\frac{3}{5}$	0	$\frac{1}{5}$	3
		0	-28	0	24	360

From this tableau we see that

$$\mathbf{c}_B = \begin{bmatrix} 0 \\ 120 \end{bmatrix} \quad \text{and} \quad \mathbf{B}^{-1} = \begin{bmatrix} 1 & -\frac{2}{5} \\ 0 & \frac{1}{5} \end{bmatrix}$$

and that

$$\mathbf{w}^T = [s \quad t] = \begin{bmatrix} 0 \\ 120 \end{bmatrix}\begin{bmatrix} 1 & -\frac{2}{5} \\ 0 & \frac{1}{5} \end{bmatrix} = [0 \quad 24].$$

We now have a "solution" to the dual problem that satisfies the nonnegativity conditions and also satisfies the first but not the second constraint of the dual problem. We pivot again, obtaining Tableau 3.23.

Tableau 3.23

		120	100	0	0	
$\mathbf{c}_B$		x	y	u	v	$\mathbf{x}_B$
100	y	0	1	$\frac{5}{4}$	$-\frac{1}{2}$	$\frac{5}{2}$
120	x	1	0	$-\frac{3}{4}$	$\frac{1}{2}$	$\frac{3}{2}$
		0	0	35	10	430

From this tableau we see that

$$\mathbf{c}_B = \begin{bmatrix} 100 \\ 120 \end{bmatrix} \quad \text{and} \quad \mathbf{B}^{-1} = \begin{bmatrix} \frac{5}{4} & -\frac{1}{2} \\ -\frac{3}{4} & \frac{1}{2} \end{bmatrix}.$$

Thus,

$$\mathbf{w}^T = [s \quad t] = [35 \quad 10].$$

This is a feasible solution to the dual problem: it satisfies the nonnegativity conditions and both of the constraints of the problem. The objective function value for the dual problem using this solution is the same as the objective function value for the primal problem with the corresponding solution. From the Duality Theorem, we have found an optimal solution to the dual problem. △

From this example we have seen that if the primal problem has a solution that is feasible and nonoptimal, then the solution determined for the dual problem is infeasible. As the simplex method progresses, the solutions determined for the dual problem are all infeasible until the optimal solution is attained for the primal problem. The dual solution corresponding to the optimal primal solution is both optimal and feasible. The goal for the primal problem when using the simplex method is to achieve optimality. The goal for a corresponding method for the dual problem is to achieve feasibility, that is, to have both nonnegativity constraints and resource constraints satisfied.

The dual simplex method handles problems for which it is easy to obtain an initial basic solution that is infeasible but satisfies the optimality criterion. That is, the initial tableau has nonnegative entries in the objective row but negative entries in the right-hand column. The following example will be used as we present our description of the dual simplex algorithm.

EXAMPLE 2. Consider the following linear programming problem:

$$\begin{aligned} &\text{Maximize} \quad z = -x_1 - 2x_2 \\ &\text{subject to} \\ &\qquad x_1 - 2x_2 + x_3 \geq 4 \\ &\qquad 2x_1 + \ x_2 - x_3 \geq 6 \\ &x_1 \geq 0, \quad x_2 \geq 0, \quad x_3 \geq 0. \end{aligned}$$

We change each constraint to an $\leq$ inequality and then introduce slack variables x_4 and x_5. The result is a problem in canonical form:

$$\begin{aligned} &\text{Maximize} \quad z = -x_1 - 2x_2 \\ &\text{subject to} \\ &-x_1 + 2x_2 - x_3 + x_4 \qquad = -4 \\ &-2x_1 - \ x_2 + x_3 \qquad + x_5 = -6 \\ &\qquad x_j \geq 0, \quad j = 1,2,\ldots,5. \end{aligned}$$

The initial tableau for the simplex algorithm is given in Tableau 3.24. It has x_4 and x_5 as the initial basic variables. The solution that this tableau

represents is

$$x_1 = 0, \quad x_2 = 0, \quad x_3 = 0, \quad x_4 = -4, \quad x_5 = -6.$$

Tableau 3.24

$\mathbf{c}_B$		-1 x_1	-2 x_2	0 x_3	0 x_4	0 x_5	$\mathbf{x}_B$
0	x_4	-1	2	-1	1	0	-4
0	x_5	-2	-1	1	0	1	-6
		1	2	0	0	0	0

This solution is not feasible, and here $z = 0$. The entries in the objective row show that the optimality criterion is satisfied. △

The dual simplex method consists of two parts: a feasibility criterion that tells us whether the current solution (which satisfies the optimality criterion) is feasible, and a procedure for getting a new solution that removes some of the infeasibilities of the current solution and consequently drives the current solution toward a feasible solution. The dual simplex method consists of the following steps.

1. Find an initial basic solution such that all entries in the objective row are nonnegative and at least one basic variable has a negative value. (Tableau 3.24 represents this step for our example.)
2. Select a departing variable by examining the basic variables and choosing the most negative one. This is the departing variable and the row it labels is the pivotal row.
3. Select an entering variable. This selection depends on the ratios of the objective row entries to the corresponding pivotal row entries. The ratios are formed only for those entries of the pivotal row that are negative. If all entries in the pivotal row are nonnegative, the problem has no feasible solution. Among all the ratios (which must all be nonpositive), select the maximum ratio. The column for which this ratio occurred is the pivotal column and the corresponding variable is the entering variable. In case of ties among the ratios, choose one column arbitrarily.
4. Perform pivoting to obtain a new tableau. The objective row can be computed as $z_j - c_j = \mathbf{c}_B^T \mathbf{t}_j - c_j$, where $\mathbf{t}_j$ is the jth column of the new tableau.
5. The process stops when a basic solution that is feasible (all variables ≥ 0) is obtained.

A flowchart for the dual simplex method is given in Figure 3.4 and a structure diagram is given in Figure 3.5.

EXAMPLE 2 (CONTINUED). Continuing with our example, we perform Step 2 of the dual simplex algorithm. We see that $x_5 = -6$ is the most

Input is a tableau which satisfies the optimality criterion.

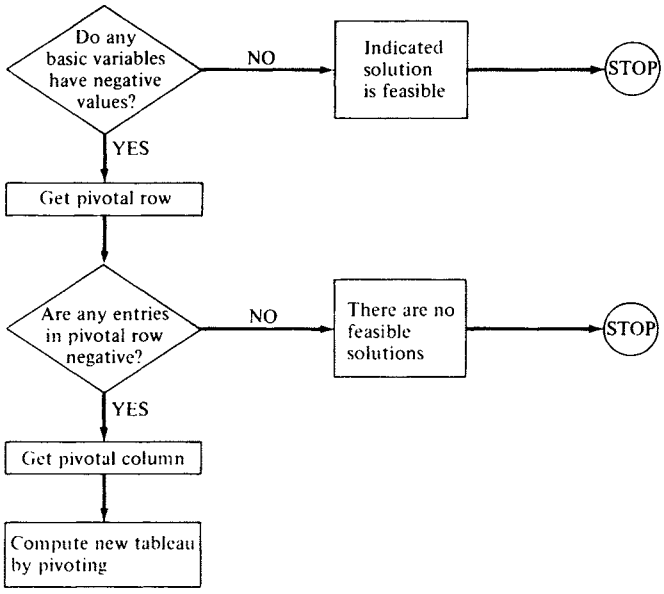


FIGURE 3.4 Flowchart for the dual simplex algorithm.

Input is a tableau which satisfies the optimality criterion.

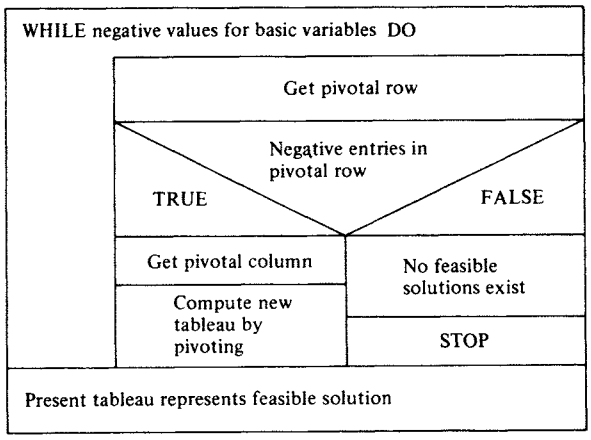


FIGURE 3.5 Structure diagram for the dual simplex algorithm.

negative basic variable, so that x_5 is the departing variable. The ratios of the entries in the objective row to corresponding negative entries in the pivotal row are

$$\text{Column 1:} \quad -\tfrac{1}{2}$$
$$\text{Column 2:} \quad -2.$$

The maximum ratio is $-\frac{1}{2}$, so that x_1 is the entering variable. We repeat Tableau 3.24 with the entering and departing variables and the pivot labeled (Tableau 3.24a).

Tableau 3.24a

			↓				
	c_B		-1 x_1	-2 x_2	0 x_3	0 x_4	0 x_5	x_B
	0	x_4	-1	2	-1	1	0	-4
←	0	x_5	(−2)	-1	1	0	1	-6
			1	2	0	0	0	0

We now perform a pivotal elimination to get Tableau 3.25. The basic solution given by Tableau 3.25 is

$$x_1 = 3, \quad x_2 = 0, \quad x_3 = 0, \quad x_4 = -1, \quad x_5 = 0.$$

This solution is still optimal (the objective row has nonnegative entries) but is infeasible. However, it is less infeasible in that only one variable is negative.

Tableau 3.25

				↓			
	c_B		-1 x_1	-2 x_2	0 x_3	0 x_4	0 x_5	x_B
←	0	x_4	0	$\frac{5}{2}$	$(-\frac{3}{2})$	1	$-\frac{1}{2}$	-1
	-1	x_1	1	$\frac{1}{2}$	$-\frac{1}{2}$	0	$-\frac{1}{2}$	3
			0	$\frac{3}{2}$	$\frac{1}{2}$	0	$\frac{1}{2}$	-3

For the next iteration of the dual simplex algorithm, x_4 is the departing variable, since it is the only negative basic variable. Forming the ratios of the entries in the objective row to the corresponding negative entries of the pivotal row, we have

$$\text{Column 3:} \quad \tfrac{1}{2} / -\tfrac{3}{2} = -\tfrac{1}{3}$$
$$\text{Column 5:} \quad \tfrac{1}{2} / -\tfrac{1}{2} = -1.$$

The maximum ratio is $-\frac{1}{3}$, so that x_3 is the entering variable. Pivoting, we now obtain Tableau 3.26.

Tableau 3.26

c_B		-1 x_1	-2 x_2	0 x_3	0 x_4	0 x_5	x_B
0	x_3	0	$-\frac{5}{3}$	1	$-\frac{2}{3}$	$\frac{1}{3}$	$\frac{2}{3}$
-1	x_1	1	$-\frac{1}{3}$	0	$-\frac{1}{3}$	$-\frac{1}{3}$	$\frac{10}{3}$
		0	$\frac{7}{3}$	0	$\frac{1}{3}$	$\frac{1}{3}$	$-\frac{10}{3}$

The basic solution represented by Tableau 3.26 is

$$x_1 = \tfrac{10}{3}, \quad x_2 = 0, \quad x_3 = \tfrac{2}{3}, \quad x_4 = 0, \quad x_5 = 0.$$

This solution satisfies the optimality criterion, *and* it is feasible since all the variables have nonnegative values. △

While using the dual simplex method in Example 2, we were fortunate in finding an initial basic solution to the given problem that satisfied the optimality criterion but was not feasible. In general, it is difficult to find such a starting point for the dual simplex method. However, the principal use of this method is to restore feasibility when additional constraints are included in a linear programming problem whose solution is known. The following example illustrates this situation.

EXAMPLE 3. There are three typical kinds of dog food that Paws eats each day: dry food, canned wet food, and dog biscuits. The nutritional analysis of his favorite brands is given in the following table in percent by weight.

	Fat	*Protein*	*Fiber*	*Moisture*	*Cost (¢/oz)*
Dry food	8.0	14.0	5.5	12.0	4.1
Wet food	6.0	9.0	1.5	78.0	2.5
Biscuit	8.0	21.0	4.5	12.0	7.3

The veterinarian has suggested that Paws get at least 5 oz of protein and at most 1 oz of fiber each day. His owner has set up the following linear programming problem to model the dietary requirements and minimize the cost, where x_1 is the amount of dry food measured in ounces offered

to Paws. Similarly x_2 denotes the amount of wet food and x_3 denotes the amount of biscuits.

$$\begin{aligned} &\text{Minimize} \quad z = 4.1x_1 + 2.5x_2 + 7.3x_3 \\ &\text{subject to} \\ &0.14x_1 + 0.09x_2 + 0.21x_3 \geq 5.0 \\ &0.055x_1 + 0.015x_2 + 0.045x_3 \leq 1.0 \\ &x_j \geq 0, \quad j = 1,2,3 \end{aligned}$$

We first transform this problem to standard form and then to canonical form introducing the slack variables x_4 and x_5:

$$\begin{aligned} &\text{Maximize} \quad z = -4.1x_1 - 2.5x_2 - 7.3x_3 \\ &\text{subject to} \\ &-0.14x_1 - 0.09x_2 - 0.21x_3 + x_4 = -5.0 \\ &0.055x_1 + 0.015x_2 + 0.045x_3 + x_5 = 1.0 \\ &x_j \geq 0, \quad j = 1,2,\ldots,5. \end{aligned}$$

Solving this linear programming problem using the dual simplex method we obtain (verify) Tableau 3.27.

Tableau 3.27

c_B		-4.1 x_1	-2.5 x_2	-7.3 x_3	0 x_4	0 x_5	x_B
-2.5	x_2	1.556	1	2.333	-11.111	0	55.556
0	x_5	0.032	0	0.010	0.167	1	0.167
		0.211	0	1.467	27.778	0	-138.889

From this tableau we see that the minimum-cost diet for Paws consists of 55.556 ounces of wet food per day (no dry food and no biscuits, poor Paws) at a cost of $1.39 per day.

At the most recent visit to the veterinarian, she also suggested that Paws' fat intake be limited to 2.5 oz per day. The new constraint,

$$0.08x_1 + 0.06x_2 + 0.08x_3 \leq 2.5,$$

expresses this limitation. Introducing the slack variable x_6 the new constraint becomes

$$0.08x_1 + 0.06x_2 + 0.08x_3 + x_6 = 2.5.$$

We now include this constraint into Tableau 3.27 and form Tableau 3.28. Observe that the new slack variable x_6 does not appear in the other two constraints nor in the objective function, so that its coefficient in these two constraints and in the objective row will be zero.

Tableau 3.28

		−4.1	−2.5	−7.3	0	0	0	
c_B		x_1	x_2	x_3	x_4	x_5	x_6	x_B
−2.5	x_2	1.556	1	2.333	−11.111	0	0	55.556
0	x_5	0.032	0	0.010	0.167	1	0	0.167
0	x_6	0.08	0.06	0.08	0	0	1	2.5
		0.211	0	1.467	27.778	0	0	−138.889

Moreover, the new slack variable will be a basic variable. Since x_2 is a basic variable, all the entries in the column labeled by x_2 must be zero except for the entry in the row labeled by x_2. Hence, we add (-0.06) times the first row to the third row and obtain Tableau 3.29.

Tableau 3.29

			↓					
		−4.1	−2.5	−7.3	0	0	0	
c_B		x_1	x_2	x_3	x_4	x_5	x_6	x_B
−2.5	x_2	1.556	1	2.333	−11.111	0	0	55.556
0	x_5	0.032	0	0.010	0.167	1	0	0.167
← 0	x_6	−0.013	0	−0.06	0.667	0	1	−0.833
		0.211	0	1.467	27.778	0	0	−138.889

This tableau represents an infeasible solution that satisfies the optimality criterion.

Using the dual simplex method, we may restore feasibility. We see that x_6 is the departing variable and that x_1 is the entering variable for this step. Completing several iterations of the dual simplex method we obtain Tableau 3.30 (verify).

Tableau 3.30

		−4.1	−2.5	−7.3	0	0	0	
c_B		x_1	x_2	x_3	x_4	x_5	x_6	x_B
−2.5	x_2	0	1	0	5.031	−35.220	33.019	22.170
−4.1	x_1	1	0	0	9.434	33.962	5.660	0.943
−7.3	x_3	0	0	1	−13.208	−7.547	−17.925	13.679
		0	0	0	45.157	3.899	25.094	−159.151

We conclude that after his fat intake was restricted, the minimum-cost diet for Paws is 0.943 oz of dry food, 22.17 oz of wet food, and 13.679 oz of dog biscuits per day at a total cost of $1.59. △

Much of our discussion of the simplex method centered on finding an initial basic feasible solution to a linear programming problem in arbitrary

form. We developed the concept of artificial variables to provide a method for constructing a starting point for such a problem. The situation with the dual simplex method is different. In this book we will use the dual simplex method to restore feasibility in a tableau that represents a solution that is infeasible but satisfies the optimality criterion. Thus, we will not need any procedures for finding an initial basic feasible solution when using the dual simplex method.

3.4 EXERCISES

In Exercises 1–5 the given tableau represents a solution to a linear programming problem that satisfies the optimality criterion, but is infeasible. Use the dual simplex method to restore feasibility.

1.

$\mathbf{c}_B$		5 x_1	6 x_2	0 x_3	0 x_4	0 x_5	$\mathbf{x}_B$
5	x_1	1	0	$\frac{1}{8}$	$-\frac{1}{8}$	0	$\frac{17}{4}$
6	x_2	0	1	$-\frac{1}{12}$	$\frac{5}{12}$	0	$\frac{19}{6}$
0	x_5	0	0	$-\frac{1}{8}$	$-\frac{7}{8}$	1	$-\frac{1}{4}$
		0	0	$\frac{1}{8}$	$\frac{15}{8}$	0	$\frac{161}{4}$

2.

$\mathbf{c}_B$		5 x_1	6 x_2	0 x_3	0 x_4	0 x_5	0 x_6	$\mathbf{x}_B$
5	x_1	1	0	0	−1	1	0	4
6	x_2	0	1	0	1	$-\frac{2}{3}$	0	$\frac{10}{3}$
0	x_3	0	0	1	7	−8	0	2
0	x_6	0	0	0	0	$-\frac{1}{3}$	1	$-\frac{1}{3}$
		0	0	0	1	1	0	40

3.

$\mathbf{c}_B$		4 x_1	5 x_2	3 x_3	0 x_4	0 x_5	0 x_6	0 x_7	$\mathbf{x}_B$
3	x_3	$\frac{3}{4}$	0	1	$\frac{1}{4}$	0	0	0	$\frac{5}{2}$
0	x_5	$\frac{11}{16}$	0	0	$-\frac{3}{16}$	1	$-\frac{1}{4}$	0	$\frac{17}{8}$
5	x_2	$\frac{9}{16}$	1	0	$-\frac{1}{16}$	0	$\frac{1}{4}$	0	$\frac{19}{8}$
0	x_7	$-\frac{1}{4}$	0	0	$-\frac{1}{4}$	0	0	1	$-\frac{1}{2}$
		$\frac{17}{16}$	0	0	$\frac{7}{16}$	0	$\frac{5}{4}$	0	$\frac{155}{8}$

4.

c_B		5 x_1	6 x_2	0 x_3	0 x_4	0 x_5	0 x_6	x_B
5	x_1	1	0	0	0	1	0	4
6	x_2	0	1	0	$\frac{1}{3}$	$-\frac{2}{3}$	0	$\frac{10}{3}$
0	x_3	0	0	1	-1	-8	0	2
0	x_6	0	0	0	$\frac{1}{3}$	$\frac{2}{3}$	1	$-\frac{1}{3}$
		0	0	0	2	1	0	40

5.

c_B		7 x_1	3 x_2	0 x_3	0 x_4	0 x_5	0 x_6	x_B
0	x_3	0	0	1	$-\frac{1}{4}$	$-\frac{17}{4}$	0	$\frac{19}{2}$
7	x_1	1	0	0	$\frac{1}{8}$	$-\frac{3}{8}$	0	$\frac{1}{4}$
3	x_2	0	1	0	0	1	0	2
0	x_6	0	0	0	$\frac{1}{8}$	$-\frac{3}{8}$	1	$-\frac{23}{4}$
		0	0	0	$\frac{7}{8}$	$\frac{3}{8}$	0	$\frac{31}{4}$

6. Use the dual simplex method to verify that Tableau 3.27 is correct,
7. For Example 3, verify using the dual simplex method that the final tableau (Tableau 3.30) is correct. Note that your answer may differ slightly from the text due to round-off error.
8. Use the dual simplex method to find a solution to the linear programming problem formed by adding the constraint

$$3x_1 + 5x_3 \geq 15$$

to the problem in Example 2.

9. Example 3 showed that adding a constraint may change the solution to a linear programming problem (i.e., the new solution has different basic variables and the basic variables have different values). There are two other possibilities that may occur when a constraint is added. Describe them.
10. **Computing project.** Compare the structure diagrams for the simplex algorithm and the dual simplex algorithm. How is duality exemplified by these diagrams?

3.5 THE REVISED SIMPLEX METHOD

The revised simplex method makes use of some of the notation and ideas we developed in Section 3.3 to obtain efficiency in computation and storage of intermediate results. The simplex method, as we described it,

performs elementary row operations on an $m \times (n + m)$ matrix for a problem in standard form with m constraints and n variables. The revised simplex method works with the much smaller $m \times m$ matrix. The savings in computation time and storage of arrays can be considerable for large problems ($n \geq 1000$) typically found in applications. Consequently the computer programs for solving linear programming problems, called LP codes, always use the revised simplex method.

We consider a linear programming problem in canonical form

$$\left.\begin{aligned} &\text{Maximize} \quad \mathbf{z} = \mathbf{c}^{\mathrm{T}}\mathbf{x} \\ &\text{subject to} \\ &\qquad \mathbf{A}\mathbf{x} = \mathbf{b} \\ &\qquad \mathbf{x} \geq \mathbf{0}. \end{aligned}\right\} \tag{1}$$

In this section we confine our attention to the case in which the canonical form above was obtained using only slack variables, not artificial variables. A more general procedure for dealing with artificial variables is available, but will not be discussed in this book.

For each tableau in the simplex algorithm we defined the matrix

$$\mathbf{B} = \left[\mathbf{A}_{i_1} \quad \mathbf{A}_{i_2} \cdots \mathbf{A}_{i_m}\right],$$

where $\mathbf{A}_{i_r}$ is the i_r column of the constraint matrix $\mathbf{A}$ and i_r is the index of the rth basic variable in the list at the left side of the tableau. The values of the basic variables $x_{i_1}, x_{i_2}, \ldots, x_{i_m}$ were represented by the vector

$$\mathbf{x}_{\mathrm{B}} = \begin{bmatrix} x_{i_1} \\ x_{i_2} \\ \vdots \\ x_{i_m} \end{bmatrix},$$

and we showed that

$$\mathbf{x}_{\mathrm{B}} = \mathbf{B}^{-1}\mathbf{b} \quad \text{or} \quad \mathbf{b} = \mathbf{B}\mathbf{x}_{\mathrm{B}}. \tag{2}$$

We also defined

$$\mathbf{c}_{\mathrm{B}} = \begin{bmatrix} c_{i_1} \\ c_{i_2} \\ \vdots \\ c_{i_m} \end{bmatrix},$$

so that

$$z = \mathbf{c}_{\mathrm{B}}^{\mathrm{T}}\mathbf{x}_{\mathrm{B}} \quad \text{or} \quad z - \mathbf{c}_{\mathrm{B}}^{\mathrm{T}}\mathbf{x}_{\mathrm{B}} = \mathbf{0}. \tag{3}$$

We can combine equations (2) and (3) into one matrix equation by writing

$$\begin{bmatrix} 1 & -\mathbf{c}_B^T \\ 0 & \mathbf{B} \end{bmatrix} \begin{bmatrix} z \\ \mathbf{x}_B \end{bmatrix} = \begin{bmatrix} 0 \\ \mathbf{b} \end{bmatrix}. \tag{4}$$

The coefficient matrix in (4) is $(m + 1) \times (m + 1)$, and the vectors are $(m + 1) \times 1$. We denote the coefficient matrix in (4) by $\mathbf{M}$; that is,

$$\mathbf{M} = \begin{bmatrix} 1 & -\mathbf{c}_B^T \\ 0 & \mathbf{B} \end{bmatrix}.$$

By multiplying the matrices together, the reader may verify that

$$\mathbf{M}^{-1} = \begin{bmatrix} 1 & \mathbf{c}_B^T\mathbf{B}^{-1} \\ 0 & \mathbf{B}^{-1} \end{bmatrix}.$$

Hence, the solution represented by any tableau can be succinctly stated as

$$\begin{aligned} \begin{bmatrix} z \\ \mathbf{x}_B \end{bmatrix} &= \mathbf{M}^{-1} \begin{bmatrix} 0 \\ \mathbf{b} \end{bmatrix} \\ &= \begin{bmatrix} 1 & \mathbf{c}_B^T\mathbf{B}^{-1} \\ 0 & \mathbf{B}^{-1} \end{bmatrix} \begin{bmatrix} 0 \\ \mathbf{b} \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{c}_B^T\mathbf{B}^{-1}\mathbf{b} \\ \mathbf{B}^{-1}\mathbf{b} \end{bmatrix}. \end{aligned} \tag{5}$$

The revised simplex method exploits the form of the solution in (5) by working only with the matrix $\mathbf{M}^{-1}$ instead of the entire tableau. In fact, because $\mathbf{M}^{-1}$ has a particularly simple form, we need consider only $\mathbf{B}^{-1}$. Now the initial tableau is constructed so that $\mathbf{B} = \mathbf{I}_m$, where $\mathbf{I}_m$ denotes the $m \times m$ identity matrix. Thus, initially, $\mathbf{B}^{-1} = \mathbf{I}_m$. The revised simplex method uses a procedure to find the $\mathbf{B}^{-1}$ for the next iteration using information about the entering and departing variables along with the current $\mathbf{B}^{-1}$. We start by writing

$$(\mathbf{B}^{-1})_{\text{new}} = \mathbf{E}\mathbf{B}^{-1}, \tag{6}$$

where $\mathbf{E}$ is an $m \times m$ matrix that can be obtained as follows.

(a) Start with $\mathbf{I}_m$. Suppose x_p is the entering variable and that x_{i_q} is the departing variable. We have previously shown that $\mathbf{t}_p$, the pth column of the current simplex tableau, which is the pivotal column, is given by

$$\mathbf{t}_p = \mathbf{B}^{-1}\mathbf{A}_p,$$

where $\mathbf{A}_p$ is the pth column of the original matrix $\mathbf{A}$. Denote the entries of the pivotal column of the current tableau by

$$\mathbf{t}_p = \begin{bmatrix} t_{1p} \\ t_{2p} \\ \vdots \\ t_{mp} \end{bmatrix}.$$

(b) Replace the qth column of $\mathbf{I}_m$ by the vector

$$\begin{bmatrix} -t_{1p}/t_{qp} \\ -t_{2p}/t_{qp} \\ \vdots \\ 1/t_{qp} \\ \vdots \\ -t_{mp}/t_{qp} \end{bmatrix} \leftarrow q\text{th entry},$$

called an **eta vector.**

This modification of the identity matrix that we have constructed is called an **eta matrix** and is the matrix $\mathbf{E}$ that we wanted. Notice that we never have to numerically invert a matrix (a procedure that may require some care); we simply obtain a sequence of matrices that are $\mathbf{B}^{-1}$ for each of our tableaux.

The revised simplex method consists of the following steps:

1. Determine the entering variable x_p by choosing the most negative $z_j - c_j$, $j = 1, 2, \ldots, s$. Pick randomly if there are ties. Recall that $z_j - c_j$ may be computed as

$$z_j - c_j = \mathbf{c}_B^T \mathbf{t}_j - c_j = \mathbf{c}_B^T \mathbf{B}^{-1} \mathbf{A}_j - c_j.$$

In terms of a matrix product, we may write

$$z_j - c_j = \begin{bmatrix} 1 & \mathbf{c}_B^T \mathbf{B}^{-1} \end{bmatrix} \begin{bmatrix} -c_j \\ \mathbf{A}_j \end{bmatrix}.$$

2. Determine the departing variable x_{i_q}. This is the variable with the minimum θ-ratio. The i_rth basic variable has a θ-ratio

$$x_{i_r}/t_{rp},$$

where x_{i_r} is the entry in $\mathbf{x}_B$ on the right-hand side of the tableau and where $t_{rp} > 0$. To find the θ-ratios, we may compute

$$\mathbf{t}_p = \begin{bmatrix} t_{1p} \\ t_{2p} \\ \vdots \\ t_{mp} \end{bmatrix} = \mathbf{B}^{-1}\mathbf{A}_p$$

and

$$\mathbf{x}_B = \mathbf{B}^{-1}\mathbf{b}.$$

We use only those entries in $\mathbf{x}_B$ that correspond to positive entries in $\mathbf{t}_p$ to form the set of θ-ratios.

3. Determine the new $\mathbf{B}^{-1}$ as described above.

4. Determine the new basic feasible solution and objective function value. From Equations (5) and (6), $(\mathbf{x}_B)_{\text{new}} = (\mathbf{B}^{-1})_{\text{new}}\mathbf{b} = \mathbf{E}\mathbf{B}^{-1}\mathbf{b} = \mathbf{E}\mathbf{x}_B$. Thus, if $\mathbf{x}_B$ is available in storage, then

$$(\mathbf{x}_B)_{\text{new}} = \mathbf{E}\mathbf{x}_B.$$

This formula is computationally faster than Equation (5), since $\mathbf{E}$ is sparse (has many zeros).

As in the simplex method, if none of the $z_j - c_j$ is negative, an optimal solution has been achieved. If none of the entries in the pivotal column $\mathbf{t}_p$ is positive, the optimal solution is unbounded. Observe that in using the revised simplex method, no tableau need be computed.

EXAMPLE 1. Consider the linear programming problem in canonical form that came from the model for the sawmill:

$$\begin{aligned} &\text{Maximize} \quad z = 120x_1 + 100x_2 \\ &\text{subject to} \\ &\quad 2x_1 + 2x_2 + x_3 \qquad = 8 \\ &\quad 5x_1 + 3x_2 \qquad + x_4 = 15 \\ &\quad x_j \geq 0 \quad j = 1,2,3,4. \end{aligned}$$

For this problem

$$\mathbf{A} = \begin{bmatrix} 2 & 2 & 1 & 0 \\ 5 & 3 & 0 & 1 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} 8 \\ 15 \end{bmatrix}, \quad \text{and} \quad \mathbf{c} = \begin{bmatrix} 120 \\ 100 \\ 0 \\ 0 \end{bmatrix}.$$

The slack variables x_3 and x_4 are our initial basic variables, so that $i_1 = 3$ and $i_2 = 4$. Consequently,

$$\mathbf{B} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad \mathbf{B}^{-1} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad \text{and} \quad \mathbf{c}_B = \begin{bmatrix} 0 \\ 0 \end{bmatrix}.$$

Therefore,

$$\mathbf{c}_B^T\mathbf{B}^{-1} = [0 \quad 0] \quad \text{and} \quad \mathbf{M}^{-1} = \begin{bmatrix} 1 & \mathbf{c}_B^T\mathbf{B}^{-1} \\ 0 & \mathbf{B}^{-1} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix},$$

so that

$$\begin{bmatrix} z \\ \mathbf{x}_B \end{bmatrix} = \mathbf{M}^{-1}\begin{bmatrix} 0 \\ \mathbf{b} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} 0 \\ 8 \\ 15 \end{bmatrix} = \begin{bmatrix} 0 \\ 8 \\ 15 \end{bmatrix}. \tag{7}$$

We first determine the entering variable by computing

$$z_j - c_j = \begin{bmatrix} 1 & \mathbf{c}_B^T\mathbf{B}^{-1} \end{bmatrix}\begin{bmatrix} -c_j \\ \mathbf{A}_j \end{bmatrix} = [1 \quad 0 \quad 0]\begin{bmatrix} -c_j \\ \mathbf{A}_j \end{bmatrix} = -c_j.$$

Therefore,

$$\begin{aligned} z_1 - c_1 &= -c_1 = -120 \\ z_2 - c_2 &= -c_2 = -100 \\ z_3 - c_3 &= -c_3 = 0 \\ z_4 - c_4 &= -c_4 = 0, \end{aligned}$$

and x_1 ($p = 1$) is the entering variable.

To find the departing variable, we form

$$\mathbf{t}_1 = \mathbf{B}^{-1}\mathbf{A}_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\begin{bmatrix} 2 \\ 5 \end{bmatrix} = \begin{bmatrix} 2 \\ 5 \end{bmatrix}$$

and copy $\mathbf{x}_B$ from (7). All the entries of $\mathbf{t}_1$ are positive, so we compute

$$\min\{\tfrac{8}{2}, \tfrac{15}{5}\} = 3.$$

The minimum θ-ratio occurs for $x_{i_2} = x_4$ ($q = 2$) and, thus, x_4 becomes the departing variable. To compute the eta matrix $\mathbf{E}$, we replace the second column of $\mathbf{I}_2$ with the eta vector

$$\begin{bmatrix} -\frac{2}{5} \\ \frac{1}{5} \end{bmatrix}.$$

Therefore,

$$\mathbf{E} = \begin{bmatrix} 1 & -\frac{2}{5} \\ 0 & \frac{1}{5} \end{bmatrix}$$

and

$$(\mathbf{B}^{-1})_{\text{new}} = \mathbf{E}\mathbf{B}^{-1} = \begin{bmatrix} 1 & -\frac{2}{5} \\ 0 & \frac{1}{5} \end{bmatrix}\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & -\frac{2}{5} \\ 0 & \frac{1}{5} \end{bmatrix}.$$

Now we have $i_1 = 3$ and $i_2 = 1$, so that

$$\mathbf{c}_{\mathrm{B}} = \begin{bmatrix} 0 \\ 120 \end{bmatrix} \quad \text{and} \quad \mathbf{c}_{\mathrm{B}}^{\mathrm{T}}\mathbf{B}^{-1} = [0 \quad 24].$$

Therefore,

$$\mathbf{M}^{-1} = \begin{bmatrix} 1 & 0 & 24 \\ 0 & 1 & -\frac{2}{5} \\ 0 & 0 & \frac{1}{5} \end{bmatrix},$$

and the current solution is

$$\begin{bmatrix} z \\ \mathbf{x}_{\mathrm{B}} \end{bmatrix} = \mathbf{M}^{-1}\begin{bmatrix} 0 \\ \mathbf{b} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 24 \\ 0 & 1 & -\frac{2}{5} \\ 0 & 0 & \frac{1}{5} \end{bmatrix}\begin{bmatrix} 0 \\ 8 \\ 15 \end{bmatrix} = \begin{bmatrix} 360 \\ 2 \\ 3 \end{bmatrix}. \tag{8}$$

Next we compute

$$z_j - c_j = \begin{bmatrix} 1 & \mathbf{c}_{\mathrm{B}}^{\mathrm{T}}\mathbf{B}^{-1} \end{bmatrix}\begin{bmatrix} -c_j \\ \mathbf{A}_j \end{bmatrix} = [1 \quad 0 \quad 24]\begin{bmatrix} -c_j \\ \mathbf{A}_j \end{bmatrix},$$

so that

$$z_1 - c_1 = [1 \quad 0 \quad 24]\begin{bmatrix} -120 \\ 2 \\ 5 \end{bmatrix} = 0$$

$$z_2 - c_2 = [1 \quad 0 \quad 24]\begin{bmatrix} -100 \\ 2 \\ 3 \end{bmatrix} = -28$$

$$z_3 - c_3 = [1 \quad 0 \quad 24]\begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} = 0$$

$$z_4 - c_4 = [1 \quad 0 \quad 24]\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = 24.$$

The entering variable is x_2 ($p = 2$). To find the departing variable, we compute

$$\mathbf{t}_2 = \mathbf{B}^{-1}\mathbf{A}_2 = \begin{bmatrix} 1 & -\frac{2}{5} \\ 0 & \frac{1}{5} \end{bmatrix}\begin{bmatrix} 2 \\ 3 \end{bmatrix} = \begin{bmatrix} \frac{4}{5} \\ \frac{3}{5} \end{bmatrix}$$

and copy $\mathbf{x}_B$ from (8). All the entries of $\mathbf{t}_2$ are positive, so that we compute

$$\min\left\{\frac{2}{\frac{4}{5}}, \frac{3}{\frac{3}{5}}\right\} = \frac{2}{\frac{4}{5}} = \frac{5}{2}.$$

The minimum θ-ratio occurs for $x_{i_1} = x_3$ $(q = 1)$ and, thus, x_3 becomes the departing variable. We compute the eta matrix $\mathbf{E}$ by replacing the first column of $\mathbf{I}_2$ by the eta vector

$$\begin{bmatrix} 1/\frac{4}{5} \\ -\frac{3}{5}/\frac{4}{5} \end{bmatrix} = \begin{bmatrix} \frac{5}{4} \\ -\frac{3}{4} \end{bmatrix}.$$

We obtain

$$\mathbf{E} = \begin{bmatrix} \frac{5}{4} & 0 \\ -\frac{3}{4} & 1 \end{bmatrix}$$

and

$$(\mathbf{B}^{-1})_{\text{new}} = \begin{bmatrix} \frac{5}{4} & 0 \\ -\frac{3}{4} & 1 \end{bmatrix}\begin{bmatrix} 1 & -\frac{2}{5} \\ 0 & \frac{1}{5} \end{bmatrix} = \begin{bmatrix} \frac{5}{4} & -\frac{1}{2} \\ -\frac{3}{4} & \frac{1}{2} \end{bmatrix}.$$

Now we have $i_1 = 2$ and $i_2 = 1$, so that

$$\mathbf{c}_B = \begin{bmatrix} 100 \\ 120 \end{bmatrix} \quad \text{and} \quad \mathbf{c}_B^T\mathbf{B}^{-1} = [35 \quad 10].$$

Therefore,

$$\mathbf{M}^{-1} = \begin{bmatrix} 1 & 35 & 10 \\ 0 & \frac{5}{4} & -\frac{1}{2} \\ 0 & -\frac{3}{4} & \frac{1}{2} \end{bmatrix},$$

and the current solution is

$$\begin{bmatrix} z \\ \mathbf{x}_B \end{bmatrix} = \begin{bmatrix} 1 & 35 & 10 \\ 0 & \frac{5}{4} & -\frac{1}{2} \\ 0 & -\frac{3}{4} & \frac{1}{2} \end{bmatrix}\begin{bmatrix} 0 \\ 8 \\ 15 \end{bmatrix} = \begin{bmatrix} 430 \\ \frac{5}{2} \\ \frac{3}{2} \end{bmatrix}. \tag{9}$$

Next we compute

$$z_j - c_j = [1 \quad 35 \quad 10]\begin{bmatrix} -c_j \\ \mathbf{A}_j \end{bmatrix}.$$

We have

$$\begin{aligned} z_1 - c_1 &= 0 \\ z_2 - c_2 &= 0 \\ z_3 - c_3 &= 35 \\ z_4 - c_4 &= 10. \end{aligned}$$

Since $z_j - c_j \geq 0$ for all j, the solution given by (9) is optimal. Δ

3.5 EXERCISES

In Exercises 1 and 2 calculate the eta vector and then the eta matrix **E** from the given information.

1. The pivotal column $\mathbf{t}_p$ is

$$\mathbf{t}_p = \begin{bmatrix} 1 \\ -2 \\ 0 \\ 3 \end{bmatrix}$$

and the departing variable labels the fourth row ($q = 4$).

2. The pivotal column $\mathbf{t}_p$ is

$$\mathbf{t}_p = \begin{bmatrix} \frac{1}{2} \\ \frac{3}{4} \\ -\frac{1}{3} \\ 0 \\ \frac{5}{12} \end{bmatrix}$$

and the departing variable labels the second row ($q = 2$).

In Exercises 3 and 4, find the new $\mathbf{B}^{-1}$ from the given information.

3. The current $\mathbf{B}^{-1}$ is

$$\mathbf{B}^{-1} = \begin{bmatrix} 1 & 0 & 2 \\ -1 & 1 & 3 \\ 0 & 2 & 1 \end{bmatrix}$$

and the pivotal column is given by

$$\mathbf{t}_p = \begin{bmatrix} 1 \\ \frac{3}{2} \\ 2 \end{bmatrix} \quad \text{and} \quad \mathbf{x}_\mathbf{b} = \begin{bmatrix} 1 \\ 3 \\ 4 \end{bmatrix}.$$

4. The current $\mathbf{B}^{-1}$ is

$$\mathbf{B}^{-1} = \begin{bmatrix} 1 & -1 & 2 & 1 \\ 0 & 1 & 0 & 1 \\ -1 & 3 & 1 & -3 \\ 2 & -1 & 2 & 4 \end{bmatrix}$$

and the pivotal column is given by

$$\mathbf{t}_p = \begin{bmatrix} 0 \\ 4 \\ -2 \\ \frac{2}{3} \end{bmatrix} \quad \text{and} \quad \mathbf{x}_\mathbf{b} = \begin{bmatrix} 1 \\ 1 \\ 4 \\ 2 \end{bmatrix}.$$

In Exercises 5–9 solve the given linear programming problem using the revised simplex method.

5. Exercise 6, Section 3.3

6. Maximize $z = x_1 + 2x_2 + 3x_3 + x_4$
subject to
$$\begin{aligned} 2x_1 + x_2 + x_3 + 2x_4 &\le 18 \\ 3x_1 + 5x_2 + 2x_3 + 3x_4 &\le 24 \\ 3x_1 + 2x_2 + x_3 + x_4 &\le 12 \\ x_j \ge 0, \quad j &= 1,2,3,4. \end{aligned}$$

7. Maximize $z = 2x_1 + 3x_2 + x_3 + x_4 + 2x_5$
subject to
$$\begin{aligned} 2x_1 + x_2 - 3x_3 + x_4 + x_5 &\le 10 \\ x_1 + 4x_2 + x_4 + 2x_5 &\le 20 \\ 3x_1 + 4x_4 + 2x_5 &\le 15 \\ x_j \ge 0, \quad j &= 1,2,\ldots,5. \end{aligned}$$

8. Maximize $z = 3x_1 + 2x_2 + 4x_5 + x_6 + 2x_8$
subject to
$$\begin{aligned} 3x_1 + x_2 + x_3 + x_4 + 2x_5 + 3x_6 + x_8 &\le 12 \\ 2x_1 + x_2 + 2x_4 + 5x_6 + x_7 + 2x_8 &\le 15 \\ 3x_1 + 2x_2 + x_3 + 3x_5 + x_7 + 3x_8 &\le 18 \\ x_j \ge 0, \quad j &= 1,2,\ldots,8. \end{aligned}$$

9. Maximize $z = 2x_1 + x_2 + 3x_3 + x_6 + 2x_7 + 3x_8$
subject to
$$\begin{aligned} 2x_1 + x_2 + x_4 + 3x_5 + x_7 &\le 24 \\ x_1 + 3x_3 + x_4 + x_5 + 2x_6 + 3x_8 &\le 30 \\ 5x_1 + 3x_2 + 3x_4 + 2x_5 + x_7 + 5x_8 &\le 18 \\ 3x_1 + 2x_2 + x_3 + x_6 + 3x_8 &\le 20 \\ x_j \ge 0, \quad j &= 1,2,\ldots,8. \end{aligned}$$

10. If
$$\mathbf{M} = \begin{bmatrix} 1 & -\mathbf{c}_B^T \\ 0 & \mathbf{B} \end{bmatrix},$$
verify that
$$\mathbf{M}^{-1} = \begin{bmatrix} 1 & \mathbf{c}_B^T\mathbf{B}^{-1} \\ 0 & \mathbf{B}^{-1} \end{bmatrix}.$$

11. Consider the standard linear programming problem
Maximize $z = \mathbf{c}^T\mathbf{x}$
subject to

$$\mathbf{Ax} \leq \mathbf{b}$$
$$\mathbf{x} \geq \mathbf{0}.$$

(a) Show that this problem can be written in canonical form as
Maximize $z = \mathbf{c}^T\mathbf{x} + (\mathbf{c}')^T\mathbf{x}'$
subject to

$$[\mathbf{A} \mid \mathbf{I}]\begin{bmatrix}\mathbf{x}\\ \mathbf{x}'\end{bmatrix} = \mathbf{b}$$
$$\begin{bmatrix}\mathbf{x}\\ \mathbf{x}'\end{bmatrix} \geq \mathbf{0}.$$

(*Hint:* $\mathbf{x}'$ will be a vector of slack variables.)

(b) Show that the initial simplex tableau represents the matrix equation

$$\begin{bmatrix}1 & -\mathbf{c}^T & -(\mathbf{c}')^T\\ 0 & \mathbf{A} & \mathbf{I}\end{bmatrix}\begin{bmatrix}z\\ \mathbf{x}\\ \mathbf{x}'\end{bmatrix} = \begin{bmatrix}0\\ \mathbf{b}\end{bmatrix}.$$

(*Hint:* Proceed as in the derivation of Equation (4).)

(c) Equation (5) shows that the solution represented by any tableau is obtained by multiplying the vector

$$\begin{bmatrix}0\\ \mathbf{b}\end{bmatrix}$$

by $\mathbf{M}^{-1}$. Show that the system of equations represented by any tableau is

$$\begin{bmatrix}1 & \mathbf{c}_B^T\mathbf{B}^{-1}\mathbf{A} - \mathbf{c}^T & \mathbf{c}_B^T\mathbf{B}^{-1} - (\mathbf{c}')^T\\ 0 & \mathbf{B}^{-1}\mathbf{A} & \mathbf{B}^{-1}\end{bmatrix}\begin{bmatrix}z\\ \mathbf{x}\\ \mathbf{x}'\end{bmatrix} = \begin{bmatrix}\mathbf{c}_B^T\mathbf{B}^{-1}\mathbf{b}\\ \mathbf{B}^{-1}\mathbf{b}\end{bmatrix}.$$

(d) Show from part (c) that, at an optimal solution to the problem in part (a), $\mathbf{c}_B^T\mathbf{B}^{-1}\mathbf{A} \geq \mathbf{c}^T$ and $\mathbf{c}_B^T\mathbf{B}^{-1} \geq (\mathbf{c}')^T$.

12. (a) Find the dual of the linear programming problem in Exercise 11(a).
(b) Show that $\mathbf{w} = (\mathbf{B}^{-1})^T\mathbf{c}_B$ is a feasible solution to the dual problem. (*Hint:* Use Exercise 11d.)
(c) Using Exercise 9 in Section 3.3, explain why $\mathbf{w} = (\mathbf{B}^{-1})^T\mathbf{c}_B$ is an optimal solution to the dual problem.

13. **Computing project.** Construct a flowchart or structure diagram for the revised simplex method.

3.6 SENSITIVITY ANALYSIS

In the Prologue it was pointed out that solving a linear programming problem is just one part of mathematically modeling a situation. After the

problem is solved, one must ask whether the solution makes sense in the actual situation. It is also very likely that the numbers that are used for the linear programming problem are not known exactly. In most cases they will be estimates of the true numbers, and many times they will not be very good estimates. Consequently, it is desirable to have ways of measuring the sensitivity of the solution to changes in the values that specify the problem. Of course, one way to proceed would be to recompute the solution using different values. However, the tableau representing an optimal solution contains the information we need to measure the sensitivity. Using the final tableau makes it unnecessary to repeat the calculations for a different set of values.

There are five things that can be singled out as subject to variation in defining a linear programming problem.

1. One or more of the objective function coefficients can change.
2. One or more of the resource values (components of **b**) can change.
3. One or more of the entries in the constraint matrix **A** can change.
4. A variable might need to be added. This may happen if management wants information about the effects of making an additional product.
5. Addition of a constraint might be necessary, especially if the solution to the original problem is somewhat unreasonable. This situation also occurs when some of the variables are constrained to take integer values.

In this section we examine simple cases of the first two possibilities, namely, when only one quantity is allowed to change. In Chapter 4 we will discuss the fifth case. Cases 3 and 4 and the situation in Cases 1 and 2 in which more than one quantity changes simultaneously are discussed in more advanced texts (Further Reading).

Another approach to this study is to assume a change in each entry of the objective function coefficient vector, for example, and to assume that the amount of the change depends on a parameter. This leads to the study of *parametric programming*. We do not study this topic here but refer the interested reader to Further Reading.

We assume that the original problem has been converted to the form

$$\begin{aligned} &\text{Maximize} \quad z = \mathbf{c}^T\mathbf{x} \\ &\text{subject to} \\ &\qquad \mathbf{Ax} = \mathbf{b} \\ &\qquad \mathbf{x} \geq \mathbf{0} \end{aligned}$$

and that an optimal solution to the problem has been obtained. We further assume that we have available the final tableau for the simplex method.

As changes are made in the problem statement, there are several things that may happen to the old optimal solution. It may remain both feasible and optimal, so that no further calculations are necessary. In fact, some computer codes for linear programming problems will automatically com-

pute a range of values for $\mathbf{b}$ and $\mathbf{c}$ in which the solution found will remain optimal. The solution to the problem may remain feasible but become nonoptimal. In this case a few iterations of the simplex algorithm will restore the optimality. The optimal solution to the original problem, being a basic feasible solution, provides an initial basic feasible solution for these iterations. On the other hand, the solution to the given problem may remain optimal, as judged by the optimality criterion, but may become infeasible. In this case a few iterations of the dual simplex algorithm will usually restore the feasibility. We now examine Cases 1 and 2.

Change in the Objective Function

Suppose c_k changes to $\hat{c}_k = c_k + \Delta c_k$. The old optimal solution must remain feasible, since neither $\mathbf{A}$ nor $\mathbf{b}$ was changed. The optimality criterion is stated in terms of

$$z_j - c_j = \mathbf{c}_B^T \mathbf{t}_j - c_j,$$

where $\mathbf{t}_j$ is the jth column of the final tableau (see Section 3.3). If k is the index of one of the basic variables, then $\mathbf{c}_B$ changes and every $z_j - c_j$ must be recomputed. If x_k is a nonbasic variable, $\mathbf{c}_B$ is unchanged and only $z_k - c_k$ changes. In this latter case we have

$$z_k - \hat{c}_k = (z_k - c_k) - \Delta c_k.$$

Therefore,

$$z_k - \hat{c}_k \geq 0$$

if and only if

$$z_k - c_k \geq \Delta c_k. \tag{1}$$

That is, the profit coefficient of the kth variable, c_k, can be increased by as much as $z_k - c_k$ and the solution will remain optimal. However, making this increase in c_k will not change the value of the objective function, since $x_k = 0$ in the optimal solution.

Now suppose that x_k is a basic variable in the optimal solution. Suppose $k = i_r$, so that the new value of $\mathbf{c}_B$ is

$$\hat{\mathbf{c}}_B = \begin{bmatrix} c_{i_1} \\ c_{i_2} \\ \vdots \\ c_{i_r} + \Delta c_{i_r} \\ \vdots \\ c_{i_m} \end{bmatrix} = \mathbf{c}_B + \Delta c_{i_r} \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{bmatrix} \leftarrow r\text{th entry}.$$

Let $\mathbf{e}_r$ denote the $m \times 1$ matrix which is all zeros except for a 1 in the rth row. Then we may write

$$\hat{\mathbf{c}}_B = \mathbf{c}_B + \Delta c_{i_r} \mathbf{e}_r.$$

Now for all values of j except $j = i_r$, we have

$$\hat{z}_j - c_j = \hat{\mathbf{c}}_B^T \mathbf{t}_j - c_j = \mathbf{c}_B^T \mathbf{t}_j + \Delta c_{i_r} \mathbf{e}_r^T \mathbf{t}_j - c_j = z_j - c_j + t_{rj} \Delta c_{i_r}. \quad (2)$$

The reader may show, using a similar argument, that

$$\hat{z}_{i_r} - \hat{c}_{i_r} = 0.$$

Recall that basic variables are determined only by the constraints, not by the objective function. Furthermore, for each basic variable x_{i_k} we must have $\hat{z}_{i_k} - c_{i_k} = 0$. This follows from (2), since $t_{ri_k} = 0$ when $k \neq r$. Consequently, the old optimal solution remains optimal when the objective function coefficient of a basic variable is changed if and only if for all nonbasic variables x_j,

$$z_j - c_j + t_{rj} \Delta c_{i_r} \geq 0$$

or

$$z_j - c_j \geq -t_{rj} \Delta c_{i_r}. \quad (3)$$

If $t_{rj} = 0$, the inequality in (3) holds for all changes Δc_{i_r}. If $t_{rj} > 0$, we can divide each side of (3) by $-t_{rj}$, reversing the inequality, to obtain

$$\Delta c_{i_r} \geq -\frac{z_j - c_j}{t_{rj}} \quad (4)$$

for those j for which x_j is nonbasic and $t_{rj} > 0$. If $t_{rj} < 0$, we again divide both sides of (3) by $-t_{rj}$, but do not reverse the inequality, and obtain

$$\Delta c_{i_r} \leq -\frac{z_j - c_j}{t_{rj}} \quad (5)$$

for those j for which x_j is nonbasic and $t_{rj} < 0$. Combining (4) and (5), we find that the old optimal solution remains optimal if the change in c_{i_r} satisfies

$$\max_j \left\{ -\frac{z_j - c_j}{t_{rj}} \middle| t_{rj} > 0 \right\} \leq \Delta c_{i_r} \leq \min_j \left\{ -\frac{z_j - c_j}{t_{rj}} \middle| t_{rj} < 0 \right\}, \quad (6)$$

where the index j runs over all nonbasic variables. If there are no j such that $t_{rj} > 0$, then the left side of (6) gives no restriction on Δc_{i_r}. It may take on arbitrarily large negative values. Likewise, if there are no j such that $t_{rj} < 0$, the right side of (6) gives no restriction on Δc_{i_r}.

If Δc_k does not satisfy the inequality in (1) when x_k is a nonbasic variable or if Δc_k does not satisfy the inequalities in (6) when x_k is a basic variable, the solution represented by the final tableau is no longer optimal. Some iterations of the simplex method will restore optimality.

EXAMPLE 1. Consider the linear programming problem from Example 2 in Section 2.3:

$$\text{Maximize} \quad z = x_1 - 2x_2 - 3x_3 - x_4 - x_5 + 2x_6$$

subject to

$$\begin{aligned} x_1 + 2x_2 + 2x_3 + x_4 + x_5 &= 12 \\ x_1 + 2x_2 + x_3 + x_4 + 2x_5 + x_6 &= 18 \\ 3x_1 + 6x_2 + 2x_3 + x_4 + 3x_5 &= 24 \\ x_j \geq 0, \quad j &= 1,2,\ldots,6. \end{aligned}$$

An optimal solution is given by Tableau 3.31.

Since x_2, x_4, and x_5 are nonbasic variables, we use (1) and see that c_2 can be increased by $4 = z_2 - c_2$, c_4 can be increased by $\frac{1}{2} = z_4 - c_4$, or c_5 can be increased by $4 = z_5 - c_5$ without changing the optimality of the solution.

Tableau 3.31

c_B		1 x_1	−2 x_2	−3 x_3	−1 x_4	−1 x_5	2 x_6	x_B
−3	x_3	0	0	1	$\frac{1}{2}$	0	0	3
2	x_6	0	0	0	$\frac{1}{2}$	1	1	9
1	x_1	1	2	0	0	1	0	6
		0	4	0	$\frac{1}{2}$	4	0	15

We now examine the objective function coefficients of the basic variables x_1, x_3, and x_6. To determine the range for a change in c_1, we compute

$$\max\left\{-\frac{z_j - c_j}{t_{3j}} \,\middle|\, t_{3j} > 0\right\} = \max\{-\tfrac{4}{2}, -\tfrac{4}{1}\} = -2$$

and

$$\min\left\{-\frac{z_j - c_j}{t_{3j}} \,\middle|\, t_{3j} < 0\right\}.$$

Since there are no j for which $t_{3j} < 0$, there is no upper bound for the change in c_1. That is, if

$$-2 \leq \Delta c_1 < \infty,$$

the current solution remains optimal.

To determine the range for a change in c_3 for which the solution remains optimal, we compute

$$\max\left\{ -\frac{z_j - c_j}{t_{1j}} \middle| t_{1j} > 0 \right\} = \max\left\{ \frac{-\frac{1}{2}}{\frac{1}{2}} \right\} = -1.$$

Again there are no values of j for which $t_{1j} < 0$, so that the change in c_3 is not bounded above. The solution remains optimal if

$$-1 \leq \Delta c_3 < \infty.$$

Finally, checking the range for a change in c_6, we see that the solution remains optimal if

$$-1 \leq \Delta c_6 < \infty.$$

Summarizing our results, the optimal solution

$$x_1 = 6, \quad x_2 = 0, \quad x_3 = 3, \quad x_4 = 0, \quad x_5 = 0, \quad x_6 = 9$$

remains optimal for a change, Δc_k, in the kth coefficient of the objective function if Δc_k satisfies the appropriate inequality listed in Table 3.4. This assumes that only one coefficient is changed.

TABLE 3.4

k	Δc_k	k	Δc_k
1	$-2 \leq \Delta c_1 < \infty$	4	$-\infty < \Delta c_4 \leq \frac{1}{2}$
2	$-\infty < \Delta c_2 \leq 4$	5	$-\infty < \Delta c_5 \leq 4$
3	$-1 \leq \Delta c_3 < \infty$	6	$-1 \leq \Delta c_6 < \infty$

△

EXAMPLE 2. The canonical form of the sawmill problem (Example 3, Section 1.2)

$$\begin{aligned} &\text{Maximize} \quad z = 120x_1 + 100x_2 \\ &\text{subject to} \\ &2x_1 + 2x_2 + x_3 \qquad = 8 \\ &5x_1 + 3x_2 \qquad + x_4 = 15 \\ &x_1 \geq 0, \quad x_2 \geq 0, \end{aligned}$$

has a final tableau as shown in Tableau 3.32.

Tableau 3.32

$\mathbf{c}_B$		120 x_1	100 x_2	0 x_3	0 x_4	$\mathbf{x}_B$
100	x_2	0	1	$\frac{5}{4}$	$-\frac{1}{2}$	$\frac{5}{2}$
120	x_1	1	0	$-\frac{3}{4}$	$\frac{1}{2}$	$\frac{3}{2}$
		0	0	35	10	430

The owner of the sawmill would like to know how much it is possible to increase the price (and hence the profit) on each of the outputs individually and still make the same amount of each type of board.

The change in c_1 that does not affect optimality is

$$\frac{-10}{\frac{1}{2}} = -20 \leq \Delta c_1 \leq \frac{-35}{-\frac{3}{4}} = \frac{140}{3}.$$

Likewise, the bounds for the change in c_2 are

$$\frac{-35}{\frac{5}{4}} = -28 \leq \Delta c_2 \leq \frac{-10}{-\frac{1}{2}} = 20.$$

These computations show that, if the owner increases prices so that the profit on 1000 board feet of finish-grade lumber is $\$166\frac{2}{3}$ (instead of \$120) or so that the profit on the same amount of construction-grade lumber is \$120 (instead of \$100), then the owner can still make 1500 board feet of finish-grade and 2500 board feet of construction-grade lumber to maximize the profit. If the owner chooses to increase c_2 to \$120, the final tableau will be Tableau 3.33 and the profit will be \$480.

Tableau 3.33

$\mathbf{c}_B$		$166\frac{2}{3}$ x_1	100 x_2	0 x_3	0 x_4	$\mathbf{x}_B$
100	x_2	0	1	$\frac{5}{4}$	$-\frac{1}{2}$	$\frac{5}{2}$
$166\frac{2}{3}$	x_1	1	0	$-\frac{3}{4}$	$\frac{1}{2}$	$\frac{3}{2}$
		0	0	60	0	480

△

Changes in the Resource Vector

Suppose that b_k changes to $\hat{b}_k = b_k + \Delta b_k$. The old optimality criterion does not depend on $\mathbf{b}$. However, the old solution may no longer be feasible

because the values of the basic variables may change. To compute their new values, we proceed as follows. Let

$$\hat{\mathbf{b}} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_k + \Delta b_k \\ \vdots \\ b_m \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_k \\ \vdots \\ b_m \end{bmatrix} + \Delta b_k \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{bmatrix} \leftarrow k\text{th entry.}$$

We may write $\hat{\mathbf{b}} = \mathbf{b} + \Delta b_k \mathbf{e}_k$. Then

$$\hat{\mathbf{x}}_B = \mathbf{B}^{-1}\hat{\mathbf{b}} = \mathbf{B}^{-1}\mathbf{b} + \Delta b_k \mathbf{B}^{-1}\mathbf{e}_k$$

and

$$\hat{\mathbf{x}}_B = \mathbf{x}_B + \Delta b_k \mathbf{B}^{-1}\mathbf{e}_k. \tag{7}$$

Now $\mathbf{B}^{-1}\mathbf{e}_k$ is the kth column of the matrix $\mathbf{B}^{-1}$. This column appears in the final simplex tableau in the column that held the kth column of the identity matrix in the initial tableau. If the new solution $\hat{\mathbf{x}}_B$ is to be feasible, then Δb_k must be chosen so that

$$\mathbf{x}_B + \Delta b_k \mathbf{B}^{-1}\mathbf{e}_k \geq \mathbf{0}.$$

EXAMPLE 3. Looking again at the sawmill problem, we see that a change, Δb_1, in the availability of the saw yields

$$\hat{\mathbf{x}}_B = \begin{bmatrix} \frac{5}{2} \\ \frac{3}{2} \end{bmatrix} + \Delta b_1 \begin{bmatrix} \frac{5}{4} \\ -\frac{3}{4} \end{bmatrix}.$$

If $\hat{\mathbf{x}}_B$ is to remain feasible, Δb_1 must satisfy

$$\tfrac{5}{2} + \tfrac{5}{4}\Delta b_1 \geq 0$$
$$\tfrac{3}{2} - \tfrac{3}{4}\Delta b_1 \geq 0$$

or

$$-2 \leq \Delta b_1 \leq 2. \tag{8}$$

Likewise, we find that, if the change Δb_2 in the available hours of the plane satisfies

$$-3 \leq \Delta b_2 \leq 5,$$

the solution remains feasible (and optimal). On the other hand, if Δb_1 does not satisfy (8), the old solution becomes infeasible. For example, if the availability of the saw is increased from 8 to 12 hr a week ($\Delta b_1 = 4$), then using (7)

$$\hat{\mathbf{x}}_B = \begin{bmatrix} \frac{5}{2} \\ \frac{3}{2} \end{bmatrix} + 4 \begin{bmatrix} \frac{5}{4} \\ -\frac{3}{4} \end{bmatrix} = \begin{bmatrix} \frac{15}{2} \\ -\frac{3}{2} \end{bmatrix}.$$

Inserting $\hat{\mathbf{x}}_B$ into Tableau 3.32, we obtain Tableau 3.34, which represents an infeasible solution that satisfies the optimality criterion. By applying the dual simplex method, we may restore feasibility. We then obtain Tableau 3.35.

Tableau 3.34

				↓		
		120	100	0	0	
$\mathbf{c}_B$		x_1	x_2	x_3	x_4	$\mathbf{x}_B$
100	x_2	0	1	$\frac{5}{4}$	$-\frac{1}{2}$	$\frac{15}{2}$
← 120	x_1	1	0	$(-\frac{3}{4})$	$\frac{1}{2}$	$-\frac{3}{2}$
		0	0	35	10	570

Tableau 3.35

		120	100	0	0	
$\mathbf{c}_B$		x_1	x_2	x_3	x_4	$\mathbf{x}_B$
100	x_2	$\frac{5}{3}$	1	0	$\frac{1}{3}$	5
0	x_3	$-\frac{4}{3}$	0	1	$-\frac{2}{3}$	2
		$\frac{140}{3}$	0	0	$\frac{100}{3}$	500

We see that the mill operator should make only construction-grade lumber if the saw is available 12 hr per day. Five thousand board feet can be made with this much sawing time for a profit of \$500. △

3.6 EXERCISES

1. Consider the linear programming problem

$$\text{Maximize} \quad z = x_1 + 2x_2 + x_3 + x_4$$

subject to

$$\begin{aligned} 2x_1 + x_2 + 3x_3 + x_4 &\le 8 \\ 2x_1 + 3x_2 + 4x_4 &\le 12 \\ 3x_1 + x_2 + 2x_3 &\le 18 \\ x_j \ge 0 \quad 1 \le j \le 4. & \end{aligned}$$

After adding slack variables x_5, x_6, and x_7 and solving by the simplex method, we obtain the final tableau shown below.

$\mathbf{c}_B$		1 x_1	2 x_2	1 x_3	1 x_4	0 x_5	0 x_6	0 x_7	$\mathbf{x}_B$
1	x_3	$\frac{4}{9}$	0	1	$-\frac{1}{9}$	$\frac{1}{3}$	$-\frac{1}{9}$	0	$\frac{4}{3}$
2	x_2	$\frac{2}{3}$	1	0	$\frac{4}{3}$	0	$\frac{1}{3}$	0	4
0	x_7	$\frac{13}{9}$	0	0	$-\frac{10}{9}$	$-\frac{2}{3}$	$-\frac{1}{9}$	1	$\frac{34}{3}$
		$\frac{7}{9}$	0	0	$\frac{14}{9}$	$\frac{1}{3}$	$\frac{5}{9}$	0	$\frac{28}{3}$

(a) For each of the cost coefficients c_j, $1 \leq j \leq 4$, find the range of values for Δc_j for which the above solution remains optimal.
(b) For each of the resources b_i, $1 \leq i \leq 3$, find the range of values for Δb_i for which the above solution remains feasible.

2. What will be an optimal solution to the problem in Exercise 1
(a) if c_1 is changed to 3?
(b) if b_2 is changed to 26?
(c) if c_3 is changed to $\frac{1}{2}$?
(d) if b_3 is changed to 127?

3. Resolve the linear programming problem in Example 2, Section 2.3, keeping the columns for the artificial variables when forming Tableau 2.20.
(a) For each of the cost coefficients c_j, $1 \leq j \leq 6$, find the range of values for Δc_j for which the solution remains optimal.
(b) For each of the resources b_i, $1 \leq i \leq 3$, find the range of values for Δb_i for which the solution remains feasible.

4. What will be an optimal solution to the problem in Exercise 3
(a) if c_2 is changed to 5?
(b) if c_3 is changed to $-\frac{7}{2}$?
(c) if b_1 is changed to 30?
(d) if b_2 is changed to 25?

5. Consider the agricultural problem in Exercise 4, Section 1.1.
(a) Suppose the farmer is able to hire additional workers who can devote 60 hr to working the crops. How much of each crop should be planted?
(b) Suppose the farmer decides to only plant 10 acres. How much of each crop should be planted?
(c) Suppose the price received for oats increases by \$1/acre, so that the profit per acre of oats is now \$21. How much of each crop should the farmer plant?
(d) What increase in profit for soybeans would induce the farmer to plant soybeans?

6. Consider the investment problem in Exercise 8, Section 1.1.
(a) Suppose the rate of return on the electronics stock increases to 5.2%. What is the optimal investment policy?

(b) Suppose the rate of return on the utility stock decreases to 7%. What is the optimal investment policy?

(c) Suppose the amount invested in the bond is at least \$90,000. What is the optimal investment policy?

7. The text discusses changing only one component of the resource vector at a time. Consider now the situation in which

$$\hat{\mathbf{b}} = \mathbf{b} + \Delta\mathbf{b} \quad \text{and} \quad \Delta\mathbf{b} = \begin{bmatrix} \Delta b_1 \\ \Delta b_2 \\ \vdots \\ \Delta b_m \end{bmatrix}.$$

That is, several components of the resource vector are changed. Following the text discussion, show that

$$\hat{\mathbf{x}}_B = \mathbf{x}_B + \mathbf{B}^{-1}\Delta\mathbf{b}$$

and that the solution $\hat{\mathbf{x}}_B$ is a feasible solution (and hence optimal) if and only if $\mathbf{x}_B + \mathbf{B}^{-1}\Delta\mathbf{b} \geq \mathbf{0}$.

3.6 PROJECT

A tractor manufacturer in a developing nation subcontracts the task of making air filters for the tractors to a small company. The filter consists of a cylindrical main chamber with a cylindrical exit duct mounted on top of it. The specifications for the filters are as follow.

1. In order to keep the dust efficiency within permissible limits, the diameter of the main chamber and the exit duct should not exceed 16 and 6.5 cm, respectively.
2. To keep the pressure drop across the air cleaner small enough to prevent excessive power loss, the main chamber diameter and exit duct diameter should not be less than 9.5 and 3.5 cm, respectively.
3. The main chamber is to be 24 cm tall, and the exit duct is to be 6.5 cm long.
4. To maintain acceptable weight and durability, each filter must contain at least 1600 cm^2 of metal.
5. At least 50 air filters must be supplied each month.

As is typical in such countries, industrial materials such as sheet metal are not available in unlimited supply. The government has allocated 9.65 m^2 of metal each month to the filter manufacturer.

A cross-sectional view of the filter is shown in Figure 3.6. Assume that the intake port and other interior structures need 40% of the total metal used for the main chamber and exit duct. Also assume that unusable scrap accounts for 15% of the total metal used for the main chamber and exit duct.

(a) Set up a linear programming model for this situation to meet the objective of minimizing the amount of sheet metal used per air cleaner.

(b) Solve the model.

(c) Perform a sensitivity analysis on the model.

(d) Comment on the solution, noting the meanings of the slack variables, the dual variables, and the ranges for "resource" and "cost" components.

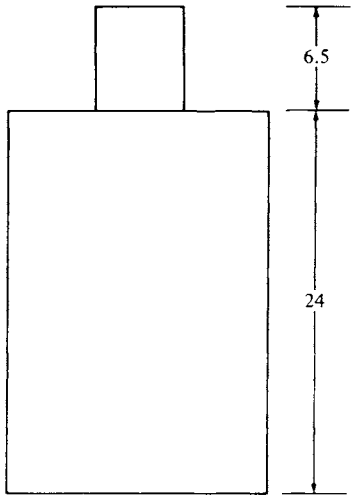


FIGURE 3.6

3.7 COMPUTER ASPECTS (OPTIONAL)

The development of linear programming and its applications has paralleled the development of the digital computer. The first linear programming problem to be solved on a computer dealt with a U.S. Air Force problem regarding the development and support of aircraft subject to strategic and physical requirements. It was solved in January 1952, on a machine at the National Bureau of Standards, now called the National Institute of Standards and Technology. The large computer programs (called **LP codes** or **LP systems**) that are available today to solve linear programming problems owe their existence to the current generation of computers with their extremely large auxiliary storage disks and their fast arithmetic operations. These large LP codes can theoretically handle problems involving as many as 5 million variables and at least 32,000 constraints in reasonable (but not short) amounts of time.

An LP system is typically available from a computer manufacturer as an option when leasing a large system, or it can be leased from one of many software development companies. Such a system may represent an investment of 10–20 person-years of development time. Typical of the commercially available systems are IBM's Mathematical Programming System Extended, MPSX, and Optimization Subroutine Library, OSL; Bonner and Moore Software Systems' Functional Mathematical Programming System, FMPS; Ketron Management Science's Mathematical Programming System III, MPSIII; and CPLEX Optimization's CPLEX.

Personal computers are now very popular for solving linear programming problems. Problems with up to 32,000 constraints and as many as

100,000 variables have been solved on such computers. Typical of PC systems are Lindo System's Lindo and Ketron Management Science's MPSIII/PC.

Two new tools for solving LP problems are modeling languages such as GAMS and LINGO and spreadsheet-based LP packages such as What's Best!. The major advantage of modeling languages is that they allow one to develop and state models much more compactly and faster than older systems, which describe an LP problem by presenting all of the coefficients in tabular form.

Spreadsheet-based systems have the main advantage of getting the results directly in ready-to-use form in the spreadsheet. Also, for many LP problems it is much faster to formulate the model in the spreadsheet than in old-fashioned tabular form. Further details are provided in Appendix B.

In this section we will describe typical features of an LP code to give the reader some idea of what to look for in the course of solving a linear programming problem. The following example will be used as an illustration.

EXAMPLE 1. A health food store packages two types of snack foods, Chewy and Nutty, by mixing sunflower seeds, raisins, and peanuts. For the coming week the store has available in bulk quantities 90 kg of sunflower seeds, 100 kg of raisins, and 60 kg of peanuts. Invoices show that they paid \$135 for the sunflower seeds, \$180 for the raisins, and \$60 for the peanuts.

Chewy consists of two parts sunflower seeds, two parts peanuts, and six parts raisins. Nutty consists of three parts sunflower seeds and seven parts peanuts. Chewy sells for \$2.00/kg, and Nutty sells for \$1.50/kg. Determine a mixing scheme that will maximize the store's profits, assuming that its entire production of Chewy and Nutty will be sold.

Input

To solve a general linear programming problem, clearly the LP code will need to know the type of problem (maximize or minimize), the coefficients of the objective function $\mathbf{c}$, the coefficients of the constraints $\mathbf{A}$, the right-hand sides of the constraints $\mathbf{b}$, and the relation ($\leq$, $=$, $\geq$) for each constraint. There are several tradeoffs that can be made between ease of use for the problem solver and ease of programming and standardization for the programmer. Some codes assume that all problems are minimization problems, that all entries in $\mathbf{b}$ are nonnegative, and that all constraints are equalities. Thus, the user of these codes must put the model into a particular form by including slack variables and multiplying the objective function and each constraint by -1 if necessary. In this case the code provides the artificial variables where necessary.

In larger problems the majority of the entries of $\mathbf{A}$ will be zero. In this case the user would not want to have to input all these zeros, so most

codes provide for entering only the nonzero elements of **A** and assume that all other entries are zero. Consequently, the input must identify the constraint and variable to which the coefficient belongs. This specification is accomplished by asking the user to assign a name (perhaps limited to six or eight characters) to each constraint and each variable. Then each nonzero coefficient can be entered by giving the names of the constraint and variable to which it belongs. A drawback to this method is that the computer will interpret a misspelled name as a new variable or constraint. Some codes try to protect against such errors by keeping the name in two pieces and flagging all input where one piece of the name agrees but the other does not. Naming the variables and constraints also provides more descriptive output, especially when the names are chosen as mnemonics for the quantities that the variables and constraints represent. A disciplined naming convention is essential for automated report generation and solution analysis.

Either the objective function is entered separately by giving the name of the variable and its objective function coefficient or it is entered as part of the constraints. The right-hand sides are entered by giving the constraint name and the coefficient value. If the constraints have not been changed into a particular form, the type of relation for each constraint must also be specified.

EXAMPLE 1 (CONTINUED). Using mnemonic labels for the variables, we can write the mathematical model of the situation as

$$\begin{aligned}\text{Maximize}\quad z = {} & 2 \times \text{CHEWY} - (1.5)(0.2) \times \text{CHEWY} \\ & -(1.8)(0.6) \times \text{CHEWY} \\ & -(1)(0.2) \times \text{CHEWY} + 1.5 \times \text{NUTTY} \\ & -(1.5)(0.3) \times \text{NUTTY} - (1)(0.7) \times \text{NUTTY}\end{aligned}$$

or

$$\begin{aligned}&\text{Maximize} && z = 0.42 \times \text{CHEWY} + 0.35 \times \text{NUTTY} \\ &\text{subject to} \\ &\text{RAISIN:} && 6 \times \text{CHEWY} \le 100 \\ &\text{PEANUT:} && 2 \times \text{CHEWY} + 7 \times \text{NUTTY} \le 60 \\ &\text{SUN:} && 2 \times \text{CHEWY} + 3 \times \text{NUTTY} \le 90.\end{aligned}$$

Most modern LP codes consist of three modules that are executed in the sequence listed.

1. The **preprocessor** attempts to reduce the user's statement of the problem to one that can be solved more quickly. The preprocessor searches for ways to reduce the size of the problem by eliminating redundant constraints and variables. It seeks to set as many entries in the constraint matrix to zero as possible to allow this matrix to be stored more compactly and to improve the numerical stability of the solution process. The preprocessor uses reduction patterns that it tries to match with parts of the

given coefficient matrix. Typical, although very simple, reduction patterns include the following.

- Empty rows or columns
- Rows involving only one variable: $x_j = b_k$
- Generalized upper bounds with only two variables: $x_i + x_j = b_k$
- Rows with all positive coefficients and a nonpositive right-hand side
- Rows with all negative coefficients and a nonnegative right-hand side
- Constraints that are implied by other constraints

2. The **optimizer** actually solves the reduced linear programming problem.

3. The **postprocessor** transforms the output back to the setting of the original problem.

Besides specifying the problem, many codes allow for the user to specify various error-correcting features and output formats, which we will discuss later. They also provide the capability of identifying the problem to the code and user by allowing the user to give a title to the problem, which will be printed as a heading on each page of output.

A typical job setup for an LP code is shown in Figure 3.7. Note that the various types of data are separated by the key words ROWS, COLUMNS, and RHS.

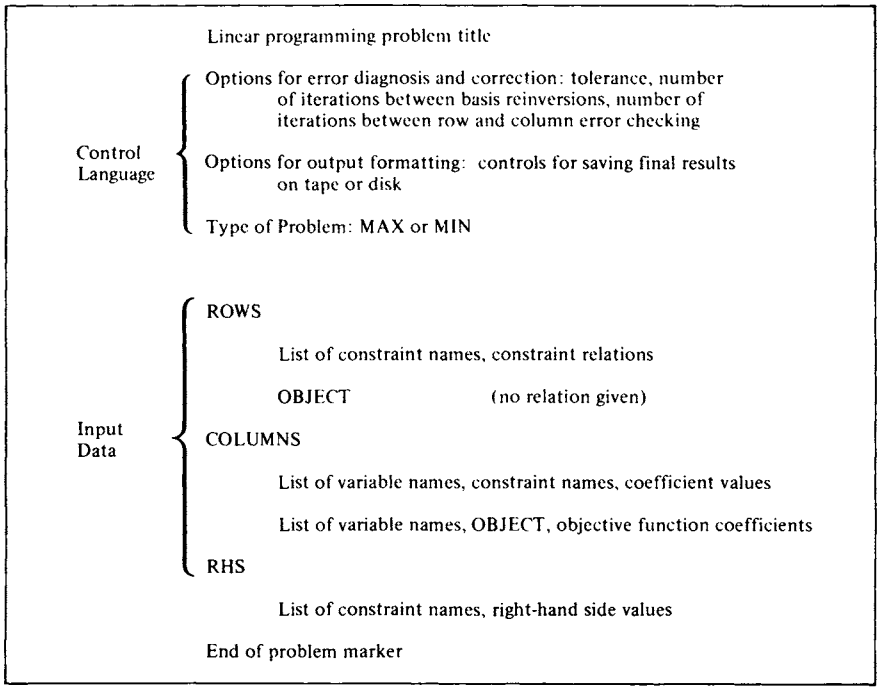


FIGURE 3.7

Figure 3.8 shows the input for this example expressed in standard MPS format as used by nearly all current optimizers and a procedural control program that is typical for batch-oriented mainframe systems. The display of the input matrix was produced by a subroutine of the optimizer, typically called TRANCOL.

Typical execution control program

```
0001        PROGRAM MIXTURE
0002        AMINMAX='MAX'
0003        CALL INPUT
0004        CALL SETUP
0005        CALL BCDOUT
0006        CALL PICTURE
0007        CALL TRANCOL
0008        CALL TRANROW
0009        CALL OPTIMIZE
0010        CALL SOLUTION
0011        STOP
0012        END
```

Standard format MPS input

```
NAME
ROWS.
 N PROFIT
 L RAISIN
 L PEANUT
 L SUN
COLUMNS
    CHEWY      PROFIT       .42000    RAISIN       6.00000
    CHEWY      PEANUT      2.00000    SUN          2.00000
    NUTTY      PROFIT       .35000    PEANUT       7.00000
    NUTTY      SUN         3.00000
RHS
    RHS1       RAISIN    100.00000    PEANUT      60.00000
    RHS1       SUN        90.00000
ENDATA
```

Display of input matrix

AT	CHEWY *LO*	NUTTY *LO*	RHS1
PROFIT	.42000	.35000	.00000
RAISIN	6.00000	.00000	100.00000
PEANUT	2.00000	7.00000	60.00000
SUN	2.00000	3.00000	90.00000

FIGURE 3.8

Algorithm

Virtually all LP codes designed for production, rather than teaching, use the revised simplex method. This method has several desirable features, including the ability to handle a large number of variables. The real limit on the size of a problem is the number of constraints (see Section 3.5). Other features will be described when we discuss error detection and correction.

Most of the large LP codes provide an option for computing $\mathbf{B}^{-1}$ that is based upon a procedure from numerical linear algebra called **LU factorization**. That is, **B** is written as **LU**, the product of a lower triangular matrix **L** and an upper triangular matrix **U**. The inverses of upper and lower triangular matrices are easily calculated. Then $\mathbf{B}^{-1} = \mathbf{U}^{-1}\mathbf{L}^{-1}$. Many large linear programming models have **sparse** matrices (ones with few nonzero entries). The matrix representations can then be highly compressed and $\mathbf{L}^{-1}$ and $\mathbf{U}^{-1}$ can be calculated in RAM, with special routines for sparse matrices, resulting in significant time savings. For this reason, more and more codes will provide an **LU**-factorization option.

The revised simplex algorithm with iterative $\mathbf{B}^{-1}$ calculation is usually programmed to check itself at specified intervals. Between checks it follows the description we gave in Section 3.4. The check involves computing the next $\mathbf{B}^{-1}$ in a manner different from the one we described. The matrix **B** can be constructed from the list of basic variables and the original problem as it was read in and stored. Then a very good method of numerically inverting **B**, such as the **LU**-factorization method described above, is used. This procedure of occasionally recomputing $\mathbf{B}^{-1}$ from the given problem serves to produce a more accurate basic feasible solution. However, in general the procedure is expensive in terms of computation time and must be used sparingly.

As was indicated in Section 2.2 most LP codes provide several options for handling degeneracy when it occurs.

Generalized Upper Bounding

Earlier commercially available mathematical programming systems typically included a procedure called generalized upper bounding (GUB, for short). This approach is historically interesting but has been superseded by improvements in algorithms and increased hardware speed.

It is usual for a large linear programming problem to have a substantial number of constraints that deal with bounds on certain variables or sums of variables or with the balance of materials between two stages of a process (the output of the first stage equals the input of the second stage). The special structure of these constraints allows them to be treated in a way different from that of a general constraint. The GUB procedure may be used on a problem whose constraints have the form shown in Figure 3.9. The GUB constraints are shaded.

FIGURE 3.9

The GUB procedure allows the reduction of the number of basic variables from one corresponding to each constraint to one corresponding to each general constraint (r variables in Figure 3.9). Thus, a problem with 1000 constraints, 800 of which are GUB constraints, can be solved with about the same effort as a problem with 200 general constraints plus the overhead of dealing with the GUB structure.

Models to which the GUB procedure may be applied arise in a number of areas, including production resource allocation (e.g., forest management or commercial fishing), multiproduct blending, and large-scale models of economic sectors.

Output

We have seen in Sections 3.3 and 3.5 that there is a large amount of information about the solution to a linear programming problem that is contained in the final tableau. A typical output from an LP code will summarize this information in a useful form. Some codes also allow the option of having the output saved on disk for future input to a report generator or to rerun the problem with slight variations.

The output will give the problem heading, the number of iterations used, and the optimal value of the objective function. Then usually a list of all the variables originally specified for the problem is produced. Those variables which are basic are so labeled. The value of each variable is given, along with the value of $z_j - c_j$ from the objective row.

The next part of the output lists the constraints and notes whether each is slack or binding. The value of the slack variable is also given. Because each constraint corresponds to a dual variable, the value of each dual variable, which may be interpreted as the marginal cost of each right-hand side (or resource) value, is also given.

Optionally, ranges of values for each of the objective function coefficients and each of the right-hand side values are given. These ranges come

from the sensitivity analysis of the optimal solution. One must interpret each range as giving the values that that particular coefficient may take, assuming that (1) no other coefficients are changed and that (2) the computed optimal value remains optimal. The output from MPS/90 for our example is shown in Figure 3.10.

Error Detection and Correction

Throughout this chapter we have used rational numbers such as $\frac{2}{3}$ and $-\frac{8}{11}$ while solving linear programming problems using the simplex or revised simplex algorithms. A computer, however, will convert these numbers to decimal representation and round off in the process. If the computer typically carries seven digits, $\frac{2}{3}$ becomes 0.6666667 and $-\frac{8}{11}$ becomes -0.7272727. After many calculations the errors made through round-off and truncation tend to accumulate. It is possible for the computer to produce a "solution" to a linear programming problem that is not feasible because of the round-off error.

Fortunately, the revised simplex method allows us to easily detect such errors as they accumulate. The key to this detection is that the revised simplex algorithm keeps the problem as it was entered and changes only the $\mathbf{B}^{-1}$ matrix. Suppose after some iterations the algorithm claims that $\hat{\mathbf{x}}_B = \hat{\mathbf{B}}^{-1}\mathbf{b}$ is a feasible solution. Since at each iteration $\mathbf{B}^{-1}$ is computed from the previous value, round-off errors could have occurred, making what we have recorded as $\mathbf{B}^{-1}$ different from the theoretical $\mathbf{B}^{-1}$.

However, we may take the original constraints in canonical form as $\mathbf{Ax} = \mathbf{b}$ and set all the nonbasic variables to zero. This yields the system

$$\mathbf{B}\mathbf{x}_B = \mathbf{B}\begin{bmatrix} x_{i_1} \\ x_{i_2} \\ \vdots \\ x_{i_m} \end{bmatrix} = \mathbf{b}.$$

We then compute $\mathbf{B}\hat{\mathbf{x}}_B$ and compare it with $\mathbf{b}$. If the difference between the two values exceeds a preset tolerance, we can invoke an error-correcting routine. This process is generally referred to as **checking row sums**. Normally, the tolerance is set at 10^{-6}, although most LP systems allow the user to change its value.

Column sums may also be checked by recalling that the entries in the objective row should be

$$\mathbf{c}_B^T\mathbf{t}_j - c_j = \mathbf{c}_B^T\mathbf{B}^{-1}\mathbf{A}_j - \mathbf{c}_j.$$

The error-correcting routine that is used when row or column sum errors are detected involves what is usually referred to as **reinverting the basis**. The matrix $\mathbf{B}$ can be formed exactly for the present list of basic variables

Log of simplex iterations

```
OPTIMIZE      OBJ = PROFIT       RHS = RHS1
TIME =     0.13            *MAXIMIZE*

 ITER    OBJ VAL     NON OPT    VEC IN     VEC OUT     PI/DJ
    1     7.00000         2          5           2     -.42000
    2     8.33333                    6           3     -.35000

OPTIMALITY REACHED
```

Solution report

```
SECTION 1 = ROWS

NUMBER      ROW       AT    ACTIVITY     SLACK ACTIVITY    LOWER LIMIT     UPPER LIMIT      PI VALUE

     1    PROFIT      BS      8.33333         -8.33333           NONE            NONE        1.00000
     2    RAISIN      UL    100.00000           .00000           NONE       100.00000        -.05333
     3    PEANUT      UL     60.00000           .00000           NONE        60.00000        -.05000
     4    SUN         BS     44.76190         45.23810           NONE        90.00000         .00000

SECTION 2 - COLUMNS

NUMBER    COLUMN    AT       ACTIVITY        INPUT COST     LOWER LIMIT      UPPER LIMIT    DJ VALUE

     5    CHEWY     BS        16.66667            .42000         .00000             NONE       .00000
     6    NUTTY     BS         3.80952            .35000         .00000             NONE       .00000
```

FIGURE 3.10

from the original problem. Then a very accurate method from numerical analysis is used to calculate $\mathbf{B}^{-1}$. From this a better value of $\mathbf{x}_B$ can be found, as $\mathbf{x}_B = \mathbf{B}^{-1}\mathbf{b}$.

In some codes this method of reinverting the basis is done at specified intervals irrespective of whether the row and column sums check. The length of this interval may be set by the user or may be determined automatically as a function of the size of the problem.

Scaling

One thing the user can do to minimize the errors generated by the algorithm is to make sure that the problem is appropriately scaled. This means that all the numbers in the constraint coefficient matrix **A** should be about the same size. If the coefficients of one constraint are more than 100 times the size of those of another constraint, severe error propagation may result. However, the large coefficients can be divided by an appropriate number to make them about the right size.

EXAMPLE 2. If two constraints in a linear problem are

$$2x_1 + 0.12x_2 + x_3 - 3x_4 = 7 \tag{1}$$

and

$$75x_1 + 250x_2 + 121x_3 + 314x_4 = 500, \tag{2a}$$

the second can be divided by 100 to yield

$$0.75x_1 + 2.5x_2 + 1.21x_3 + 3.14x_4 = 5. \tag{2b}$$

Constraints (1) and (2b) should be used as input to the revised simplex algorithm.

In the same way, if the coefficients of one variable are significantly different in magnitude from those of all other variables, the units of the variable should be changed to bring the size of the coefficients in line. This means that the corresponding objective row coefficient will also have to be changed.

EXAMPLE 3. Consider the linear programming problem

$$\begin{aligned} &\text{Maximize} \quad z = 23x_1 + 2x_2 + 3x_3 \\ &\text{subject to} \\ &\qquad 140x_1 + 5x_2 - x_3 \le 10 \\ &\qquad 210x_1 + x_2 + 3x_3 \le 14 \\ &\qquad x_1 \ge 0, \quad x_2 \ge 0, \quad x_3 \ge 0. \end{aligned}$$

Replacing x_1 with $x_1' = 100x_1$, we obtain the following problem:

$$\begin{aligned} &\text{Maximize} \quad z = 0.23x_i' + 2x_2 + 3x_3 \\ &\text{subject to} \\ &\qquad 1.4x_1' + 5x_2 - x_3 \leq 10 \\ &\qquad 2.1x_1' + x_2 + 3x_3 \leq 14 \\ &\qquad x_1' \geq 0, \quad x_2 \geq 0, \quad x_3 \geq 0. \end{aligned}$$

Thus, if the units of x_1 are grams/kilogram, the units of x_1' will be 100 grams/kilogram.

Restarting

Most large codes allow the user the option of saving the computations that have been done to a certain point. One scheme allows the user to specify the maximum number of iterations permitted. When that maximum number is reached, the list of basic variables and the original problem statement are written to disk for restarting at a later time. Another scheme allows the final solution and list of basic variables to be saved along with the original problem. This scheme is particularly useful because the restarting procedure allows additional constraints or variables to be put into the problem. In this case the dual simplex method can be used to restore feasibility if necessary.

The restart procedures are especially useful when analysis shows that the original model produced results that are inappropriate to the situation that was modeled and the model must be changed to better represent the actual situation. This procedure is also used when solving a family of related problems. The list of basic variables from the solution to one model is used as the starting point for solving the other methods.

Further Reading

Beale, E. M. L. *Mathematical Programming in Practice*. Pitman, London, 1968.

Dantzig, George B., and Van Slyke, R. M. "Generalized Upper Bounding Techniques for Linear Programming." *J. Comput. System Sci.* **1** (1967), 213–226.

Gale, David. *The Theory of Linear Economic Models*. McGraw–Hill, New York, 1960.

Gale, David, Kuhn, Harold W., and Tucker, Albert W. "On Symmetric Games," in *Contributions to the Theory of Games* (H. W. Kuhn and A. W. Tucker, Eds.), *Annals of Mathematical Studies*, No. 24, Princeton Univ. Press, Princeton, NJ, 1950.

Gass, Saul I. *Linear Programming*, fifth ed. McGraw–Hill, New York, 1984.

Geoffrion, A. M. "Elements of Large-Scale Mathematical Programming." *Management Sci.* **16** (1970), 652–691.

Lenstra, J. K., Rinnooykan, A. H. G., and Schrijver, A. (Eds.). *History of Mathematical Programming: A Collection of Personal Reminiscences*. Elsevier Science Publishers, Amsterdam, 1991.

Murty, Katta. *Linear and Combinatorial Programming*. Wiley, New York, 1976.

Nazareth, J. L. *Computer Solution of Linear Programs*. Oxford Univ. Press, New York, 1987.
Orchard-Hays, William. *Advanced Linear Programming Computing Techniques*. McGraw-Hill, New York, 1968.
Salkin, Harvey S., and Saha, Jahar (Eds.). *Studies in Linear Programming*. North-Holland, Amsterdam, 1975.
Thesen, Arne. *Computer Methods in Operations Research*. Academic Press, New York, 1978.
Wagner, Harvey. *Principles of Operations Research*. Prentice-Hall, Englewood Cliffs, NJ, 1969.
White, William W. "A Status Report on Computing Algorithms for Mathematical Programming." *ACM Comput. Surveys* **5** (1973), 135–66.

4

Integer Programming

IN THE LINEAR programming problems considered so far, the variables have been permitted to assume all nonnegative real values. However, there are many problems in which the variables must assume only integer values. For example, it would be meaningless to have an answer calling for the manufacture of half a table or for the chartering of 1.2 airplanes. In some problems, such as the transportation problem with integer values for supply and demand, the simplex method will yield integer answers; however, in many other problems it will not. In this chapter we formulate a number of problems that require integer variables and present three algorithms for solving these integer programming problems.

4.1 EXAMPLES

EXAMPLE 1 (THE TRANSPORTATION PROBLEM). Suppose a manufacturer making one product has m factories and n warehouses. The demand

at the jth warehouse is d_j, $j = 1, 2, \ldots, n$, and the supply available from the ith factory is s_i, $i = 1, 2, \ldots, m$. The cost of shipping one unit from the ith factory to the jth warehouse is c_{ij}. Our problem is to determine the amount, x_{ij}, of the product to be sent from the ith factory to the jth warehouse.

If we assume that the total supply at least equals the total demand,

$$\sum_{i=1}^{m} s_i \geq \sum_{j=1}^{n} d_j,$$

so that our problem is feasible, then the mathematical model is

$$\text{Minimize} \quad z = \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} x_{ij}$$

subject to

$$\sum_{j=1}^{n} x_{ij} \leq s_i, \quad i = 1, 2, \ldots, m$$

$$\sum_{i=1}^{m} x_{ij} \geq d_j, \quad j = 1, 2, \ldots, n$$

$$x_{ij} \geq 0 \quad \text{and} \quad \text{integral},$$

$$i = 1, 2, \ldots, m; \qquad j = 1, 2, \ldots, n.$$

If this problem is converted to standard form, then the only entries in the constraint matrix are 1s, −1s, and 0s. It can be shown using a result of Hoffman and Kruskal that in this case the simplex method will automatically yield integer solutions if the s_i and d_j are integers. However, the simplex method is a rather poor way of solving the transportation problem. In Chapter 5 we present a special algorithm for this problem that is rather efficient. This algorithm was developed because the transportation model arises repeatedly in practice. △

EXAMPLE 2 (THE KNAPSACK PROBLEM). Consider the problem faced by a hiker who cannot carry more than k pounds of equipment. She has n items that she is considering bringing. To each item she assigns a relative value, c_j, with the most important items having the highest values. Let a_j be the weight of the jth item. The hiker's problem is to decide which of the n items to carry; she will choose those that maximize the total relative value subject to the weight limitation.

To construct the mathematical model, let $x_j = 1$ if the jth item is chosen and let $x_j = 0$ if the jth item is not chosen. Then the model is

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

subject to

$$\sum_{j=1}^{n} a_j x_j \leq k$$

$$x_j = 0 \quad \text{or} \quad 1, \quad j = 1, 2, \ldots, n.$$

Note that by limiting the value of x_j to 0 or 1, the left-hand side of the constraint represents just the weight of the items that are chosen. This type of an integer programming problem is called a **zero–one programming problem.** △

EXAMPLE 3 (THE ASSIGNMENT PROBLEM). Suppose n people, P_1, $P_2, \ldots, P_n$, are being considered for n jobs, J_1, $J_2, \ldots, J_n$. Using various criteria, including past performance, aptitude, and job ability, we specify a value c_{ij} that would accrue if the ith person is given the jth job. We assume that each person is assigned to exactly one job and that each job is assigned to exactly one person. Our problem is to assign the people to the jobs so that the total value of the assignment is maximized.

To construct the mathematical model, define the variables x_{ij} so that

$$x_{ij} = \begin{cases} 1 & \text{if } P_i \text{ is assigned to } J_j \\ 0 & \text{otherwise.} \end{cases}$$

Then the mathematical model is

$$\text{Maximize} \quad z = \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij}$$

subject to

$$\sum_{i=1}^{n} x_{ij} = 1, \quad j = 1, 2, \ldots, n \tag{1}$$

$$\sum_{j=1}^{n} x_{ij} = 1, \quad i = 1, 2, \ldots, n \tag{2}$$

$$x_{ij} = 0 \quad \text{or} \quad 1, \quad i, j = 1, 2, \ldots, n.$$

Under the condition that x_{ij} has a value of either 0 or 1, exactly one of the summands in Equation (1) can be nonzero, and, likewise, exactly one of the summands in Equation (2) can be nonzero. Constraint (1) says that job

j is assigned to exactly one person; constraint (2) says that person i is assigned to exactly one job. Just as in the transportation problem, the result of Hoffman and Kruskal applies and the simplex algorithm yields a zero–one solution to the assignment problem. However, there is a special algorithm that efficiently handles this problem; it will be discussed in Chapter 5. △

EXAMPLE 4 (THE TRAVELING SALESMAN PROBLEM). A traveling salesman has to visit each of n cities, $C_1, C_2, \ldots, C_n$. He must start from his home office in city C_1 and return to C_1 after visiting each city exactly once. Such a route is called a **tour**. The order in which he visits cities $C_2, C_3, \ldots, C_n$ does not matter. He knows the distance between each pair of cities and wants to choose a tour that minimizes the total distance traveled.

To formulate the mathematical model, let c_{ij} be the distance between C_i and C_j. Let the variable x_{ij} be defined by

$$\begin{aligned} x_{ij} &= 1 \quad \text{if the route includes traveling from } C_i \text{ to } C_j \\ &= 0 \quad \text{otherwise.} \end{aligned}$$

The condition that the route must go to exactly one city after leaving C_i may be written

$$\sum_{j=1}^{n} x_{ij} = 1, \quad i = 1, 2, \ldots, n.$$

The condition that the route goes through every city exactly once can be phrased by saying that each city must be reached from exactly one city, or

$$\sum_{i=1}^{n} x_{ij} = 1, \quad j = 1, 2, \ldots, n.$$

Our mathematical model is then

$$\text{Minimize} \quad z = \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij}$$

subject to

$$\sum_{i=1}^{n} x_{ij} = 1, \quad j = 1, 2, \ldots, n \tag{3}$$

$$\sum_{j=1}^{n} x_{ij} = 1, \quad i = 1, 2, \ldots, n \tag{4}$$

$$x_{ij} = 0 \quad \text{or} \quad 1, \quad i, j = 1, 2, \ldots, n.$$

Consider the feasible solution for this problem when $n = 12$:

$$x_{12} = x_{23} = x_{34} = x_{45} = x_{56} = x_{61} = 1$$
$$x_{78} = x_{89} = x_{9,10} = x_{10,11} = x_{11,12} = x_{12,7} = 1$$

and

$$x_{ij} = 0 \quad \text{for all other values of } i \text{ and } j.$$

This solution is feasible, since each index from 1 to 12 occurs exactly once in the first position and exactly once in the second position. However, it is not an acceptable solution, since there are two disconnected subtours. We must design a way to eliminate disconnected routes from our set of feasible solutions.

To this end we introduce $n - 1$ new variables, $u_2, u_3, \ldots, u_n$, and $(n-1)^2 - (n-1)$ new constraints. The constraints are

$$u_i - u_j + nx_{ij} \leq n - 1, \quad i, j = 2, 3, \ldots, n, \quad \text{and} \quad i \neq j \tag{5}$$
$$u_i \geq 0 \quad \text{and} \quad \text{integral}, \quad i = 2, 3, \ldots, n.$$

Before we had $2n$ constraints and n^2 variables; these variables had values of either 0 or 1 and n of them x_{ii} were always 0. We now have

$$2n + (n-1)^2 - (n-1) = n^2 - n + 2$$

linear constraints and

$$n^2 + n - 1$$

integer-valued variables.

We now show that the constraints (3), (4), and (5) do not permit disconnected routes and still include all routes satisfying the original problem statement. First we assume that there is a subtour; that is, the route leads back to C_1 before visiting all the cities. Then there must be another subtour, since each city is visited exactly once. This subtour will start and end at some city in the list $C_2, C_3, \ldots, C_n$; it will not include C_1; and it will include $r \leq n - 1$ cities. The r variables x_{ij} that describe this subtour will be equal to 1. We add up the r constants (5) that correspond to these nonzero x_{ij}. This new constraint is satisfied by any solution that satisfies (5). As we take the sum to form this new constraint, we have $-u_j$ when the route enters city C_j and $+u_j$ when it leaves. Since the route enters and leaves each of the r cities exactly once, the u_j's cancel out in the sum. Thus, the new constraint is

$$nr \leq (n-1)r,$$

which is a contradiction of our assumption that there was a subtour of length $r \leq n - 1$.

For example, if we had the subtour starting at C_4,

$$C_4 \to C_5 \to C_3 \to C_2 \to C_4,$$

so that

$$x_{45} = x_{53} = x_{32} = x_{24} = 1,$$

then we would form our new constraint by adding the constraints

$$u_4 - u_5 + nx_{45} \leq n - 1$$
$$u_5 - u_3 + nx_{53} \leq n - 1$$
$$u_3 - u_2 + nx_{32} \leq n - 1$$
$$u_2 - u_4 + nx_{24} \leq n - 1$$

and obtain

$$4n \leq 4(n - 1).$$

We have now shown that constraints (3), (4), and (5) allow no subtours. Now we show that these constraints do not exclude any potential routes. To do this we show that each u_i can be assigned a nonnegative integer value for any route and that these values satisfy the constraints given in (5).

Let t_i be the position in the route at which C_i is visited. Thus, $t_1 = 1$ for C_1. If we consider the route that starts $C_1 \to C_4 \to C_6 \to C_2 \to \cdots$, then $t_1 = 1$, $t_4 = 2$, $t_6 = 3$, $t_2 = 4, \ldots$. Let $u_i = t_i$ for $i = 2, 3, \ldots, n$. We show that for each i and j, (5) holds. Either $x_{ij} = 1$ or $x_{ij} = 0$. If $x_{ij} = 1$, then C_j is visited immediately after C_i, so that

$$t_j = t_i + 1.$$

Substituting this equation into (5), we have

$$u_i - u_j + nx_{ij} = t_i - (t_i + 1) + n = n - 1$$

as we needed. If $x_{ij} = 0$, then since $u_i \leq n$ and $u_j \geq 2$, we have

$$u_i - u_j \leq n - 2 \leq n - 1,$$

so that (5) holds.

We have shown that a model for the traveling salesman problem is

$$\text{Minimize} \quad z = \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij}$$

subject to

$$\sum_{i=1}^{n} x_{ij} = 1, \quad j = 1, 2, \ldots, n$$

$$\sum_{j=1}^{n} x_{ij} = 1, \quad i = 1, 2, \ldots, n$$

$$u_i - u_j + nx_{ij} \leq n - 1, \quad i, j = 2, 3, \ldots, n, \quad \text{and} \quad i \neq j$$

$$x_{ij} = 0 \quad \text{or} \quad 1, \quad i, j = 1, 2, \ldots, n$$

$$u_i \geq 0 \quad \text{and} \quad \text{integral}, \quad i = 2, 3, \ldots, n. \qquad \triangle$$

EXAMPLE 5 (STOCK CUTTING PROBLEM). A plumber can buy plastic pipe in 6- and 12-ft lengths. The current job requires eight 4-ft lengths, five 5-ft lengths, and three 7-ft lengths. The plumber wants to figure out how many of each of the two stock lengths should be bought to minimize waste.

We determine all the possible ways the stock lengths can be cut to yield the necessary lengths. A 6-ft piece can be cut to give

one 4-ft length and 2 ft of scrap	(cutting pattern 1)
one 5-ft length and 1 ft of scrap	(cutting pattern 2).

A 12-ft piece can be cut to give

one 4-ft piece and 8 ft of scrap	(cutting pattern 3),
two 4-ft pieces and 4 ft of scrap	(cutting pattern 4),
three 4-ft pieces	(cutting pattern 5),
one 4-ft piece, one 5-ft piece, and 3 ft of scrap	(cutting pattern 6),
one 4-ft piece, one 7-ft piece, and 1 ft of scrap	(cutting pattern 7),
one 5-ft piece and 7 ft of scrap	(cutting pattern 8),
two 5-ft pieces and 2 ft of scrap	(cutting pattern 9),
one 7-ft piece and 5 ft of scrap	(cutting pattern 10),
one 7-ft piece and one 5-ft piece	(cutting pattern 11).

Let piece 1 be of length $l_1 = 4$ ft, let piece 2 be of length $l_2 = 5$ ft, and let piece 3 be of length $l_3 = 7$ ft. Let

a_{ij} = number of pieces of length l_i in cutting pattern j;
b_i = number of pieces of length l_i which are needed;
c_j = waste in cutting pattern j;
x_j = number of times cutting pattern j is used.

Our mathematical model is

$$\text{Minimize} \quad z = \mathbf{c}^T\mathbf{x}$$
$$\text{subject to}$$
$$\mathbf{A}\mathbf{x} = \mathbf{b}$$
$$\mathbf{x} \geq \mathbf{0} \quad \text{and} \quad \text{integral},$$

where

$$\mathbf{A} = \begin{bmatrix} 1 & 0 & 1 & 2 & 3 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 2 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 \end{bmatrix}$$

$$\mathbf{c}^{\mathrm{T}} = [2 \quad 1 \quad 8 \quad 4 \quad 0 \quad 3 \quad 1 \quad 7 \quad 2 \quad 5 \quad 0]$$

and

$$\mathbf{b} = \begin{bmatrix} 8 \\ 5 \\ 3 \end{bmatrix}. \qquad \triangle$$

EXAMPLE 6 (FIXED CHARGE PROBLEM). A manufacturing corporation makes n products, and naturally the board of directors wants to minimize manufacturing costs. Each unit of product j that is made costs c_j dollars to produce (raw materials, labor, direct machine costs, etc.). Moreover, if any units of product j are made, there is a fixed cost of k_j dollars, which represents the initial cost of setting up the production and distribution process.

Let x_j be the number of units of product j that are made. Suppose that the production process is constrained by a system of inequalities such as that in Exercises 1 or 2 in Section 1.1. Our objective function is

$$\text{Minimize} \quad z = \sum_{j=1}^{n} (c_j x_j + k_j y_j)$$

where the production constraints not involving the new variables y_i hold and in addition

$$y_j = \begin{cases} 1 & \text{if} \quad x_j > 0 \\ 0 & \text{if} \quad x_j = 0. \end{cases} \tag{6}$$

That is, y_j indicates whether any of the jth product is manufactured. The constraints in (6) are nonlinear functions of x_j and y_j. These constraints are not defined by hyperplanes as they must be for a linear programming problem.

However, this problem can be cast as a problem with *linear* constraints in which some of the variables are restricted to be integers and others may take any value. Such problems are called **mixed integer programming problems**.

Suppose we know an upper bound on the number of units of x_i that can be produced. That is, suppose we know numbers M_j, such that

$$x_j \leq M_j, \quad j = 1, 2, \ldots, n.$$

We now show that we may reformulate the definition of y_j in (6) as

$$\left.\begin{aligned} y_j &\geq \frac{x_j}{M_j} \\ y_j &= 0 \quad \text{or} \quad 1. \end{aligned}\right\} \tag{7}$$

If $x_j > 0$, then $y_j \geq x_j/M_j > 0$ implies that $y_j = 1$ since y_j can be only 0 or 1. If $x_j = 0$, then $y_j \geq x_j/M_j \geq 0$, so that $y_j = 0$ or 1. But since we are minimizing, the objective function will be smaller if $y_j = 0$. Therefore, at the minimum value of the objective function, if $x_j = 0$, then $y_j = 0$. The constraints given by (7) are now linear. We combine (7) with those constraints describing the production process to obtain our mixed integer programming model. △

EXAMPLE 7 (EITHER–OR PROBLEM). Suppose that we have a situation in which either the constraint

$$\sum_{j=1}^{n} a_{1j}x_j \leq b_1 \tag{8}$$

or the constraint

$$\sum_{j=1}^{n} a_{2j}x_j \leq b_2 \tag{9}$$

holds. We can convert this condition to one in which the constraints are linear if we have available numbers M_1 and M_2, such that

$$\sum_{j=1}^{n} a_{1j}x_j - b_1 \leq M_1$$

and

$$\sum_{j=1}^{n} a_{2j}x_j - b_2 \leq M_2$$

for all feasible solutions.

Let y be a zero–one variable. Consider the problem

$$\sum_{j=1}^{n} a_{1j}x_j - b_1 \leq M_1 y \tag{10}$$

$$\sum_{j=1}^{n} a_{2j}x_j - b_2 \leq M_2(1 - y) \tag{11}$$

$$y = 0 \quad \text{or} \quad 1.$$

If $y = 0$ in our new problem constraint, then (10) is the same as constraint (8), and constraint (11) is redundant, since it holds for all feasible solutions. If $y = 1$, then constraint (11) is the same as constraint (9), and constraint (10) is redundant. △

We now examine a general integer programming problem and describe some methods for solving it. We consider the following problem:

$$\begin{aligned} &\text{Maximize} \quad z = \mathbf{c}^{\mathrm{T}}\mathbf{x} \\ &\text{subject to} \\ &\qquad \mathbf{A}\mathbf{x} \leq \mathbf{b} \\ &\qquad \mathbf{x} \geq \mathbf{0} \\ &x_j = \text{integer} \quad \text{if} \quad j \in I, \end{aligned}$$

where I is a subset of $\{1, 2, \ldots, n\}$. If $I = \{1, 2, \ldots, n\}$, then the problem is called a **pure integer programming problem**. If I is a proper subset of $\{1, 2, \ldots, n\}$, then the problem is called a **mixed integer programming problem**. In a pure integer programming problem every variable is required to be an integer. In a mixed integer programming problem only some of the variables are required to have integer values. Examples 6 and 7 are mixed integer programming problems. Examples 1–5 are pure integer programming problems. In Examples 2 and 3 the variables are restricted to the values 0 or 1.

One might attempt to solve an integer programming problem by treating it as a linear programming problem (that is, by not restricting the variables to integer values) and then rounding the answer to the nearest integer. Under extremely fortunate circumstances one might not have to round at all. But there are other situations in which rounding will produce an incorrect answer.

EXAMPLE 8. Consider the integer programming problem

$$\begin{aligned} &\text{Maximize} \quad z = 7x + 8y \\ &\text{subject to} \\ &\qquad 10x + 3y \leq 52 \\ &\qquad 2x + 3y \leq 18 \\ &x \geq 0, \quad y \geq 0, \quad \text{and integers.} \end{aligned}$$

If we ignore the restriction that x and y are integers, the simplex method gives the solution (verify)

$$x = 4\tfrac{1}{4}, \quad y = 3\tfrac{1}{6}$$

with optimal value

$$z = 55\tfrac{1}{12}.$$

If we round the values of x and y to the nearest integer values that are feasible, we get

$$x = 4, \quad y = 3,$$

and

$$z = 52.$$

However, the solution

$$x = 3, \quad y = 4$$

is also feasible, and the value of the objective function for this solution is

$$z = 53. \qquad \Delta$$

4.1 EXERCISES

In Exercises 1–6 formulate the given problem as an integer programming problem.

1. **Equipment purchasing problem.** A ribbon manufacturer is considering the purchase of two different types of printing machines that will be used to emboss designs on the ribbon. Machine A can print 100 m per minute and requires 50 m^2 of floor space, whereas machine B can print 200 m per minute and requires 140 m^2 of floor space. Suppose that the manufacturer must print at least 600 m per minute and has no more than 350 m^2 of floor space. If a model A machine costs \$22,000 and a model B machine costs \$48,000, how many machines of each type should be bought to minimize the cost?
2. **A production problem.** A chair manufacturer makes three different types of chairs, each of which must go through sanding, staining, and varnishing. In addition, the model with the vinyl-covered back and seat must go through an upholstering process. The following table gives the time required for each operation on each type of chair, the available time for each operation in hours per month, and the profit per chair for each model. How many chairs of each type should be made to maximize the total profit?

Model	*Sanding (hr)*	*Staining (hr)*	*Varnishing (hr)*	*Upholstering (hr)*	*Profit ($)*
A—solid back and seat	1.0	0.5	0.7	0	10
B—ladder back, solid seat	1.2	0.5	0.7	0	13
C—vinyl-covered back and seat	0.7	0.3	0.3	0.7	8
Total time available per month	600	300	300	140	

3. Pam Hardy currently has six favorite country and western songs. There are 10 compact disks that contain different groups of these songs available. Suppose that the jth CD costs c_j dollars. Set up a model that Pam could use to

determine the cheapest selection of CDs to buy to get at least one version of each of her favorite songs.

4. Tommy Jones's mother is planning his 10th birthday party and will serve a variety of soft drinks, which will be chosen from the list below.

Drink	*Cola*	*Root beer*	*Cherry*	*Lemon*	*Orange*	*Grape*	*Ginger ale*
Price per bottle (cents)	69	59	62	62	65	55	65

From past experience it has been determined that at least 12 bottles of soft drinks are needed. Also, at least 2 bottles of ginger ale, at least 2 bottles of cola, and no more than 3 bottles of fruit-flavored soft drinks are needed. How many bottles of each type should be bought to minimize the total cost?

5. A manager for a large corporation must prepare a list of projects that her group will complete over the next year. She has under consideration 10 such projects but will not be able to do all of them because of limits on personnel and budget. She has assigned a weight to each project that represents to her the value of completing the project. The personnel, capital requirements, and weights for each project are given in the following table.

	Project									
	1	2	3	4	5	6	7	8	9	10
Person-weeks	250	195	200	70	30	40	100	170	40	120
Cost (thousands of dollars)	400	300	350	100	70	70	250	250	100	200
Value of completion	70	50	60	20	10	20	30	45	10	40

The manager has available 1000 person-weeks and $1,500,000 to allocate among the projects. Which projects should she choose to complete to maximize the value?

6. Each day at the Graphic Arts Co. the press operator is given a list of jobs to be done during the day. He must determine the order in which he does the jobs based on the amount of time it takes to change from one job setup to the next. Clearly he will arrange the jobs in an order that minimizes the total setup time. Assume that each day he starts the press from a rest state and returns it to that state at the end of the day. Suppose on a particular day that he must do six jobs for which he estimates the changeover times given in the following table. What schedule of jobs should the operator use?

	j						
i	1	2	3	4	5	6	*Rest*
	⟨From job i to job j (min)⟩						
1	0	10	5	15	10	20	5
2	10	0	10	10	20	15	10
3	5	5	0	5	10	10	15
4	8	10	3	0	9	14	10
5	4	7	8	6	0	10	10
6	10	5	10	15	10	0	8
Rest	7	7	9	12	10	8	0

4.1 PROJECTS

1. Meg Watkins is trying to decide which college to attend. From among the applications that she submitted, she has been admitted to four schools. One is a large state university that is about 250 miles from her home. At this school she may live in a dormitory for two years, but then must find accommodations in the community. Another is a small private school about 1000 miles from her home that has an excellent reputation. At this school there are dormitory accommodations for all students. Meg, under pressure from her father, also applied to and was accepted by the private church-related school in her hometown. Since she is a local student, the school would expect her to live at home.

   Another possibility open to Meg is to go to the local community college for two years and then transfer to another school. The reputation of the community college has been improving over the last few years. The state university would carry forward her acceptance for two years and transfer all credits. The distant private school will also carry forward her acceptance but most likely will transfer nine credits fewer than two full years of credit. The local private school has no formal statement on transfer from two-year schools. The accompanying table gives the cost for attending each school and Meg's assessment of the worth (or utility) of having certain blocks of credit from each school.

	State	*Distant private*	*Local private*	*Community college*
Tuition	$2500/year	$14,000/year	$9000/year	—
Living				
On campus	$3500/year	$3500/year	$125/month	$125/month
Off campus	$4000/year			
Humanities	7	9	6	6
Social science	6	8	5	6
Science	8	5	4	3
Major	8	10	6	—

Set up a model that Meg could use to maximize the future worth of her education assuming that she can earn $3000/summer and that her father will

provide \$6000/year for her education and living expenses. You may wish to consider allowing Meg the option of working while going to school or of going to summer school for certain courses.

2. Consider the problem of making change in a supermarket. Suppose that the cash register shows that your change is to be C cents. Initially, we assume $C < 100$. The coins that are available to make change have values

$$w_1 = 1, \quad w_2 = 5, \quad w_3 = 10, \quad w_4 = 25, \quad \text{and} \quad w_5 = 50.$$

(a) Set up an integer programming problem for finding which combination of coins yields the correct change using the smallest number of coins.
(b) Construct a table giving the solution to part (a) for $C = 1, 2, \ldots, 99$.
(c) One algorithm for change-making calls for giving as many of the largest-denomination coins as possible, then using the next largest denomination for the remaining amount, and so on. Does this algorithm give the correct answer for each $C = 1, 2, \ldots, 99$?
(d) Suppose our monetary system had coins with values

$$w_1 = 1, \quad w_2 = 5, \quad w_3 = 20, \quad \text{and} \quad w_4 = 25.$$

Use the algorithm in part (c) to make up $C = 40$. Is this a minimal solution?
(e) Consider the problem of giving C dollars in change, where $C = 1, 2, \ldots, 9$. Would it be advantageous to use a \$2 bill in addition to the \$1 and \$5 bills?
(f) What other situations have models similar to the change-making problem?

Further Reading

Chang, S. K., and Gill, A. "Algorithmic Solution of the Change-Making Problem." *J. ACM* **17** (1970), 113–122.

Hoffman, A. J., and Kruskal, J. B. "Integral Boundary Points of Convex Polyhedra," in *Linear Inequalities and Related Systems* (H. W. Kuhn and A. W. Tucker, Eds.), pp. 223–246. Princeton Univ. Press, Princeton, NJ, 1956.

4.2 CUTTING PLANE METHODS

In this section we discuss one approach that has been used to solve integer programming problems. The algorithms for such problems are not as nice as those for linear programming problems in the sense that there is not one algorithm that works well for all integer programming problems. Among the difficulties with these algorithms is their inefficiency for even medium-sized problems. For example, computations for the traveling salesman problem (Example 4 in Section 4.1) become prohibitively long for over 200 cities.

Consider the pure integer programming problem

$$\text{Maximize} \quad z = \mathbf{c}^T\mathbf{x} \tag{1}$$

subject to

$$\mathbf{Ax} = \mathbf{b} \tag{2}$$

$$\mathbf{x} \geq \mathbf{0} \quad \text{and} \quad \text{integral}, \tag{3}$$

where **A** is an $m \times n$ matrix, **c** is an $n \times 1$ column vector, and **b** is an $m \times 1$ column vector.

The **cutting plane algorithms** were developed by Ralph Gomory in 1958. The idea behind the algorithms is to start with the problem given by (1), (2), and (3); ignore the restriction that **x** must be a vector with integer components; and solve the resulting problem by the simplex method. If the resulting solution is integral, we are finished. Otherwise, we add a new constraint that "cuts off" (eliminates) some nonintegral solutions, including the one just obtained by the simplex method. However, the new constraint is carefully constructed so as not to eliminate any feasible integer solutions. We solve the new problem and repeat the simplex algorithm. By adding enough constraints, we eventually reach an optimal integer solution.

Suppose the problem has a feasible solution and that a finite optimal value exists. We may assume that **A**, **B**, and **c** have integer entries. The ith constraint as it appears in the final simplex tableau for the related linear programming problem is

$$\sum_{j=1}^{n} t_{ij}x_j = x_{Bi}, \tag{4}$$

where x_{Bi} is the value of the ith basic variable for the optimal solution of the related linear programming problem. We denote by $[a]$ the **integer part** of a. That is, $[a]$ is the largest integer K, such that $K \leq a$. For example,

$$[\tfrac{3}{2}] = 1, \qquad [\tfrac{3}{5}] = 0$$
$$[-\tfrac{2}{3}] = -1, \quad [-3\tfrac{1}{3}] = -4$$
$$[3] = 3, \qquad [-2] = -2.$$

Since $[t_{ij}] \leq t_{ij}$ and $x_j \geq 0$, we can write from (4)

$$\sum_{j=1}^{n} [t_{ij}]x_j \leq x_{Bi}. \tag{5}$$

Any integer vector **x** that satisfies (4) must also satisfy (5). For such **x** the left-hand side of (5) is an integer. Thus, we may write the constraint (6) that **x** must also satisfy:

$$\sum_{j=1}^{n} [t_{ij}]x_j \leq [x_{Bi}]. \tag{6}$$

We can transform (6) into an equation by introducing the slack variable u_i:

$$\sum_{j=1}^{n} [t_{ij}]x_j + u_i = [x_{Bi}]. \tag{7}$$

Since u_i is the difference between two integers, we may require that u_i be an integer. We have shown that any feasible solution to the given integer programming problem will also satisfy (7) for some value of u_i.

Now assume that x_{Bi} is not an integer. We may write

$$[t_{ij}] + g_{ij} = t_{ij}$$

and

$$[x_{Bi}] + f_i = x_{Bi},$$

where $0 \leq g_{ij} < 1$ and $0 < f_i < 1$. The quantity f_i is called the **fractional part** of x_{Bi}. Thus, if

$$t_{ij} = \tfrac{9}{7}, \quad \text{then} \quad g_{ij} = \tfrac{2}{7},$$

and if

$$t_{ij} = -\tfrac{2}{3}, \quad \text{then} \quad g_{ij} = \tfrac{1}{3}.$$

If we subtract (4) from (7), we have

$$\sum_{j=1}^{n} (-g_{ij})x_j + u_i = -f_i. \tag{8}$$

Equation (8) is the **cutting plane constraint** to be added to the constraints in (2). We now proceed as follows.

Step 1. Solve the related linear programming problem obtained from the given integer programming problem by dropping the integrality requirements. If the solution is a vector with integer components, stop. Otherwise, go to Step 2.

Step 2. Generate a cutting plane constraint as in (8). A heuristic rule to use for choosing the constraint in the cutting plane construction is choose the constraint in the final simplex tableau that gives the largest f_i.

Step 3. Consider the new integer programming problem, which consists of the same objective function, the constraints in (2), and the cutting plane (8), which have been added in Step 2. Return to Step 1.

Since the coefficients of all the basic variables except the ith one in the list will be zero in (4), we may rewrite (4) as

$$x_{r_i} + \sum_{j \in N} t_{ij}x_j = x_{Bi}, \tag{9}$$

where N is the set of indices of the nonbasic variables and where the variable labeling this ith constraint is x_{r_i}. Suppose that x_{Bi} is not an integer. The Gomory cutting plane obtained from (9) is

$$\sum_{j \in N} (-g_{ij})x_j + u_i = -f_i, \tag{10}$$

where

$$[t_{ij}] + g_{ij} = t_{ij} \quad \text{and} \quad [x_{Bi}] + f_i = x_{Bi}.$$

If we set the nonbasic variables x_j equal to zero in (10), we obtain $u_i = -f_i < 0$. Since the slack variable u_i is negative, we see that the optimal solution to the related linear programming problem whose ith constraint is given by (9) is no longer a feasible solution to the new linear programming problem with the added constraint (10). We have cut off the current optimal solution. The dual simplex method can now be used to solve the new problem; it will remove the infeasibility caused by adding the constraint (10).

We have shown that the cutting plane method fulfills the two criteria. It does not remove any integer vectors from the set of feasible solutions, but it does remove noninteger optimal solutions.

EXAMPLE 1. Consider the pure integer programming problem

$$\begin{aligned} &\text{Maximize} \quad z = 5x_1 + 6x_2 \\ &\text{subject to} \\ &\qquad 10x_1 + 3x_2 \leq 52 \\ &\qquad 2x_1 + 3x_2 \leq 18 \\ &x_1 \geq 0, \quad x_2 \geq 0, \quad \text{integers.} \end{aligned}$$

The final simplex tableau for the related linear programming problem is given in Tableau 4.1 (verify), where x_3 and x_4 are the slack variables for the first and second constraints, respectively. Since the solution is nonintegral, we must add a cutting plane constraint. We find $f_1 = \frac{1}{4}$ and $f_2 = \frac{1}{6}$, so that we choose the first row for constructing the cutting plane. We have

$$[\tfrac{1}{8}] + \tfrac{1}{8} = \tfrac{1}{8}$$

and

$$[-\tfrac{1}{8}] + \tfrac{7}{8} = -\tfrac{1}{8}.$$

Tableau 4.1

		5	6	0	0	
$\mathbf{c}_B$		x_1	x_2	x_3	x_4	$\mathbf{x}_B$
5	x_1	1	0	$\frac{1}{8}$	$-\frac{1}{8}$	$\frac{17}{4}$
6	x_2	0	1	$-\frac{1}{12}$	$\frac{5}{12}$	$\frac{19}{6}$
		0	0	$\frac{1}{8}$	$\frac{15}{8}$	$\frac{161}{4}$

The Gomory cutting plane is then

$$-\tfrac{1}{8}x_3 - \tfrac{7}{8}x_4 + u_1 = -\tfrac{1}{4}.$$

Our new tableau is Tableau 4.2. We must now use the dual simplex method on Tableau 4.2 to restore feasibility. We obtain Tableau 4.3.

Tableau 4.2

	c_B		5 x_1	6 x_2	0 x_3 ↓	0 x_4	0 u_1	x_B
	5	x_1	1	0	$\frac{1}{8}$	$-\frac{1}{8}$	0	$\frac{17}{4}$
	6	x_2	0	1	$-\frac{1}{12}$	$\frac{5}{12}$	0	$\frac{19}{6}$
←	0	u_1	0	0	$(-\frac{1}{8})$	$-\frac{7}{8}$	1	$-\frac{1}{4}$
			0	0	$\frac{1}{8}$	$\frac{15}{8}$	0	$\frac{161}{4}$

Tableau 4.3

c_B		5 x_1	6 x_2	0 x_3	0 x_4	0 u_1	x_B
5	x_1	1	0	0	−1	1	4
6	x_2	0	1	0	1	$-\frac{2}{3}$	$\frac{10}{3}$
0	x_3	0	0	1	7	−8	2
		0	0	0	1	1	40

Since the solution is still nonintegral, we formulate another Gomory cutting plane using the x_2 row. We have

$$[1] + 0 = 1$$
$$[-\tfrac{2}{3}] + \tfrac{1}{3} = -\tfrac{2}{3}$$
$$[\tfrac{10}{3}] + \tfrac{1}{3} = \tfrac{10}{3}.$$

The cutting plane equation is

$$0x_4 - \tfrac{1}{3}u_1 + u_2 = -\tfrac{1}{3}.$$

We show the new tableau in Tableau 4.4.

Tableau 4.4

	c_B		5 x_1	6 x_2	0 x_3	0 $-x_4$	0 u_1 ↓	0 u_2	x_B
	5	x_1	1	0	0	−1	1	0	4
	6	x_2	0	1	0	1	$-\frac{2}{3}$	0	$\frac{10}{3}$
	0	x_3	0	0	1	7	−8	0	2
←	0	u_2	0	0	0	0	$(-\frac{1}{3})$	1	$-\frac{1}{3}$
			0	0	0	1	1	0	40

Tableau 4.5

c_B		5 x_1	6 x_2	0 x_3	0 x_4	0 u_1	0 u_2	x_B
5	x_1	1	0	0	−1	0	3	3
6	x_2	0	1	0	1	0	−2	4
0	x_3	0	0	1	7	0	−24	10
0	u_1	0	0	0	0	1	−3	1
		0	0	0	1	0	3	39

Using the dual simplex method on Tableau 4.4, we obtain Tableau 4.5. We have obtained the optimal integer solution

$$x_1 = 3, \quad x_2 = 4, \quad z = 39.$$

The first cutting plane,

$$-\tfrac{1}{8}x_3 - \tfrac{7}{8}x_4 + u_1 = -\tfrac{1}{4},$$

can be expressed in terms of x_1 and x_2 by substituting

$$x_3 = 52 - 10x_1 - 3x_2$$
$$x_4 = 18 - 2x_1 - 3x_2,$$

which come from the constraints of the original problem. We obtain

$$3x_1 + 3x_2 + u_1 = 22 \tag{11}$$

or

$$x_1 + x_2 \leq \tfrac{22}{3}. \tag{12}$$

The second cutting plane,

$$-\tfrac{1}{3}u_1 + u_2 = -\tfrac{1}{3},$$

gives

$$x_1 + x_2 \leq 7. \tag{13}$$

In this calculation we used (11) to write u_1 in terms of x_1 and x_2. We sketch the original feasible region and the cutting planes in Figure 4.1. △

Several remarks are in order. For a linear programming problem the number of positive components in the solution vector is $\leq m$, where $\mathbf{A}$ is $m \times s$. For an integer programming problem this is no longer true. In fact, the optimal value may not occur at an extreme point of the convex region defined by the constraints.

For a pure integer programming problem, the set of feasible solutions is the set of points with integer coordinates, called **lattice points**, lying within the convex region defined by the constraints (see Figure 4.1). If the convex region is bounded, there are just a finite number of feasible solutions.

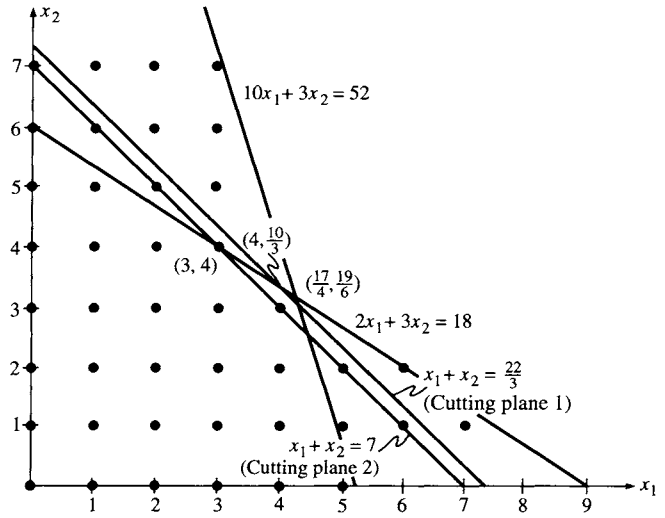


FIGURE 4.1

However, for problems with several variables it is computationally impractical to evaluate the objective function at each of these points and then choose the point with the largest value. Instead the cutting plane method works from an extreme point and closes in on one of the lattice points at which the objective function attains its optimal value.

The cutting plane algorithm that we just gave has several major drawbacks. The integral answer does not appear until the very last step. This is unlike the simplex method, in which we can stop at any stage and obtain a feasible solution that is better than the previous feasible solutions. In the Gomory algorithm, if we stop before the end, we have no integral solution. Also, the cutting planes are constructed by using fractional parts of the coefficients and the right-hand side of a constraint, and, consequently, round-off errors may cause the method to converge slowly.

Another method used for solving integer programming problems occurring in common applications avoids this difficulty by introducing a pair of constraints based on the integer part of one of the basic variables. This method, called the branch and bound method, is described in the Section 4.3.

Gomory's Method for Mixed Integer Programming

We can attempt to solve a mixed integer programming problem in the same way as a pure integer programming problem. We use the simplex method to obtain the optimal solution to the related linear programming

problem. This will be a solution to the mixed integer programming problem if those variables appearing in the basis for the optimal solution that are required to have integer values actually do have such values. We suppose now that x_{r_i}, the ith basic variable in the optimal solution of the related linear programming problem is required to be integral and that x_{Bi} is not an integer. We saw that we may write the ith constraint (9) as

$$x_{r_i} + \sum_{j \text{ nonbasic}} t_{ij}x_j = x_{Bi}. \tag{14}$$

Let $x_{Bi} = [x_{Bi}] + f_i$. We may write (14) as

$$x_{r_i} + \sum_{j \text{ nonbasic}} t_{ij}x_j = [x_{Bi}] + f_i$$

or

$$\sum_{j \in N} t_{ij}x_j = f_i + \left([x_{Bi}] - x_{r_i}\right). \tag{15}$$

We now divide the set of indices for the nonbasic variables into two subsets N^+ and N^-, where

$$N^+ = \{j \mid j \text{ is the index of a nonbasic variable and } t_{ij} \geq 0\}$$

$$N^- = \{j \mid j \text{ is the index of a nonbasic variable and } t_{ij} < 0\}.$$

Then (15) may be written as

$$\sum_{j \in N^+} t_{ij}x_j + \sum_{j \in N^-} t_{ij}x_j = f_i + \left([x_{Bi}] - x_{r_i}\right). \tag{16}$$

Using (16), we want to derive a cutting plane constraint that will cut off the current optimal solution to the related linear programming problem because in that solution x_{r_i} does not have an integer value. The cutting plane constraint must also have the property, as before, that no feasible solutions for the mixed integer problem should be cut off. We can interpret this condition as saying that if x_{r_i} satisfies (16) and if x_{r_i} is an integer, then x_{r_i} must satisfy the cutting plane constraint.

We consider two cases: the right-hand side of (16) is positive, or it is negative. Assume, first, that $f_i + ([x_{Bi}] - x_{r_i}) < 0$. The quantity in parentheses is an integer by assumption and $0 < f_i < 1$. We see that $[x_{Bi}] - x_{r_i}$ must be a negative integer for all these assumptions to hold.

Thus, the largest value that the right-hand side of (16) can have while still remaining negative is $f_i - 1$. Our original constraint (14) implies, in this case,

$$\sum_{j \in N^+} t_{ij}x_j + \sum_{j \in N^-} t_{ij}x_j \leq f_i - 1. \tag{17}$$

We can make the left-hand side of (17) smaller by removing the first sum, since it represents a nonnegative quantity. We obtain

$$\sum_{j \in N^-} t_{ij} x_j \leq f_i - 1.$$

Dividing by $f_i - 1$ and reversing the inequality, since $f_i - 1$ is negative, and then multiplying by f_i yields

$$\sum_{j \in N^-} \frac{f_i}{f_i - 1} t_{ij} x_j \geq f_i. \tag{18}$$

Since the first sum in (17) is nonnegative, we may add it to (18) to obtain

$$\sum_{j \in N^+} t_{ij} x_j + \frac{f_i}{f_i - 1} \sum_{j \in N^-} t_{ij} x_j \geq f_i. \tag{19}$$

For the other case, we assume that $f_i + ([x_{Bi}] - x_{r_i}) \geq 0$. Using reasoning similar to that in the first case, we see that $[x_{Bi}] - x_{r_i}$ must be nonnegative. Thus, our original constraint (14) implies that

$$\sum_{j \in N^-} t_{ij} x_j + \sum_{j \in N^-} t_{ij} x_j \geq f_i. \tag{20}$$

We may replace the second sum in (20) by any larger quantity and still maintain the inequality. We have

$$\sum_{j \in N^-} t_{ij} x_j \leq 0 \leq \frac{f_i}{f_i - 1} \sum_{j \in N^-} t_{ij} x_j.$$

Consequently, (20) implies that

$$\sum_{j \in N^+} t_{ij} x_j + \frac{f_i}{f_i - 1} \sum_{j \in N^-} t_{ij} x_j \geq f_i,$$

which is the same as (19).

We have now shown that if (14) is a constraint in a mixed integer problem whose corresponding basic variable is supposed to have an integer value but does not, then (19) is satisfied by every vector $\mathbf{x}$ that satisfies (14), assuming that x_{r_i} is an integer. From (19) we can construct the equation of the cutting plane. We reverse the inequality in (19) and add a slack variable u_i, obtaining

$$-\sum_{j \in N^+} t_{ij} x_j - \frac{f_i}{f_1 - 1} \sum_{j \in N^-} t_{ij} x_j + u_i = -f_i. \tag{21}$$

Equation (21) is the Gomory cutting plane. For the current optimal solution, $x_j = 0$ if $j \in N^+$ or $j \in N^-$. Therefore, introducing the constraint (21) gives

$$u_i = -f_i < 0$$

for the current optimal solution, and hence it has been cut off, since u_i must be nonnegative.

Note that (21) may be written as

$$u_i = -f_i + \sum_{j \in N} d_j x_j,$$

where

$$d_j = \begin{cases} t_{ij} & \text{if} \quad t_{ij} \geq 0 \\ \dfrac{f_i}{f_i - 1} t_{ij} & \text{if} \quad t_{ij} < 0. \end{cases} \tag{22}$$

It is possible that some of the nonbasic variables are also required to be integer valued. We can refine the cutting plane defined by (21) if we use this information. The refinement will simply cut off more solutions that do not satisfy the integrality conditions. We write

$$t_{ij} = [t_{ij}] + g_{ij}.$$

Then, instead of the definitions of d_j given in (22), we have for the refined cutting plane

$$d_j = \begin{cases} t_{ij} & \text{if} \quad t_{ij} \geq 0 \quad \text{and } x_j \text{ may be nonintegral} \\ \dfrac{f_i}{f_i - 1} t_{ij} & \text{if} \quad t_{ij} < 0 \quad \text{and } x_j \text{ may be nonintegral} \\ g_{ij} & \text{if} \quad g_{ij} \leq f_i \quad \text{and } x_j \text{ must be integral} \\ \dfrac{f_i}{f_i - 1}(g_{ij} - 1) & \text{if} \quad g_{ij} > f_i \quad \text{and } x_j \text{ must be integral.} \end{cases} \tag{23}$$

△

EXAMPLE 2. Consider the mixed integer programming problem

$$\begin{aligned} &\text{Maximize} \quad z = 5x_1 + 6x_2 \\ &\text{subject to} \\ &\qquad 10x_1 + 3x_2 \leq 52 \\ &\qquad 2x_1 + 3x_2 \leq 18 \\ &\qquad x_1 \geq 0 \quad \text{and} \quad \text{integral} \\ &\qquad x_2 \geq 0. \end{aligned}$$

The final simplex tableau for the related linear programming problem is the same as that in Example 1 and is given in Tableau 4.1. Since the value for x_1 is not an integer, we must add a cutting plane constraint. The nonbasic variables are x_3 and x_4, the slack variables. Neither is constrained to be an integer. We therefore use (21) to define our cutting plane. In this case, $i = 1$ and

$$x_{Bi} = 4 + \tfrac{1}{4},$$

so that $f_1 = \frac{1}{4}$. From Tableau 4.1 we see that $N^+ = \{3\}$ and $N^- = \{4\}$. The cutting plane is

$$-t_{13}x_3 - \frac{f_1}{f_1 - 1}t_{14}x_4 + u_1 = -f_1$$

or

$$-\tfrac{1}{8}x_3 - \frac{\frac{1}{4}}{-\frac{3}{4}}(-\tfrac{1}{8})x_4 + u_1 = -\tfrac{1}{4}$$

or

$$-\tfrac{1}{8}x_3 - \tfrac{1}{24}x_4 + u_1 = -\tfrac{1}{4}.$$

Putting this constraint into Tableau 4.1, we get Tableau 4.6.

Tableau 4.6

				↓			
c_B		5 x_1	6 x_2	0 x_3	0 x_4	0 u_1	x_B
5	x_1	1	0	$\frac{1}{8}$	$-\frac{1}{8}$	0	$\frac{17}{4}$
6	x_2	0	1	$-\frac{1}{12}$	$\frac{5}{12}$	0	$\frac{19}{6}$
← 0	u_1	0	0	$(-\frac{1}{8})$	$-\frac{1}{24}$	1	$-\frac{1}{4}$
		0	0	$\frac{1}{8}$	$\frac{15}{8}$	0	$\frac{161}{4}$

We use the dual simplex method to restore feasibility in Tableau 4.7. The solution that Tableau 4.7 represents,

$$x_1 = 4, \quad x_2 = \tfrac{10}{3}, \quad z = 40,$$

satisfies the integrality conditions so that it is an optimal solution to the given mixed integer programming problem.

Tableau 4.7

c_B		5 x_1	6 x_2	0 x_3	0 x_4	0 u_1	x_B
5	x_1	1	0	0	$-\frac{1}{6}$	1	4
6	x_2	0	1	0	$\frac{4}{9}$	$-\frac{2}{3}$	$\frac{10}{3}$
0	x_3	0	0	1	$\frac{1}{3}$	-8	2
		0	0	0	$\frac{11}{6}$	1	40

△

EXAMPLE 3. Consider the mixed integer programming problem

$$\text{Maximize} \quad z = 4x_1 + 5x_2 + 3x_3$$

subject to

$$3x_1 \qquad + 4x_3 \le 10$$
$$2x_1 + x_2 + x_3 \le 7$$
$$3x_1 + 4x_2 + x_3 \le 12$$
$$x_1 \ge 0, \quad x_3 \ge 0, \quad \text{and} \quad \text{integral}$$
$$x_2 \ge 0.$$

The final tableau of the related linear programming problem is Tableau 4.8 (verify). Since x_1 and x_3 are constrained to be integer valued, the solution represented by Tableau 4.8 is not feasible for the given mixed integer problem. We introduce a cutting plane from the first constraint in Tableau 4.8 ($i = 1$). Note that x_4 must also be integer valued, since it is the difference of two integers. The set of indices of nonbasic variables is $N = \{1, 4, 6\}$. We calculate the value of d_j using (23) for each of these indices. The fractional part of $x_3 = \frac{5}{2}$ is $f_1 = \frac{1}{2}$.

Tableau 4.8

c_B		4 x_1	5 x_2	3 x_3	0 x_4	0 x_5	0 x_6	x_B
3	x_3	$\frac{3}{4}$	0	1	$\frac{1}{4}$	0	0	$\frac{5}{2}$
0	x_5	$\frac{11}{16}$	0	0	$-\frac{3}{16}$	1	$-\frac{1}{4}$	$\frac{17}{8}$
5	x_2	$\frac{9}{16}$	1	0	$-\frac{1}{16}$	0	$\frac{1}{4}$	$\frac{19}{8}$
		$\frac{17}{16}$	0	0	$\frac{7}{16}$	0	$\frac{5}{4}$	$\frac{155}{8}$

For $j = 1$, x_1 must be integer valued, and we have

$$\tfrac{3}{4} = 0 + \tfrac{3}{4} \quad \text{or} \quad g_{11} = \tfrac{3}{4} > f_1.$$

Therefore,

$$d_1 = \frac{\frac{1}{2}}{\frac{1}{2} - 1}\left(\tfrac{3}{4} - 1\right) = \frac{\frac{1}{2}}{-\frac{1}{2}}\left(-\tfrac{1}{4}\right) = \tfrac{1}{4}.$$

For $j = 4$, x_4 must be integer valued, and we have

$$\tfrac{1}{4} = 0 + \tfrac{1}{4} \quad \text{or} \quad g_{14} = \tfrac{1}{4} < f_1.$$

Therefore,

$$d_4 = \tfrac{1}{4}.$$

For $j = 6$, x_6 may have any value, $t_{16} = 0$, and therefore

$$d_6 = 0.$$

The cutting plane constraint is

$$u_1 = -\tfrac{1}{2} + \tfrac{1}{4}x_1 + \tfrac{1}{4}x_4$$

or

$$-\tfrac{1}{4}x_1 - \tfrac{1}{4}x_4 + u_1 = -\tfrac{1}{2}.$$

We add this constraint to Tableau 4.8 to get Tableau 4.9. This tableau represents an optimal but infeasible solution to the related linear programming problem. We apply the dual simplex method to restore feasibility. The result is shown in Tableau 4.10, which yields the optimal solution

$$x_1 = 0, \quad x_2 = \tfrac{5}{2}, \quad x_3 = 2, \quad z = \tfrac{37}{2}.$$

Tableau 4.9

	c_B		4 x_1	5 x_2	3 x_3	0 x_4	0 x_5	0 x_6	0 u_1	x_B
	3	x_3	$\frac{3}{4}$	0	1	$\frac{1}{4}$	0	0	0	$\frac{5}{2}$
	0	x_5	$\frac{11}{16}$	0	0	$-\frac{3}{16}$	1	$-\frac{1}{4}$	0	$\frac{17}{8}$
	5	x_2	$\frac{9}{16}$	1	0	$-\frac{1}{16}$	0	$\frac{1}{4}$	0	$\frac{19}{8}$
←	0	u_1	$-\frac{1}{4}$	0	0	$(-\frac{1}{4})$	0	0	1	$-\frac{1}{2}$
			$\frac{17}{16}$	0	0	$\frac{7}{16}$	0	$\frac{5}{4}$	0	$\frac{155}{8}$

↓

Tableau 4.10

	c_B		4 x_1	5 x_2	3 x_3	0 x_4	0 x_5	0 x_6	0 u_1	x_B
	3	x_3	$\frac{1}{2}$	0	1	0	0	0	1	2
	0	x_5	$\frac{7}{8}$	0	0	0	1	$-\frac{1}{4}$	$-\frac{3}{4}$	$\frac{5}{2}$
	5	x_2	$\frac{5}{8}$	1	0	0	0	$\frac{1}{4}$	$-\frac{1}{4}$	$\frac{5}{2}$
	0	x_4	1	0	0	1	0	0	−4	2
			$\frac{5}{8}$	0	0	0	0	$\frac{5}{4}$	$\frac{7}{4}$	$\frac{37}{2}$

△

4.2 EXERCISES

In Exercises 1 and 2 assume that every variable is constrained to be an integer. Using the given final simplex tableau, find the equation of the cutting plane.

1.

c_B		2 x_1	3 x_2	1 x_3	0 x_4	0 x_5	x_B
0	x_4	0	0	$-\frac{1}{8}$	1	$-\frac{1}{2}$	$\frac{5}{2}$
3	x_2	0	1	$\frac{1}{2}$	0	1	$\frac{7}{8}$
2	x_1	1	0	$\frac{3}{8}$	0	$\frac{1}{4}$	$\frac{13}{8}$
		0	0	$\frac{5}{4}$	0	$\frac{7}{2}$	$\frac{47}{8}$

2.

c_B		2 x_1	3 x_2	4 x_3	1 x_4	0 x_5	0 x_6	0 x_7	x_B
3	x_2	$-\frac{5}{24}$	1	0	0	$\frac{1}{12}$	$\frac{1}{2}$	$\frac{9}{25}$	2
1	x_4	$\frac{11}{24}$	0	0	1	$-\frac{1}{3}$	1	$-\frac{3}{4}$	$\frac{7}{24}$
4	x_3	1	0	1	0	$\frac{3}{4}$	$\frac{11}{12}$	$\frac{13}{12}$	$\frac{1}{3}$
		$\frac{11}{6}$	0	0	0	$\frac{35}{12}$	$\frac{37}{6}$	$\frac{29}{6}$	$\frac{183}{8}$

In Exercises 3 and 4 solve the given integer programming problem by the cutting plane method and sketch the graph of the set of feasible solutions and the cutting planes.

3. Maximize $z = x + y$
subject to
$$2x + 3y \leq 12$$
$$2x + y \leq 6$$
$$x \geq 0, \quad y \geq 0, \quad \text{integers.}$$

4. Maximize $z = x + 4y$
subject to
$$x + 6y \leq 36$$
$$3x + 8y \leq 60$$
$$x \geq 0, \quad y \geq 0, \quad \text{integers.}$$

In Exercises 5–12 solve the given integer programming problem using the cutting plane algorithm.

5. Maximize $z = 4x + y$
subject to
$$3x + 2y \leq 5$$
$$2x + 6y \leq 7$$
$$3x + 7y \leq 6$$
$$x \geq 0, \quad y \geq 0, \quad \text{integers.}$$

6. Maximize $z = x_2 + 4x_3$
subject to
$$3x_1 - 6x_2 + 9x_3 \leq 9$$
$$3x_1 + 2x_2 + x_3 \leq 7$$
$$x_1 \geq 0, \quad x_2 \geq 0, \quad x_3 \geq 0, \quad \text{integers.}$$

7. Maximize $z = 5x + 2y$
subject to
$$12x - 7y \leq 84$$
$$6x + 10y \leq 69$$
$$x \geq 0, \quad y \geq 0, \quad \text{integers.}$$

8. Maximize $z = x_1 + 2x_2 + x_3 + x_4$
subject to
$$2x_1 + x_2 + 3x_3 + x_4 \leq 8$$
$$2x_1 + 3x_2 + 4x_4 \leq 12$$
$$3x_1 + x_2 + 2x_3 \leq 18$$
$$x_j \geq 0, \quad j = 1,2,3,4, \quad \text{integers.}$$

9. Repeat Exercise 3 under the assumption that only x must be an integer.
10. Repeat Exercise 6 under the assumption that only x_1 and x_3 must be integers.
11. Repeat Exercise 8 under the assumption that only x_1 and x_2 must be integers.
12. Repeat Exercise 6 under the assumption that only x_2 must be an integer.

4.2 PROJECT

In Step 2 of the procedure for generating a cutting plane constraint, we gave the following heuristic rule for choosing the variable to be affected by the cutting plane: *from among the basic variables constrained to be integers, choose the basic variable whose value has the largest fractional part.* This arbitrary rule was included so that there was an unambiguous procedure for developing the cutting plane constraint. But other rules are possible. Consider the following problem.

Maximize $z = 5x + 2y$
subject to
$$6x - 15y \leq 24$$
$$6x + 10y \leq 69$$
$$3x + 10y \leq 60$$
$$x \geq 0, \quad y \geq 0, \quad \text{integers.}$$

(a) Sketch a graph of the feasible region of the associated linear programming problem and identify the feasible and optimal solutions of the given integer programming problem.

(b) Solve the integer programming problem using the heuristic rule as given in Step 2. (*Warning:* the computation is lengthy.)

(c) Solve the integer programming problem using the following as the heuristic rule: *from among the basic variables constrained to be integers, choose the basic variable whose value has the smallest fractional part.*

(d) What other rules can you propose for choosing the basic variable to be affected by the cutting plane?

4.3 BRANCH AND BOUND METHODS

If the set of feasible solutions to the related linear programming problem of a mixed integer programming problem is bounded, then the integer valued variables can take on only finitely many values in this region. We had previously discussed solving a linear programming problem by enumerating the extreme points of the set of feasible solutions and then choosing an optimal solution from among these. We dismissed this brute force method in favor of the simplex algorithm that listed only some of the extreme points and chose an order for the list in which the value of the objective function improved each time. However, using cutting planes and the simplex algorithm can involve very lengthy computations. Thus, it may be advantageous to again consider some enumerating technique.

We examine the possibility of cleverly enumerating the integer values that should be considered for a mixed integer programming problem. The cleverness is needed so that the task does not become overwhelming. One technique that is used is that of implicit enumeration. This involves generating a list of some of the feasible integral solutions and saving the best solution in the list for comparison with lists that will be generated subsequently.

A set S of feasible solutions to a mixed integer programming problem is said to be **implicitly enumerated** if S does not contain any solution that is better than the best currently known solution. Our strategy will be to partition the set of feasible solutions into several subsets and then to dismiss many of these subsets because they are implicitly enumerated. We can derive certain relations from the constraints of the problem and the value of the best current solution. These relations will be violated by any set of solutions that is implicitly enumerated; that is, these relations give conditions that are necessary for a solution to satisfy if it is to improve the value of the objective function. If many subsets of the set of feasible solutions can be implicitly enumerated, we can greatly limit the number of solutions that have to be explicitly examined.

It will be easiest to describe the enumeration technique if we initially limit ourselves to a zero–one programming problem. Consider such a problem and assume that it has n variables. The set of all feasible

solutions, S, can be partitioned into two subsets, S_0 and S_1, where

$$S_0 = \{\mathbf{x} \in S \mid x_1 = 0\}$$

and

$$S_1 = \{\mathbf{x} \in S \mid x_1 = 1\}.$$

Likewise, both S_0 and S_1 can be partitioned into two subsets—say, S_0 is divided into S_{00} and S_{01}, and S_1 is divided into S_{10} and S_{11}. We define S_{00} by

$$S_{00} = \{\mathbf{x} \in S \mid x_1 = 0 \quad \text{and} \quad x_2 = 0\}.$$

Similarly,

$$S_{01} = \{\mathbf{x} \in S \mid x_1 = 0 \quad \text{and} \quad x_2 = 1\}$$
$$S_{10} = \{\mathbf{x} \in S \mid x_1 = 1 \quad \text{and} \quad x_2 = 0\}$$
$$S_{11} = \{\mathbf{x} \in S \mid x_1 = 1 \quad \text{and} \quad x_2 = 1\}.$$

Each of the four subsets S_{ij}, $i, j = 0$ or 1, can be partitioned further. We can describe this procedure with a tree diagram, as shown in Figure 4.2. The numbered circles are called **nodes**; the lines connecting them are called **branches**. Node 1 represents the set S of all feasible solutions. The partitioning of S into S_0 and S_1 is represented by the branches leading to nodes 2 and 3. Node 2 represents S_0 and node 3 represents S_1.

A sequence of nodes and branches from node 1 to any other node k is called a **path** to node k. Each branch represents the imposition of one constraint on the variables (setting one variable equal to 0 or 1). Node k

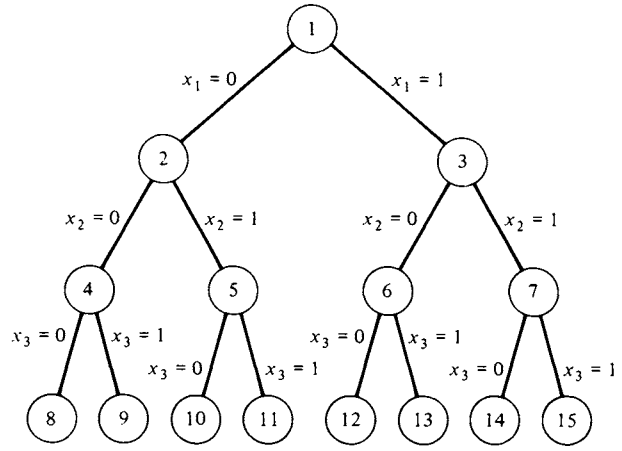


FIGURE 4.2

represents the set of all solutions to the original constraints that also satisfy the constraints imposed by the branches in the path to node k. For example, node 9 represents

$$\{\mathbf{x} \in S \mid x_1 = 0, \quad x_2 = 0, \quad x_3 = 1\}.$$

If two nodes are connected by a path, the lower node represents a subset of the solutions that are represented by the higher node.

If all possible values for the n variables in a zero–one programming problem are enumerated, we have to generate 2^n paths. The bottom node of each path would correspond to exactly one value for $\mathbf{x}$, and this value may be infeasible depending on the nature of the constraints. Our enumeration strategy will try to eliminate as many of these paths as possible from consideration. We describe one method that might be used to perform this elimination.

Suppose our zero–one programming problem has an optimal solution for which the value of the objective function is z^*. Assume that we have found a feasible solution to the problem that has an objective function value of z_L. Obviously, $z_L \leq z^*$. Keep in mind that z^* is unknown and is precisely what we are trying to find. At this point we know that we do not have to consider any set of solutions whose objective function values are smaller than z_L. Consider the set of solutions to the zero–one problem that is represented by node k. We can find an upper bound z_U^k for the values of the objective function at the solutions represented by node k by solving a related linear programming problem. If

$$z_U^k \leq z_L,$$

then no path that includes node k will yield an improved solution; the objective function values will all be no larger than z_U^k. Consequently, we can eliminate all paths through node k without explicitly evaluating the objective function at each solution on the path. This set of solutions has been implicitly enumerated. In this case, node k is called a **terminal node**.

Node k is also called a terminal node if there are no feasible solutions satisfying the constraints imposed at node k. Besides the original constraints and those imposed by the path leading to node k, we often add the constraint

$$\mathbf{c}^T\mathbf{x} \geq z_L$$

since we are interested only in solutions that are better than the current best feasible solution. Finally, node k is called a terminal node if it represents a single feasible solution—that is, if all the variables have been assigned a unique value.

Both branch and bound methods and methods that search in the tree of solutions generate the nodes of the tree until all paths end in terminal

nodes. At that point all solutions will have been explicitly or implicitly enumerated and the best solution found is an optimal solution. The difference between the two types of methods is the manner in which the nodes are generated. Search methods follow a path until it ends at a terminal node before examining another path. Branch and bound methods examine the nodes in a less straightforward manner. They may generate, more or less simultaneously, many different paths. Branch and bound methods use more computer storage than search methods do, but they have much more flexibility in examining the nodes and thus may be faster.

We will give the details of one branch and bound method for solving an arbitrary mixed integer programming problem. This method was developed in 1966 by R. J. Dakin and is a modification of the Land–Doig method. It is widely used in the computer codes for integer programming.

Consider the mixed integer programming problem

$$\left.\begin{array}{rl} \text{Maximize} & z = \mathbf{c}^{\mathrm{T}}\mathbf{x} \\ \text{subject to} & \\ & \mathbf{A}\mathbf{x} = \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0} \\ x_j, j \in I, & \text{integral,} \end{array}\right\} \tag{1}$$

where $\mathbf{A}$ is $m \times s$, $\mathbf{b}$ is $m \times 1$, $\mathbf{c}$ is $s \times 1$, and $\mathbf{x}$ is $s \times 1$.

Dakin's method will generate a tree very similar to the one we formed for the zero–one programming problem. For each node there will be two branches to check. If neither of the branches ends in a terminal node, the method follows the most promising branch. The other branch is called **dangling** and must be examined before the algorithm terminates. Dakin's method proceeds as follows:

Step 1 (Initial Solution). Solve the problem given by (1) as a linear programming problem, ignoring the integrality restrictions. If all x_j, $j \in I$, have integral values, we are done. If not, go to Step 2.

Step 2 (Branching Variable Selection). Choose, from among those variables, x_j, $j \in I$, that do not have integral values at this node, one variable to be used to form the branching constraints. An easily implemented rule for this choice is to use the variable whose value has the largest fractional part. There are other more complicated rules that may even involve one iteration of the dual simplex method to determine which variable to choose. The x_j selected must be a basic variable; otherwise, its value would be zero. Suppose it is the ith basic variable in the final tableau for the node, so that its value is x_{Bi}. We can write

$$x_{Bi} = [x_{Bi}] + f_i,$$

where $0 < f_i < 1$. Since x_j must have an integral value, it must satisfy either

$$x_j \leq [x_{Bi}] \tag{2}$$

or

$$x_j \geq [x_{Bi}] + 1. \tag{3}$$

Step 3 (Formation of New Nodes). We create two new mixed integer problems represented by the node under consideration in Step 2. One problem is formed by adding constraint (2) and the other problem is formed by adding constraint (3). Solve each of these problems as a linear programming problem using the dual simplex method.

Step 4 (Test for Terminal Node). Each of the nodes formed in Step 3 may be a terminal node for one of two reasons. First, the problem represented by the node may have no feasible solutions. Or the values of x_j, $j \in I$, are all integers. In the former case, label the nodes as terminal nodes and go to Step 5. In the latter case, besides labeling the nodes, compare the values of the objective function with the current best value. If the objective function value for the new node is better, replace the current best value with it. Go to Step 5.

Step 5 (Node Selection).

(a) If both nodes were terminal nodes in Step 4, the next node to be considered is the next one on the list of dangling nodes. If this dangling node has an objective function value larger than the current best value, then use this node and go to Step 2. Otherwise, check the next node in the list of dangling nodes. When the list of dangling nodes is exhausted, stop. The current best value is the optimal solution.

(b) If exactly one node in Step 4 was terminal, use the nonterminal node and go to Step 2.

(c) If both nodes in Step 4 were nonterminal, we choose the more promising one. Usually the node with the largest objective function value is considered more promising. The other node is recorded in the list of dangling nodes to be considered later.

EXAMPLE 1. Consider the pure integer programming problem

$$\begin{aligned} &\text{Maximize} \quad z = 7x_1 + 3x_2 \\ &\text{subject to} \\ &\qquad 2x_1 + 5x_2 \leq 30 \\ &\qquad 8x_1 + 3x_2 \leq 48 \\ &x_1 \geq 0, \quad x_2 \geq 0, \quad \text{integers.} \end{aligned}$$

We solve the related linear programming problem and obtain the final tableau shown in Tableau 4.11. From this tableau we see that an optimal solution is

$$x_1 = 4\tfrac{7}{17}, \quad x_2 = 4\tfrac{4}{17}, \quad z = 43\tfrac{10}{17}.$$

We choose x_1 as the branching variable, since it has the largest fractional part. The constraints to be added are

$$x_1 \leq 4 \quad \text{and} \quad x_1 \geq 5.$$

Tableau 4.11

c_B		7 x_1	3 x_2	0 x_3	0 x_4	x_B
3	x_2	0	1	$\frac{4}{17}$	$-\frac{1}{17}$	$\frac{72}{17}$
7	x_1	1	0	$-\frac{3}{34}$	$\frac{5}{34}$	$\frac{75}{17}$
		0	0	$\frac{3}{34}$	$\frac{29}{34}$	$\frac{741}{17}$

To carry out Step 3 we add each of these constraints in turn to the final tableau in Step 1. We get the tableaux shown in Tableaux 4.12 and 4.13. For Tableau 4.12 we have written

$$x_1 = \tfrac{75}{17} + \tfrac{3}{34}x_3 - \tfrac{5}{34}x_4$$

from the second row of Tableau 4.11 and then introduced a slack variable, u_1, so that our new constraint is

$$x_1 + u_1 = 4$$

or

$$\tfrac{3}{34}x_3 - \tfrac{5}{34}x_4 + u_1 = 4 - \tfrac{75}{17} = -\tfrac{7}{17}.$$

In the same way the new constraint for Tableau 4.13 becomes

$$-\tfrac{3}{34}x_3 + \tfrac{5}{34}x_4 + u_1 = -5 + \tfrac{75}{17} = -\tfrac{10}{17}.$$

Tableau 4.12

	c_B		7 x_1	3 x_2	0 x_3	0 ↓ x_4	0 u_1	x_B
	3	x_2	0	1	$\frac{4}{17}$	$-\frac{1}{17}$	0	$\frac{72}{17}$
	7	x_1	1	0	$-\frac{3}{34}$	$\frac{5}{34}$	0	$\frac{75}{17}$
←	0	u_1	0	0	$\frac{3}{34}$	$\left(-\frac{5}{34}\right)$	1	$-\frac{7}{17}$
			0	0	$\frac{3}{34}$	$\frac{29}{34}$	0	$\frac{741}{17}$

Tableau 4.13

	c_B		7 x_1	3 x_2	0 x_3	0 x_4	0 u_1	x_B
	3	x_2	0	1	$\frac{4}{17}$	$-\frac{1}{17}$	0	$\frac{72}{17}$
	7	x_1	1	0	$-\frac{3}{34}$	$\frac{5}{34}$	0	$\frac{75}{17}$
←	0	u_1	0	0	$(-\frac{3}{34})$	$\frac{5}{34}$	1	$-\frac{10}{17}$
			0	0	$\frac{3}{34}$	$\frac{29}{34}$	0	$\frac{741}{17}$

We now apply the dual simplex method to each of these tableaux to obtain Tableaux 4.14 and 4.15, respectively.

Tableau 4.14

	c_B		7 x_1	3 x_2	0 x_3	0 x_4	0 u_1	x_B
	3	x_2	0	1	$\frac{1}{5}$	0	$-\frac{2}{5}$	$\frac{22}{5}$
	7	x_1	1	0	0	0	1	4
	0	x_4	0	0	$-\frac{3}{5}$	1	$-\frac{34}{5}$	$\frac{14}{5}$
			0	0	$\frac{3}{5}$	0	$\frac{29}{5}$	$\frac{206}{5}$

Tableau 4.15

	c_B		7 x_1	3 x_2	0 x_3	0 x_4	0 u_1	x_B
	3	x_2	0	1	0	$\frac{1}{3}$	$\frac{8}{3}$	$\frac{8}{3}$
	7	x_1	1	0	0	0	-1	5
	0	x_3	0	0	1	$-\frac{5}{3}$	$-\frac{34}{3}$	$\frac{20}{3}$
			0	0	0	1	1	43

At this point we have the tree in Figure 4.3. The objective function value in node 3 is larger than that in node 2. Consequently, node 2 is recorded in our list of dangling nodes and we continue from node 3. We add the constraints

$$x_1 \leq 2 \quad \text{and} \quad x_2 \geq 3$$

to the problem represented by node 3 to form two new problems. As before, we write

$$x_2 = -\tfrac{1}{3}x_4 - \tfrac{8}{3}u_1 + \tfrac{8}{3}$$

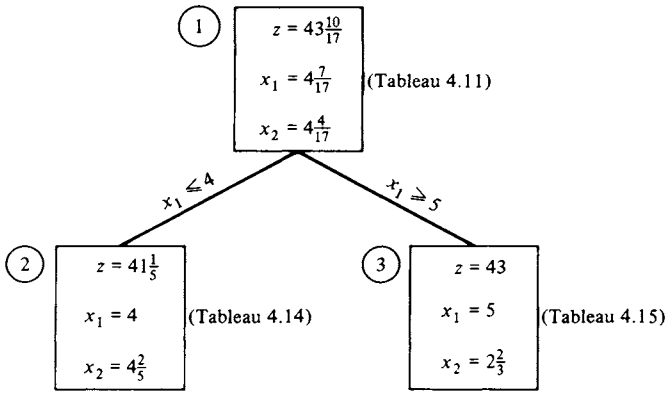


FIGURE 4.3

from the first row of Tableau 4.15. Then by adding a slack variable to each of the constraints, we have

$$x_2 + u_2 = 2 \quad \text{and} \quad -x_2 + u_2 = -3$$

or

$$-\tfrac{1}{3}x_4 - \tfrac{8}{3}u_1 + u_2 = 2 - \tfrac{8}{3} = -\tfrac{2}{3} \tag{4}$$

and

$$\tfrac{1}{3}x_4 + \tfrac{8}{3}u_1 + u_2 = -3 + \tfrac{8}{3} = -\tfrac{1}{3}. \tag{5}$$

We add each of constraints (4) and (5) to Tableau 4.15 to form Tableaux 4.16 and 4.17. We use the dual simplex method on each of these tableaux to try to restore feasibility. Tableau 4.18 is obtained from Tableau 4.16 by this process. However, Tableau 4.17 represents an infeasible solution that satisfies the optimality criterion, because there are no negative entries in the pivotal row (labeled by u_2). Thus, Tableau 4.17 represents a terminal node.

Tableau 4.16

			7	3	0	0	0	0	
	c_B		x_1	x_2	x_3	x_4	u_1	u_2	x_B
	3	x_2	0	1	0	$\frac{1}{3}$	$\frac{8}{3}$	0	$\frac{8}{3}$
	7	x_1	1	0	0	0	-1	0	5
	0	x_3	0	0	1	$-\frac{5}{3}$	$-\frac{34}{3}$	0	$\frac{20}{3}$
←	0	u_2	0	0	0	$-\frac{1}{3}$	$-\frac{8}{3}$	1	$-\frac{2}{3}$
			0	0	0	1	1	0	43

↓

Tableau 4.17

c_B		7 x_1	3 x_2	0 x_3	0 x_4	0 u_1	0 u_2	x_B
3	x_2	0	1	0	$\frac{1}{3}$	$\frac{8}{3}$	0	$\frac{8}{3}$
7	x_1	1	0	0	0	-1	0	5
0	x_3	0	0	1	$-\frac{5}{3}$	$-\frac{34}{3}$	0	$\frac{20}{3}$
0	u_2	0	0	0	$\frac{1}{3}$	$\frac{8}{3}$	1	$-\frac{1}{3}$
		0	0	0	1	1	0	43

Tableau 4.18

c_B		7 x_1	3 x_2	0 x_3	0 x_4	0 u_1	0 u_2	x_B
3	x_2	0	1	0	0	0	1	2
7	x_1	1	0	0	$\frac{1}{8}$	0	$-\frac{3}{8}$	$\frac{21}{4}$
0	x_3	0	0	1	$-\frac{1}{4}$	0	$-\frac{17}{4}$	$\frac{19}{2}$
0	u_1	0	0	0	$\frac{1}{8}$	1	$-\frac{3}{8}$	$\frac{1}{4}$
		0	0	0	$\frac{7}{8}$	0	$\frac{3}{8}$	$\frac{171}{4}$

At this point we have the tree in Figure 4.4. Note that in node 3, x_1 had an integer value, but in node 4, the value of x_1 is no longer an integer. Thus, we cannot expect a variable once it has an integer value to continue having an integer value.

We now use Step 2 on node 4, adding the constraints

$$x_1 \leq 5 \quad \text{and} \quad x_1 \geq 6.$$

Tableau 4.19

	c_B		7 x_1	3 x_2	0 x_3	0 x_4 ↓	0 u_1	0 u_2	0 u_3	x_B
	3	x_2	0	1	0	0	0	1	0	2
	7	x_1	1	0	0	$\frac{1}{8}$	0	$-\frac{3}{8}$	0	$\frac{21}{4}$
	0	x_3	0	0	1	$-\frac{1}{4}$	0	$-\frac{17}{4}$	0	$\frac{19}{2}$
	0	u_1	0	0	0	$\frac{1}{8}$	1	$-\frac{3}{8}$	0	$\frac{1}{4}$
←	0	u_3	0	0	0	$\left(-\frac{1}{8}\right)$	0	$\frac{3}{8}$	1	$-\frac{1}{4}$
			0	0	0	$\frac{7}{8}$	0	$\frac{3}{8}$	0	$\frac{171}{4}$

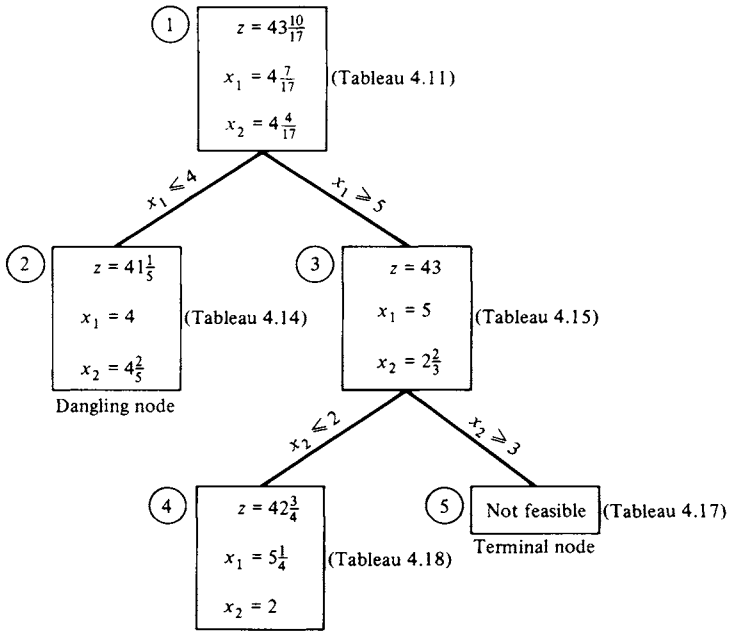


FIGURE 4.4

Tableau 4.20

	c_B		7 x_1	3 x_2	0 x_3	0 x_4	0 u_1	0 u_2 ↓	0 u_3	x_B
	3	x_2	0	1	0	0	0	1	0	2
	7	x_1	1	0	0	$\frac{1}{8}$	0	$-\frac{3}{8}$	0	$\frac{21}{4}$
	0	x_3	0	0	1	$-\frac{1}{4}$	0	$-\frac{17}{4}$	0	$\frac{19}{2}$
	0	u_1	0	0	0	$\frac{1}{8}$	1	$-\frac{3}{8}$	0	$\frac{1}{4}$
←	0	u_3	0	0	0	$\frac{1}{8}$	0	$(-\frac{3}{8})$	1	$-\frac{3}{4}$
			0	0	0	$\frac{7}{8}$	0	$\frac{3}{8}$	0	$\frac{171}{4}$

We obtain Tableaux 4.19 and 4.20 after introducing the slack variable u_3 and rewriting these constraints. We use the dual simplex algorithm on each of these tableaux to try to restore feasibility. Tableau 4.21 is obtained from Tableau 4.19, and Tableau 4.22 is obtained from Tableau 4.20 by this process. Since both Tableaux 4.21 and 4.22 give integer solutions, they correspond to terminal nodes. The solution

$$x_1 = 6, \quad x_2 = 0, \quad z = 42$$

Tableau 4.21

c_B		7 x_1	3 x_2	0 x_3	0 x_4	0 u_1	0 u_2	0 u_3	x_B
3	x_2	0	1	0	0	0	1	0	2
7	x_1	1	0	0	0	0	0	1	5
0	x_3	0	0	1	0	0	−5	−2	10
0	u_1	0	0	0	0	1	0	1	0
0	x_4	0	0	0	1	0	−3	−8	2
		0	0	0	0	0	3	7	41

Tableau 4.22

c_B		7 x_1	3 x_2	0 x_3	0 x_4	0 u_1	0 u_2	0 u_3	x_B
3	x_2	0	1	0	$\frac{1}{3}$	0	0	$\frac{8}{3}$	0
7	x_1	1	0	0	0	0	0	−1	6
0	x_3	0	0	1	$-\frac{5}{3}$	0	0	$-\frac{34}{3}$	18
0	u_1	0	0	0	0	1	0	−1	1
0	u_2	0	0	0	$-\frac{1}{3}$	0	1	$-\frac{8}{3}$	2
		0	0	0	1	0	0	1	42

from Tableau 4.22 is the better one, since it gives a larger objective function value. We then check the list of dangling nodes to see whether other branches of the tree must be pursued. The only dangling node has the objective function value $z = 41\frac{1}{5}$, which is smaller than the value obtained in Tableau 4.22. Branching at this dangling node will not increase the objective function value. Therefore, we have found an optimal solution to the original problem:

$$x_1 = 6, \quad x_2 = 0, \quad z = 42.$$

The tree of problems that was constructed by the branch and bound algorithm for this example is given in Figure 4.5. △

The following problem shows an application of the branch and bound algorithm to a mixed integer programming problem.

EXAMPLE 2. Consider the mixed integer programming problem

$$\text{Maximize} \quad z = 2x_1 + x_2 + 3x_3$$

subject to

$$4x_1 + 3x_2 - 3x_2 \leq 6$$

$$2x_1 + 3x_2 + 3x_3 \leq 4$$

$$x_j \geq 0, \quad j = 1, 2, 3; \quad x_1, x_3, \quad \text{integers}.$$

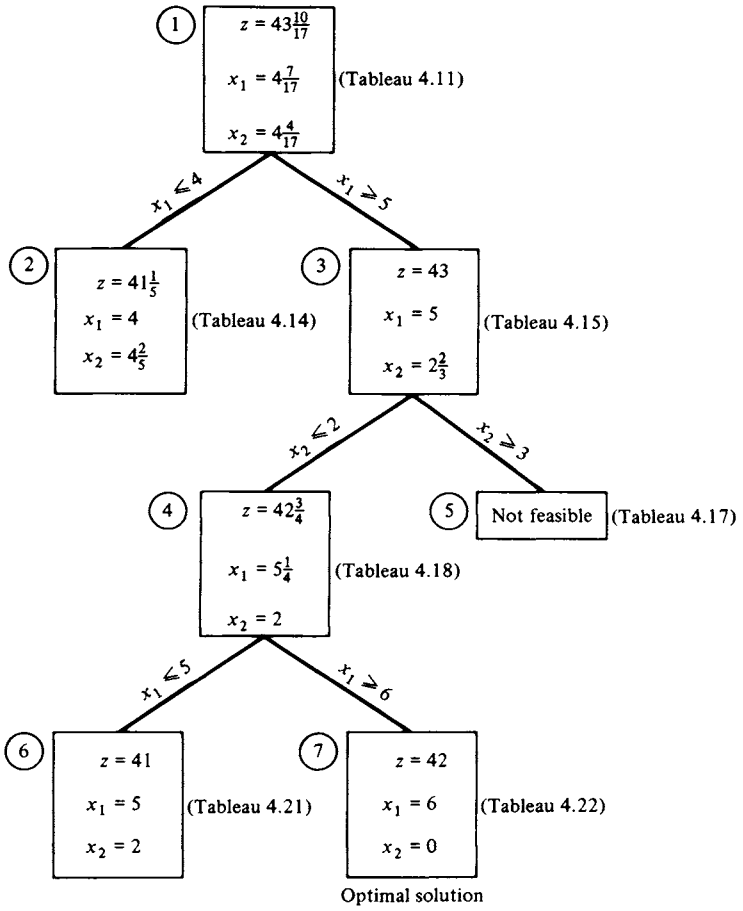


FIGURE 4.5

Solution. The branch and bound method yields the tree shown in Figure 4.6. Observe that node 4 represents a possible solution to the problem since x_1 and x_3 both have integer values. However, node 5 gives an objective function value that is larger than the one in node 4, so we might have a solution in which x_1 and x_3 have integer values with an objective function value greater than $3\frac{1}{3}$. Thus, we must generate nodes 6 and 7. We do not have to go beyond node 6 since its objective function value is smaller than that of node 4 and any further branching will not increase the value of the objective function. Thus, node 4 represents the optimal solution to the given problem:

$$x_1 = 0, \quad x_2 = \tfrac{1}{3}, \quad x_3 = 1, \quad z = 3\tfrac{1}{3}. \qquad \triangle$$

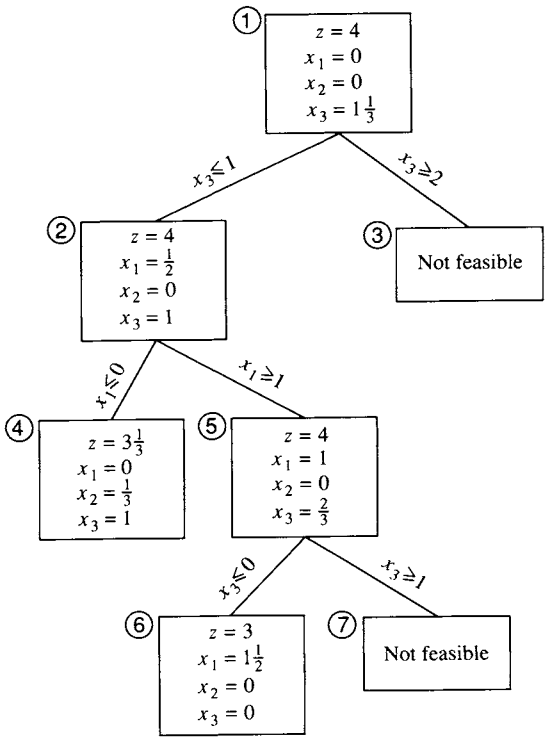


FIGURE 4.6

4.3 EXERCISES

In Exercises 1–10 solve the indicated mixed integer programming problem in Section 4.2 and draw the tree of problems used in finding the solution.

1. Exercise 3
2. Exercise 4
3. Exercise 5
4. Exercise 6
5. Exercise 7
6. Exercise 8
7. Exercise 9
8. Exercise 10
9. Exercise 11
10. Exercise 12

In Exercises 11–14, use either the cutting plane method or the branch and bound method to solve the indicated mixed integer programming problem in Section 4.1.

11. Exercise 1
12. Exercise 2
13. Exercise 4
14. Exercise 5 (If a computer program is available for solving linear programming problems, make use of it.)

4.3 PROJECT

Consider the air filter manufacturer in Section 3.5, Project 1. Suppose that there is an additional specification for the purposes of standardization: the diameters of the main chamber and exit duct must be integer values. Rework the problem with this additional specification and comment on its economic impact on the company.

4.4 COMPUTER ASPECTS (OPTIONAL)

The computation of a solution to a mixed integer programming problem can be an extremely difficult task. We have discussed two algorithms for finding such a solution. Experimental evidence indicates, however, that there are integer programming problems for which each of these algorithms would be very inefficient. There have been many other algorithms reported in the literature, along with several papers comparing their effectiveness. Most researchers now feel that there will never be a universally good algorithm for mixed integer programming. This is perhaps disappointing, because in the linear programming case the simplex algorithm with modifications to make it numerically stable is a universal algorithm, and one might expect that restricting the range of some of the variables to integer values would simplify the computations. In fact, just the opposite happens.

The state of the art for computational methods for integer programming is as follows. There are commercially available codes, just as for the simplex method, virtually all of which use a branch and bound technique. At various universities there are also experimental implementations of various modifications of the branch and bound algorithm and of cutting plane techniques. Several authors have proposed lists of problems against which one can test an algorithm. In running these tests, it is found that each algorithm is successful on some problems but no algorithm is success-

ful on all problems. Worse than this is the discovery that by simply renumbering the variables or reordering the constraints of a problem, an algorithm that solved the original problem well may perform poorly on the reformulated problem.

Recently an extremely important development has taken place in the area of preprocessing a mixed integer programming problem to reduce its complexity. The problem and an initial solution are assessed as to their closeness to be a pure integer model. This approach has succeeded in turning many difficult problems into manageable ones. For example, Ketron Management System's MIPIII does extensive preprocessing using these ideas.

The size of a mixed integer programming problem that can be successfully solved in a reasonable amount of computer time is much smaller than that for a linear programming problem. Mixed integer problems that have thousands of integer variables, hundreds of thousands of continuous variables, and tens of thousands of constraints can be solved successfully. Of course, as the computational speed of new computers increases, larger problems will become feasible.

To be a successful problem solver of mixed integer programming problems then, one must be able to do much more than provide data to a computer program. First, one must be skilled at formulating problems and must know alternative formulations so that when the problem is given to the computer it will be in a form on which the code will work efficiently. Several factors that should be considered while constructing the model are as follows.

(a) Having a large number of variables constrained to be integers may cause the solution procedure to be very complex and lengthy. Each integer variable can cause branches in a branch and bound algorithm. A large number of these variables will cause a very large tree of problems to be constructed. It will require an extensive amount of storage for the tree and a large amount of time to process the branches. One way of limiting the number of integer variables is to agree that an almost optimum solution will be acceptable. Having a solution 10% less than optimal would not be bad if uncertainties in input parameters would cause almost that much deviation anyway. After settling for an almost optimal solution, one can argue that variables with large values need not be constrained to be integers. Simply take the noninteger values for these variables and round off. Most likely, rounding off will not change the objective function value greatly. For example, if a variable represents the number of workers but its magnitude will be at least 1000, allowing it to be noninteger and rounding off will make an error of less than 0.05% in the variable, and if the model is well formulated, this error should not propagate to a very large error in

the objective function. In fact, some suggest that any variable whose value will be larger than 20 should not be constrained to be an integer.

(b) Tightly bounding integer variables will allow a branch and bound algorithm to find terminal nodes more quickly as the extra constraints become inconsistent with the bounds. Geometrically, the tight bounds reduce the number of lattice points within the convex set of feasible solutions to the related linear programming problem that must be considered.

(c) It helps to formulate the problem as a network problem or special-type (transportation, assignment) integer programming problem to take advantage of the special algorithms that are available. These special algorithms exist because they are more efficient than the general algorithm on a particular type of problem. Even though the combinatorial problems arising in network theory can be difficult, it is generally better to make use of the additional information or structure that the problem has.

Second, the user should be prepared to monitor the progress of the computer solution by dividing the problem into stages and carefully interpreting the output from each stage. If the user suspects that a particular algorithm is not proceeding quickly enough to the solution, he or she may choose to change algorithms for a few iterations. Or perhaps the structure of the problem and the output will indicate that a different selection of decision rules for the algorithm will speed the process.

Finally, the user should be familiar with the heuristic methods that are applicable to his or her type of problem. Perhaps one of these methods will determine a good starting solution or good decision rules.

Further Reading

Dakin, N. J. "A Tree Search Algorithm for Mixed Integer Programming Problems." *Comput. J.* **9** (1966), 250–255.

Garfinkel, Robert S., and Nemhauser, George L. *Integer Programming*. Wiley, New York, 1972.

Geoffrion, A. M., and Marsten, R. E. "Integer Programming Algorithms: A Framework and State-of-the-Art Survey." *Management Sci.* **18** (1972), 465–491.

Gilmore, P. C., and Gomory, Ralph E. "A Linear Programming Approach to the Cutting Stock Problem." *Operations Res.* **9** (1961), 849–859.

Gilmore, P. C., and Gomory, Ralph E. "A Linear Programming Approach to the Cutting Stock Problem—Part II." *Operations Res.* **11** (1963), 863–888.

Gilmore, P. C., and Gomory, Ralph E. "Multistage Cutting Stock Problems of Two or More Dimensions." *Operations Res.* **13** (1965), 94–120.

Kastning, Claus. *Integer Programming and Related Areas: A Classified Bibliography*. Lecture Notes in Economics and Mathematical Systems, No. 128. Springer-Verlag, New York, 1976.

Lawler, Eugene L. (Ed.). *The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization*. Wiley, New York, 1985.

Nemhauser, George L., and Wolsey, Laurence A. *Integer and Combinatorial Optimization.* Wiley, New York, 1988.

von Randow, Rabe, (Ed.). *Integer Programming and Related Areas: A Classified Bibliography: 1978–1981.* Lecture Notes in Economics and Mathematical Systems, No. 197. Springer-Verlag, New York, 1982.

von Randow, Rabe (Ed.). *Integer Programming and Related Areas: A Classified Bibliography: 1981–1984.* Lecture Notes in Economics and Mathematical Systems, No. 243. Springer-Verlag, New York, 1985.

Salkin, Harvey M. *Integer Programming.* Addison–Wesley, Reading, MA, 1975.

Taha, Hamdy A. *Integer Programming: Theory, Applications.* Academic Press, New York, 1975.

5

Special Types of Linear Programming Problems

IN THIS CHAPTER we study a number of special types of linear programming problems. These problems arise in transportation systems, in communication systems, in pipeline systems, in electrical networks, in the planning and scheduling of projects, and in many other applications. Each of these problems can be formulated and solved as a linear programming problem. However, because of their special structure these problems can be more effectively solved by other methods that have been specifically developed to handle them.

5.1 THE TRANSPORTATION PROBLEM

We have previously considered two examples of the transportation problem—Example 3 in Section 1.1 and Example 1 in Section 4.1. We present another example here that will be used to indicate the ways in which the simplex method can be adapted to the special structure of this problem. Our eventual goal is the transportation algorithm.

EXAMPLE 1. The plastic manufacturing company described in Example 3 of Section 1.1 has decided to build a combination plant–warehouse in San Antonio. This expansion will give it three plants and four warehouses. The following shipping costs have been determined for the new plant and warehouse:

	To			
From	*Los Angeles*	*Chicago*	*New York City*	*San Antonio*
Salt Lake City	—	—	—	6
Denver	—	—	—	5
San Antonio	7	6	8	1

The San Antonio plant can supply 100 tons per week, and the warehouse needs 120 tons per week to meet its demand. We want to determine how many tons of sheet polyethylene should be shipped from each plant to each warehouse to minimize the transportation cost.

MATHEMATICAL MODEL. The cost matrix is given by

$$\mathbf{C} = \begin{matrix} \\ \text{Salt Lake City} \\ \text{Denver} \\ \text{San Antonio,} \end{matrix} \begin{matrix} \begin{matrix} \text{Los} \\ \text{Angeles} \end{matrix} & \text{Chicago} & \begin{matrix} \text{New York} \\ \text{City} \end{matrix} & \begin{matrix} \text{San} \\ \text{Antonio} \end{matrix} \\ \begin{bmatrix} 5 & 7 & 9 & 6 \\ 6 & 7 & 10 & 5 \\ 7 & 6 & 8 & 1 \end{bmatrix} \end{matrix} \tag{1}$$

where we have labeled the rows with the points of origin and the columns with the destinations. The supply vector is

$$\mathbf{s} = \begin{bmatrix} 120 \\ 140 \\ 100 \end{bmatrix}$$

and the demand vector is

$$\mathbf{d} = \begin{bmatrix} 100 \\ 60 \\ 80 \\ 120 \end{bmatrix}.$$

We let x_{ij} denote the amount shipped from the ith plant to the jth warehouse, where the plants and warehouses are numbered as in Equation (1). Our model is to minimize

$$z = \sum_{i=1}^{3} \sum_{j=1}^{4} c_{ij} x_{ij}$$

subject to

$$\left.\begin{aligned} \sum_{j=1}^{4} x_{ij} &\le s_i, \quad i = 1,2,3 \\ \sum_{i=1}^{3} x_{ij} &\ge d_j, \quad j = 1,2,3,4 \end{aligned}\right\} \tag{2}$$

$$x_{ij} \ge 0, \quad \text{integers.}$$

Clearly the company must be able to provide a supply of polyethylene at least equal to the demand. In our example, we have

$$\sum_{i=1}^{3} s_i = 360 = \sum_{j=1}^{4} d_j. \qquad \Delta$$

Later we will indicate what to do if supply exceeds demand. We will say that the model is infeasible if demand exceeds supply. The student may show that, when demand equals supply, each of the constraints in (2) is an equality.

We will now develop an algorithm for solving the general transportation problem

$$\text{Minimize} \quad z = \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} x_{ij}$$

subject to

$$\left.\begin{aligned} \sum_{j=1}^{n} x_{ij} &= s_i, \quad i = 1,2,\ldots,m \\ \sum_{i=1}^{m} x_{ij} &= d_j, \quad j = 1,2,\ldots,n \end{aligned}\right\} \tag{3}$$

$$\sum_{i=1}^{m} s_i = \sum_{j=1}^{n} d_j \tag{4}$$

$$x_{ij} \ge 0, \quad i = 1,2,\ldots,m, \quad j = 1,2,\ldots,n.$$

This form of the transportation problem has the following properties.

1. The problem always has a feasible solution (Exercise 17).
2. The set of feasible solutions is bounded (Exercise 18).
3. Because of (4), one of the $m + n$ constraints in (3) is redundant (Exercise 15).

We may substitute the values for the coefficients in our model and write it as

$$\text{Minimize} \quad z = 5x_{11} + 7x_{12} + 9x_{13} + 6x_{14} + 6x_{21} + 7x_{22} + 10x_{23} + 5x_{24} + 7x_{31} + 6x_{32} + 8x_{33} + x_{34}$$

subject to

$$\left.\begin{aligned}
x_{11} + x_{12} + x_{13} + x_{14} &= 120 \\
x_{21} + x_{22} + x_{23} + x_{24} &= 140 \\
x_{31} + x_{32} + x_{33} + x_{34} &= 100 \\
x_{11} + x_{21} + x_{31} &= 100 \\
x_{12} + x_{22} + x_{32} &= 60 \\
x_{13} + x_{23} + x_{33} &= 80 \\
x_{14} + x_{24} + x_{34} &= 120
\end{aligned}\right\} \tag{5}$$

$$x_{ij} \geq 0, \quad \text{integers}, \quad i = 1,2,3, \quad j = 1,2,3,4.$$

Note that each variable appears in exactly two constraints. Since there are seven constraints in the problem, we expect that there will be seven nonzero variables in a basic feasible solution. Actually, there will be only six nonzero variables in any basic feasible solution, because one constraint is redundant. This redundancy occurs because the supply is equal to the demand.

The model can be solved by the simplex algorithm, and its solutions will always be an integer vector, since the constraint matrix contains only 0's and 1's. However, there is a much more efficient algorithm that uses a 3×4 tableau instead of the 6×12 tableau for the simplex algorithm.

We construct the transportation tableau by writing a 3×4 matrix to hold the values of x_{ij}. The supply vector is placed to the right of this matrix, the transpose of the demand vector is written below the matrix, and the unit costs are written in the insets. For our problem we obtain Tableau 5.1.

Tableau 5.1

5	7	9	6	120
6	7	10	5	140
7	6	8	1	100
100	60	80	120	

Supply

Demand

There are several ways of systematically filling in the tableau with values for x_{ij}. For now, we start by allocating as much of the supply in row 1 (from plant 1) as possible to the cheapest route. This means $x_{11} = 100$, which satisfies the demand in the first column and leaves 20 units to be allocated to the next cheapest route in the first row. We set $x_{14} = 20$, which exhausts the supply from the first plant. Consequently, $x_{12} = x_{13} = 0$. Moving to the second row, we follow the same procedure of allocating as much as possible to the cheapest route. We set $x_{24} = 100$, since 20 units have already been provided in row 1. The 40 remaining units from plant 2 are shipped to warehouse 2, since that route is the cheapest remaining one. The rest of the tableau is filled in using the same reasoning (Tableau 5.2). Observe that the solution represented by Tableau 5.2 is a basic feasible solution. There are six nonzero variables. Later we will develop a technique to show that the columns of the simplex tableau corresponding to these variables are linearly independent.

Tableau 5.2

5	7	9	6	120
100	0	0	20	
6	7	10	5	140
0	40	0	100	
7	6	8	1	100
0	20	80	0	
100	60	80	120	

Supply

Demand

Have we found the minimum cost solution? Our scheme for distributing the polyethylene represents \$2160 in transportation costs. It is most likely

not the minimum, since no goods made in San Antonio are shipped to San Antonio.

We now look for a way of systematically modifying our initial solution to arrive at the minimum cost distribution scheme in a small number of steps. For each of the *unused* routes, routes along which no goods were shipped, we calculate the change in the total shipping cost due to shipping one unit of goods along that route subject to the restriction that the corresponding changes must be made in the *used* routes. For example, shipping one unit from plant 1 to warehouse 2 ($x_{12} = 1$) leads to the *modifications* of Tableau 5.2 as shown in Tableau 5.3.

Tableau 5.3

5	7 +1	9	6 20 − 1	120
6	7 40 − 1	10	5 100 + 1	140
7	6	8	1	100
100	60	80	120	

Supply

Demand

The change in the total shipping cost due to this one modification is

$$7 - 6 - 7 + 5 = -1.$$

That is, the total shipping cost for the modified tableau (Tableau 5.4) is \$2159. Thus, the cost has decreased and we have improved our solution by \$1.

Tableau 5.4

5 100	7 1	9 0	6 19	120
6 0	7 39	10 0	5 101	140
7 0	6 20	8 80	1 0	100
100	60	80	120	

Supply

Demand

We now compute the possible improvements for each of the unused routes. There are six unused routes shown in Tableau 5.2, namely, (1, 2), (1, 3), (2, 1), (2, 3), (3, 1), and (3, 4). In doing these computations remember that all changes that are forced by the initial change must be assigned to a route that is being used. Thus, in computing the possible improvement for route (3, 4), for example, we have two choices for decreasing by 1 the amount shipped along a route in use. We may choose either (1, 4), or (2, 4). However, only the choice of (2, 4) works, since choosing (1, 4) leads to a change in an unused route that cannot be balanced with a corresponding change in a route that is being used. These two cases are illustrated in Tableaux 5.5 and 5.6.

Tableau 5.5

5	7	9	6	120
6	7 40 + 1	10	5 100 - 1	140
7	6 20 - 1	8	1 +1	100
100	60	80	120	

Supply

This case works

Demand

Tableau 5.6

Cannot balance this increase →

5 100 + 1	7	9	6 20 - 1	120
6	7	10	5	140
7	6	8	1 +1	100
100	60	80	120	

Supply

Demand

We show in Tableaux 5.7a–5.7f (displayed in abbreviated form) the modifications that would have to be made to the amounts shipped as shown in Tableau 5.2 if each of the unused routes were brought into the solution. The corresponding change in cost is noted below each tableau. We see that the shipping cost would be substantially reduced if we

Tableau 5.7

5	7 +1	9	6 -1	120
6	7 -1	10	5 +1	140
7	6	8	1	100
100	60	80	120	

Cost decreases by 1
(a)

5	7	9 +1	6 -1	120
6	7 -1	10	5 +1	140
7	6 +1	8 -1	1	100
100	60	80	120	

Cost decreases by 1
(b)

5 -1	7	9	6 +1	120
6 +1	7	10	5 -1	140
7	6	8	1	100
100	60	80	120	

Cost increases by 2
(c)

Tableau 5.7 (*continued*)

5	7	9	6	120
6	7 (−1)	10 (+1)	5	140
7	6 (+1)	8 (−1)	1	100
100	60	80	120	

Cost increases by 1
(d)

5 (−1)	7	9	6 (+1)	120
6	7 (+1)	10	5 (−1)	140
7 (+1)	6 (−1)	8	1	100
100	60	80	120	

Cost increases by 4
(e)

5	7	9	6	120
6	7 (+1)	10	5 (−1)	140
7	6 (−1)	8	1 (+1)	100
100	60	80	120	

Cost decreases by 3
(f)

allocated more to routes (2, 2) and (3, 4) and then modified routes (3, 2) and (2, 4) accordingly. If we set $x_{34} = 20$ [the largest amount by which we can decrease route (3, 2)] and increase x_{22} by 20, then we get Tableau 5.8, which represents a tableau whose total shipping cost is \$2100. Consequently, Tableau 5.2 did *not* represent an optimal solution.

Tableau 5.8

5	7	9	6	120
100	0	0	20	
6	7	10	5	140
0	60	0	80	
7	6	8	1	100
0	0	80	20	
100	60	80	120	

Supply

Demand

By developing a way to easily compute the possible improvements, we shall see how to reallocate shipments and how to determine when an optimal solution has been reached. We first formulate the dual problem to the transportation problem.

Let v_1, v_2, and v_3 be the dual variables corresponding to the supply constraints, and let w_1, w_2, w_3, and w_4 be the dual variables corresponding to the demand constraints. Then the dual problem to the transportation problem (5) is

$$\text{Maximize} \quad z' = \sum_{i=1}^{3} s_i v_i + \sum_{j=1}^{4} d_j w_j$$

subject to

$$v_i + w_j \leq c_{ij}, \quad i = 1,2,3, \quad j = 1,2,3,4$$

$$v_i, \quad w_j \text{ unrestricted.}$$

Tableau 5.2 represents the initial basic feasible solution to the transportation problem with basic variables x_{11}, x_{14}, x_{22}, x_{24}, x_{32}, and x_{33}. Since the variables x_{ij} of the transportation problem are doubly indexed, so are the imputed values z_{ij}, and from the properties of the simplex algorithm, $z_{ij} = c_{ij}$ when x_{ij} is a basic variable. It follows from the discussion of complementary slackness in Section 3.2 that there are values for the dual variables for which the left-hand side of the dual constraint

$$v_i + w_j \leq c_{ij}$$

is equal to z_{ij}, for all i and j. This means that for the pairs (1, 1), (1, 4), (2, 2), (2, 4), (3, 2), and (3, 3), we have

$$v_i + w_j = c_{ij}. \tag{6}$$

This gives us six equations in seven unknowns. The values of the dual variables can be found from (6) by giving one of the unknowns—say, the one that appears most often—an arbitrary value—say, 0—and then solving for the remaining unknowns. Our system is

$$\begin{aligned}
x_{11}: &\quad v_1 + w_1 = 5\\
x_{14}: &\quad v_1 + w_4 = 6\\
x_{22}: &\quad v_2 + w_2 = 7\\
x_{24}: &\quad v_2 + w_4 = 5\\
x_{32}: &\quad v_3 + w_2 = 6\\
x_{33}: &\quad v_3 + w_3 = 8.
\end{aligned}$$

Setting $v_1 = 0$, we have

$$v_2 = -1, \quad v_3 = -2, \quad w_1 = 5, \quad w_2 = 8, \quad w_3 = 10, \quad \text{and} \quad w_4 = 6.$$

Now the entries in the objective row of the simplex tableau corresponding to the solution in Tableau 5.2 can be determined. The only nonzero entries will be those for the nonbasic variables. Each entry will be of the form

$$z_{ij} - c_{ij} = v_i + w_j - c_{ij}$$

since z_{ij} is the left-hand side of the (i, j) constraint of the dual problem. The entries are

$$\begin{aligned}
x_{12}: &\quad v_1 + w_2 - c_{12} = 0 + 8 - 7 = 1\\
x_{13}: &\quad v_1 + w_3 - c_{13} = 0 + 10 - 9 = 1\\
x_{21}: &\quad v_2 + w_1 - c_{21} = -1 + 5 - 6 = -2\\
x_{23}: &\quad v_2 + w_3 - c_{23} = -1 + 10 - 10 = -1\\
x_{31}: &\quad v_3 + w_1 - c_{31} = -2 + 5 - 7 = -4\\
x_{34}: &\quad v_3 + w_4 - c_{34} = -2 + 6 - 1 = 3.
\end{aligned}$$

Since we are dealing with a minimization problem, the largest positive value determines the entering variable. In this case it is x_{34}. To determine the departing variable, examine route (3, 4). If we try to send one unit along unused route (3, 4) and make the corresponding modification in the routes that are being used (basic variables), we have Tableau 5.9. The total shipping cost is changed by

$$7 - 5 - 6 + 1 = -3 \quad (\textit{Note}: v_3 + w_4 - c_{34} = 3),$$

Tableau 5.9

5	7	9	6	120
6	7	10	5	140
	40 + 1		100 − 1	
7	6	8	1	100
	20 − 1		+1	
100	60	80	120	

Supply

Demand

so that it has been *reduced* by \$3. We can send as many as 20 units along route (3, 4), so that if $x_{34} = 20$, we drive x_{32} to zero. Thus, x_{32} becomes the departing variable. Modifying Tableau 5.2, we obtain Tableau 5.8, as we did before. Since each unit sent along route (3, 4) reduced the total shipping cost by \$3, when 20 units are sent the cost should be reduced by \$60. Indeed, the cost represented by Tableau 5.8 is \$2100.

We now give the calculations necessary to determine the next tableau. The system of equations relating the dual variables is

$$\begin{aligned}
x_{11}: &\quad v_1 + w_1 = 5\\
x_{14}: &\quad v_1 + w_4 = 6\\
x_{22}: &\quad v_2 + w_2 = 7\\
x_{24}: &\quad v_2 + w_4 = 5\\
x_{33}: &\quad v_3 + w_3 = 8\\
x_{34}: &\quad v_3 + w_4 = 1.
\end{aligned}$$

To obtain a solution to this system of six equations in seven unknowns, we arbitrarily set $w_4 = 0$, obtaining

$$v_1 = 6, \quad v_2 = 5, \quad v_3 = 1, \quad w_1 = -1, \quad w_2 = 2, \quad \text{and} \quad w_3 = 7.$$

Consequently, the values of $z_{ij} - c_{ij}$ for the nonbasic variables are

$$\begin{aligned}
x_{12}: &\quad v_1 + w_2 - c_{12} = 6 + 2 - 7 = 1\\
x_{13}: &\quad v_1 + w_3 - c_{13} = 6 + 7 - 9 = 4\\
x_{21}: &\quad v_2 + w_1 - c_{21} = 5 - 1 - 6 = -2\\
x_{23}: &\quad v_2 + w_3 - c_{23} = 5 + 7 - 10 = 2\\
x_{31}: &\quad v_3 + w_1 - c_{31} = 1 - 1 - 7 = -7\\
x_{32}: &\quad v_3 + w_2 - c_{32} = 1 + 2 - 6 = -3.
\end{aligned}$$

The entering variable is x_{13}. If we send one unit along the unused route (1, 3) and make the modifications in the routes being used (basic variables), we have Tableau 5.10. The total shipping cost is changed by

$$9 - 6 - 8 + 1 = -4 \quad (\textit{Note}: v_1 + w_3 - c_{13} = 4),$$

so that it has been reduced by \$4. We can send as many as 20 units along route (1, 3), so that, if $x_{13} = 20$, we drive x_{14} to 0. Thus, x_{14} becomes the departing variable. Modifying Tableau 5.8, we obtain Tableau 5.11. Since each unit sent along route (1, 3) reduced the total shipping cost by \$4, when 20 units are sent the cost should be reduced by \$80. Indeed, the cost represented by Tableau 5.11 is \$2020.

Tableau 5.10

5	7	9	6	120
		+1	20 − 1	
6	7	10	5	140
7	6	8	1	100
		80 − 1	20 + 1	
100	60	80	120	

Supply

Demand

Tableau 5.11

5	7	9	6	120
100	0	20	0	
6	7	10	5	140
0	60	0	80	
7	6	8	1	100
0	0	60	40	
100	60	80	120	

Supply

Demand

Performing these steps several more times, we come to the point at which no entering variable can be chosen (all $z_{ij} - c_{ij}$ are negative because this is a minimization problem), and the procedure stops. An

optimal solution has been obtained; it is given in Tableau 5.12. The cost of this solution is \$1900. Note that it chooses the plant in San Antonio to supply all the demand of the San Antonio warehouse.

Tableau 5.12

5	7	9	6	120
100	0	20	0	
6	7	10	5	140
0	60	60	20	
7	6	8	1	100
0	0	0	100	
100	60	80	120	

Supply

Demand

△

We formalize the procedure used in this example as "the transportation algorithm." Given the transportation problem

$$\text{Minimize} \quad z = \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} x_{ij}$$

subject to

$$\sum_{j=1}^{n} x_{ij} = s_i, \quad i = 1, 2, \ldots, m$$

$$\sum_{i=1}^{m} x_{ij} = d_j, \quad j = 1, 2, \ldots, n$$

$$x_{ij} \geq 0, \quad \text{integers}$$

$$\sum_{i=1}^{m} s_i = \sum_{j=1}^{n} d_j,$$

we choose an initial basic feasible solution using the **minimum cost rule**: For each row $i = 1, 2, \ldots, m$ assign the maximum possible amount x_{ij} of the remaining supply to the cheapest route. The transportation algorithm then consists of the following steps.

1. Find the entering variable.

(a) Solve the dual constraint equations corresponding to the basic variables for the remaining $m + n - 1$ dual variables. The value of one dual variable will have to be chosen arbitrarily.

(b) Evaluate the objective row coefficients by computing $z_{ij} - c_{ij} = v_i + w_j - c_{ij}$ for each pair (i, j), where x_{ij} is a nonbasic variable.

(c) The entering variable is x_{ij}, where $z_{ij} - c_{ij}$ is the largest positive value from Step (b). If all $z_{ij} - c_{ij}$ are nonpositive, stop; an optimal solution has been obtained.

2. Find the departing variable (these steps will be modified to cover certain degenerate cases).

(a) Determine which basic variables x_{pq} will decrease when x_{ij} is increased.

(b) The departing variable is the one from the list in Step (a) whose value is smallest.

3. Form the new tableau.

(a) Set the departing variable to 0.

(b) Set the entering variable equal to the previous value of the departing variable.

(c) Adjust the values of the other basic variables to make the supply and demand constraints hold.

To show that this algorithm works for all possible transportation problems, we must check the following points.

1. The minimum cost rule yields a basic feasible solution.
2. The dual constraint equations can always be solved.
3. The values of the objective row entries $z_{ij} - c_{ij}$ are independent of the choice of value for one of the dual variables.
4. The departing variable can always be computed by the given scheme. A more precise statement of this scheme will also be needed.

We need to introduce some definitions so that we can further discuss the points we raised above. We will call each block in the transportation tableau a **cell**. The value of x_{ij} is recorded in cell (i, j). In our example we saw that if we changed the entry in cell (i, j) from 0 to 1, where x_{ij} was a nonbasic variable, then this change forced changes in the values of some of the basic variables. We now systematize the recording of these changes.

A **loop** in a transportation tableau is a sequence of cells in the tableau that satisfies the following criteria.

1. The sequence consists of alternating horizontal and vertical segments.
2. Each segment joins exactly two cells.
3. The first cell of the sequence is the last, and no other cell is used twice.

Properties 1 and 2 tell us that if (i, j) is a cell in a loop and if we reached it horizontally (along row i), then the next cell in the loop must be in column j. Likewise, if we reached cell (i, j) vertically, then the next cell in the loop must be in row i. Consequently, we use the cells in each row two at a time

when forming a loop. A loop must therefore have an even number of cells in it. We sketch some of the loops we found when computing possible improvements for our example in Tableau 5.13. The heavy dot in the cell indicates that the cell is included in the loop. This loop could be written as

$$(1,2),\quad (1,4),\quad (2,4),\quad (2,2)$$

if we proceeded horizontally from (1,2). The example in Tableau 5.14 shows how a loop can be more complicated. In this example the loop is

$$(3,1),\quad (1,1),\quad (1,4),\quad (2,4),\quad (2,2),\quad (3,2).$$

Another example is given in Tableau 5.15. This loop is

$$(1,3),\quad (3,3),\quad (3,2),\quad (2,2),\quad (2,4),\quad (1,4).$$

Tableau 5.13

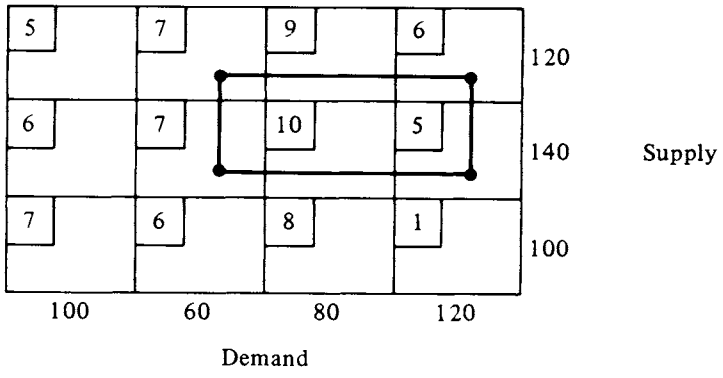


Tableau 5.14

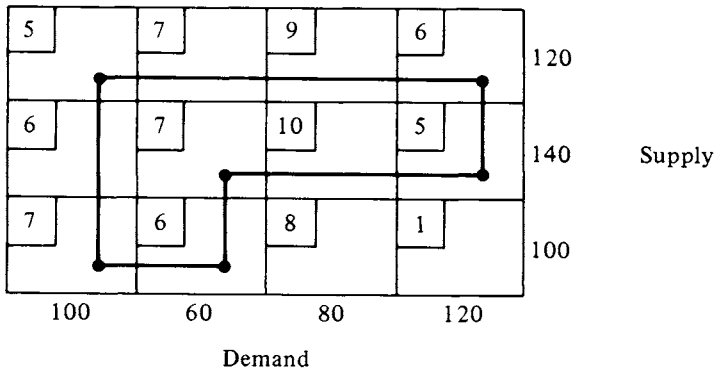
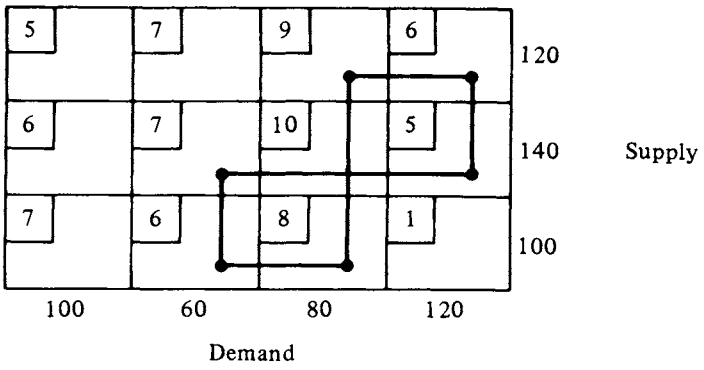


Tableau 5.15



Note that cell (2,3) is not in the loop, even though two segments cross there.

Recall that a basic feasible solution to a linear programming problem with m constraints, none of which was redundant, had at most m nonzero values, and the corresponding columns of the coefficient matrix **A** were linearly independent. Loops give us a very convenient way of determining the linear independence of the corresponding columns of **A** for a transportation problem. We saw in (3) that each column of the coefficient matrix **A** corresponded to one of the variables x_{ij} and hence to one of the cells in the transportation tableau.

THEOREM 5.1. *The columns of* **A** *determined by*

$$(i_1, j_1)(i_2, j_2), \ldots, (i_k, j_k)$$

are linearly dependent if and only if the corresponding cells (or some of them) can be arranged in a loop.

Proof. Omitted. △

This theorem shows that in computing possible improvements we were discovering how to write each column of **A** that corresponds to a nonbasic variable as a linear combination of the columns that correspond to the basic variables. It also allows us to conclude that the minimum cost rule gives a basic feasible solution—one whose corresponding columns of **A** are linearly independent.

Theorem 5.1 also allows us to formalize the procedure for computing the departing variable. If x_{ij} has been chosen as the entering variable, then cell (i, j) must belong to a loop consisting of basic cells. That is, the column of **A** corresponding to x_{ij} must be a linear combination of the columns of **A** corresponding to the basic variables. We list the cells of the loop for x_{ij} in order, starting with (i, j). The cells that appear in the

even-numbered positions in the sequence will have their values decreased when the value of x_{ij} is increased. To maintain feasibility, these values cannot be decreased to a negative number. Therefore, we may choose the departing variable x_{kl} as one that satisfies the following criteria.

(a) The variable x_{kl} is a basic variable and cell (k, l) appears in the loop for cell (i, j).

(b) Cell (k, l) appears in an even-numbered position in the loop.

(c) The value of x_{kl} is the smallest of the values of all variables in even-numbered positions in the loop.

For example, going from Tableau 5.2 to 5.8, we found that x_{34} was the entering variable and that it belonged to the loop

$$(3,4), \quad (2,4), \quad (2,2), \quad (3,2)$$

We have $x_{24} = 100$ and $x_{32} = 20$, so that x_{32} has the smaller value. Thus, x_{32} is the departing variable.

The special form of the transportation problem allows us to do the pivotal elimination in a very concise form. Suppose the value of the departing variable x_{kl} is α. Then each variable in an odd-numbered position in the loop of the entering variable has, as its new value, α plus its old value. Each variable in an even-numbered position in the loop has, as its new value, its old value minus α. In this scheme the entering variable automatically is set at α, and the departing variable is set at 0.

The equations that are the dual constraints can always be solved. There are more unknowns than equations, and the equations are consistent. In the following example we show a fast technique for solving these equations by hand.

EXAMPLE 2. Consider a transportation problem between three sources and five destinations in which the supply vector is

$$\mathbf{s} = \begin{bmatrix} 100 \\ 160 \\ 140 \end{bmatrix}$$

and the demand vector is

$$\mathbf{d} = \begin{bmatrix} 90 \\ 60 \\ 80 \\ 100 \\ 70 \end{bmatrix}.$$

The costs are shown in Tableau 5.16. We find an initial basic feasible solution by using the minimum-cost rule. It is shown in Tableau 5.17 and has a value of $z = 1990$. Note that this solution was found in the order

shown in Table 5.1 and that each assignment satisfied either a demand constraint or a supply constraint but not both (with the exception of the last assignment). Also note that the solution has

$$3 + 5 - 1 = 7$$

nonzero variables.

Tableau 5.16

9	3	6	7	3	100
7	5	2	10	6	160
5	4	9	8	10	140
90	60	80	100	70	

Supply

Demand

Tableau 5.17

9 0	3 60	6 0	7 0	3 40	100
7 50	5 0	2 80	10 0	6 30	160
5 40	4 0	9 0	8 100	10 0	140
90	60	80	100	70	

Supply

Demand

Table 5.1

Assignment	*Cell*	*Constraint satisfied*
1	(1, 2)	Demand
2	(1, 5)	Supply
3	(2, 3)	Demand
4	(2, 5)	Demand
5	(2, 1)	Supply
6	(3, 1)	Demand
7	(3, 4)	Demand and supply

To find the entering variable, we must compute the possible improvement or $z_{ij} - c_{ij}$ for each nonbasic variable. Recall that

$$z_{ij} - c_{ij} = v_i + w_j - c_{ij}.$$

For the basic variables we solve the system of equations

$$\begin{aligned}
&\text{(a)} \quad v_1 + w_2 = c_{12} = 3 \\
&\text{(b)} \quad v_1 + w_5 = c_{15} = 3 \\
&\text{(c)} \quad v_2 + w_1 = c_{21} = 7 \\
&\text{(d)} \quad v_2 + w_3 = c_{23} = 2 \\
&\text{(e)} \quad v_2 + w_5 = c_{25} = 6 \\
&\text{(f)} \quad v_3 + w_1 = c_{31} = 5 \\
&\text{(g)} \quad v_3 + w_4 = c_{34} = 8.
\end{aligned}$$

We adjoin a row below the tableau for the values of w_s and a column to the right for the values of v_r and leave out the zeros in the nonbasic cells. We start by assuming $v_1 = 0$. We obtain in the following order

$$\begin{aligned}
w_2 &= 3 && \text{from (a)} \\
w_5 &= 3 && \text{from (b)} \\
v_2 &= 3 && \text{from (e)} \\
w_3 &= -1 && \text{from (d)} \\
w_1 &= 4 && \text{from (c)} \\
v_3 &= 1 && \text{from (f)} \\
w_4 &= 7 && \text{from (g).}
\end{aligned}$$

These results are shown in Tableau 5.18. Now the values for $z_{ij} - c_{ij}$ can be filled in for the blanks in Tableau 5.18. The values of the basic variables are circled to distinguish them. We obtain Tableau 5.19. Only $z_{22} - c_{22}$ is

positive. Hence, x_{22} is the entering variable. It belongs to the loop of basic variables.

$$(2,2),\quad (2,5),\quad (1,5),\quad (1,2).$$

Tableau 5.18

						v
	9	3 (60)	6	7	3 (40)	100	0
	7 (50)	5	2 (80)	10	6 (30)	160	3
	5 (40)	4	9	8 (100)	10	140	1
	90	60	80	100	70		
w	4	3	-1	7	3		

Supply

Demand

Tableau 5.19

						v
	9 -5	3 (60)	6 -7	7 0	3 (40)	100	0
	7 (50)	5 1	2 (80)	10 0	6 (30)	160	3
	5 (40)	4 0	9 -9	8 (100)	10 -6	140	1
	90	60	80	100	70		
w	4	3	-1	7	3		

Supply

Demand

We have $x_{25} = 30$ and $x_{12} = 60$, so that the smaller x_{25} becomes the departing variable. We increase x_{22} and x_{15} by 30 and decrease x_{25} and

x_{12} by 30. This gives Tableau 5.20 with objective function value $z = 1960$. Solving for v_r and w_s and then computing $z_{ij} - c_{ij}$, we obtain Tableau 5.21.

Tableau 5.20

9	3 (30)	6	7	3 (70)	100	
7 (50)	5 (30)	2 (80)	10	6	160	Supply
5 (40)	4	9	8 (100)	10	140	
90	60	80	100	70		

Demand

Tableau 5.21

							v
	9 −4	3 (30)	6 −6	7 1	3 (70)	100	0
	7 (50)	5 (30)	2 (80)	10 0	6 −1	160	2
	5 (40)	4 −1	9 −9	8 (100)	10 −7	140	0
	90	60	80	100	70		
w	5	3	0	8	3		

Demand

Supply

In Tableau 5.21 the entering variable is x_{14} and it belongs to the loop

(1,4), (3,4), (3,1), (2,1), (2,2), (1,2).

We have

$$x_{34} = 100, \quad x_{21} = 50, \quad \text{and} \quad x_{12} = 30,$$

so that x_{12} is the departing variable. Decreasing x_{34}, x_{21}, and x_{12} by 30 and increasing x_{14}, x_{31}, and x_{22} by the same amount gives us Tableau 5.22. We have also solved for the dual variables and given the values of $z_{ij} - c_{ij}$. Since all these values are nonpositive, we have found an optimal solution. Its value is $z = 1930$.

Tableau 5.22

						v
9	3	6	7	3	100	0
-5	-1	-7	(30)	(70)		
7	5	2	10	6	160	3
(20)	(60)	(80)	0	0		
5	4	9	8	10	140	1
(70)	-1	-9	(70)	-6		
90	60	80	100	70		
w 4	2	-1	7	3		

Supply

Demand

△

Degeneracy

Degeneracy in a transportation problem has the same meaning as it did for a general linear programming problem. That is, a basic variable has value zero. This means that we have designated a route as being used although no goods are being sent along it. In a transportation problem degeneracy can occur in two ways. It is possible that while finding an initial basic feasible solution both a supply and a demand constraint are satisfied simultaneously. On the other hand, there may be a tie for choosing a departing variable. In either case at least one basic variable will have value zero. As in the simplex method, degeneracy generally causes no difficulties. A transportation problem has never been known to cycle.

An algorithm to find an initial basic feasible solution to the transportation problem assigns a value to a variable to satisfy a demand constraint or

a supply constraint. This variable is then an initial basic variable that corresponds to the constraints satisfied. If both a demand and a supply constraint are simultaneously satisfied by the assignment of a value to x_{rs}, *except for the final allocation*, we have degeneracy. To maintain the correspondence between basic variables and satisfied constraints, we must designate a variable other than x_{rs} as also being basic and assign to it the value zero.

EXAMPLE 3. Consider the transportation problem defined by Tableau 5.23. Using the minimum cost rule, the cells are filled in the order

$$(1,2),\quad (1,5),\quad (2,3),\quad (2,5).$$

Tableau 5.23

9	3	6	7	3	100
7	5	2	10	6	160
5	4	9	8	10	100
50	60	80	100	70	

Supply

Demand

At this point the tableau looks like Tableau 5.24. The next assignment of 50 units to cell (2, 1) will complete both the first column and the second row.

Thus, we have degeneracy. To systematically choose a variable as basic and having value zero, we agree in this case to say that only the second row has been completed. Moving to the third row, the smallest available cost is for cell (3, 1), and x_{31} is assigned the value zero. This makes x_{31} a basic variable. We complete the determination of an initial basic feasible solution by letting $x_{34} = 100$. Degeneracy during the iterations of the transportation algorithm may be ignored. △

Tableau 5.24

9	3	6	7	3	
0	60	0	0	40	100
7	5	2	10	6	
	0	80		30	160
5	4	9	10	10	
	0	0		0	100
50	60	80	100	70	

Supply

Demand

Starting Procedures

We have already described the minimum cost rule for obtaining an initial basic feasible solution to the transportation problem. A number of other methods are available. A desirable starting method is one that will provide a starting solution that is not far from the optimal one in the sense that only a small number of iterations of the transportation algorithm are required. We now describe Vogel's method, which is widely used.

Vogel's Method

Let $\mathbf{C} = [c_{ij}]$ be the cost matrix of a transportation problem.

1. For each row and each column of $\mathbf{C}$ find the difference between the smallest and the next smallest entry. This difference represents the minimum penalty incurred when one fails to assign goods to the cheapest route.

2. Select the row or column with the largest difference. Ties may be broken arbitrarily.

3. Allocate as much as possible to the cell with the smallest cost in that row or column. Let us say that this allocation is made to cell (r, s). Decrease the available supply in row r and the required demand in column s by the amount allocated. This allocation will satisfy a demand constraint, a supply constraint, or perhaps both. The indication of which constraint has been satisfied is the reduction of the available supply in row r or the required demand in column s to zero. Remove the constraint that is satisfied from further consideration by crossing out the corresponding row or column of the cost matrix. If both a demand and a supply constraint are satisfied simultaneously, remove only one from further consideration. In

this case both the available supply and the required demand have been reduced to zero.

4. Repeat Steps 1, 2, and 3 until *either* exactly one row or exactly one column remains. In doing Step 1, do not compute differences for any row with 0 available supply or any column with 0 required demand. When exactly one row or one column remains, the entries in that row or column are fully determined by the previous allocations and are filled in accordingly.

EXAMPLE 4. We apply Vogel's method to the transportation problem with the cost matrix and supply and demand vectors shown below:

$$\mathbf{C} = \begin{bmatrix} 8 & 6 & 3 & 9 \\ 2 & 6 & 1 & 4 \\ 7 & 8 & 6 & 3 \end{bmatrix}, \quad \mathbf{s} = \begin{bmatrix} 120 \\ 140 \\ 100 \end{bmatrix}, \quad \text{and} \quad \mathbf{d} = \begin{bmatrix} 100 \\ 60 \\ 80 \\ 120 \end{bmatrix}.$$

We compute the differences according to Step 1 of the algorithm and circle the largest.

					Differences
	8	6	3	9	3
	2	6	1	4	1
	7	8	6	3	3
Differences:	⑤	0	2	1	

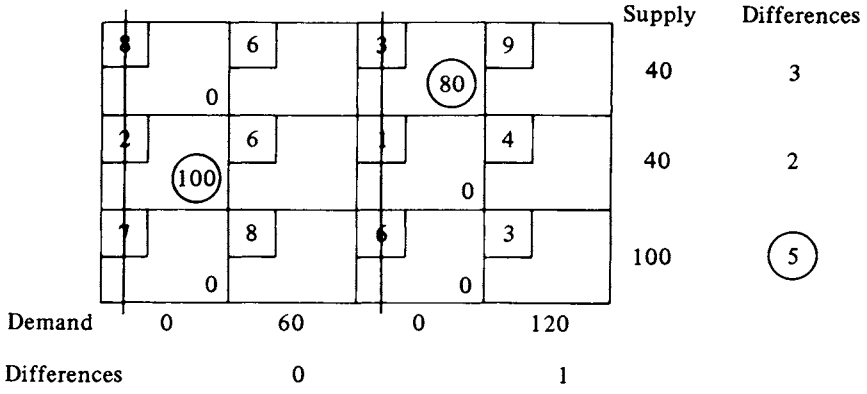
Thus, we allocate as much as possible (100 units) to the cheapest route in the first column ($x_{21} = 100$), fill in the rest of this column with zeros, and cross out the column. The allocation in x_{21} is circled to indicate that x_{21} is a basic variable. The revised supplies and demands from Step 3 of the algorithm and the new differences are shown in Tableau 5.25.

Tableau 5.25

				Supply	Differences
8	6	3	9	120	③
0					
2	6	1	4	40	3
⑩⓪					
7	8	6	3	100	3
0					
Demand 0	60	80	120		
Differences	0	2	1		

We can arbitrarily choose among rows 1, 2, and 3. Choosing the first row, we allocate 80 units to the cheapest route in that row ($x_{13} = 80$), circle the allocation, fill in the rest of the row with zeros, and cross out the third column. We revise the supplies and demands, compute the new differences, and show the result in Tableau 5.26.

Tableau 5.26



Choosing row 3, we allocate 100 units to the cheapest route in that row ($x_{34} = 100$) and cross out the row. Tableau 5.27 shows the consequences of this allocation along with the new differences. The largest difference is now in column 4, so that we allocate 20 units to the cheapest route in that column ($x_{24} = 20$), cross out the column, and obtain Tableau 5.28.

Tableau 5.27

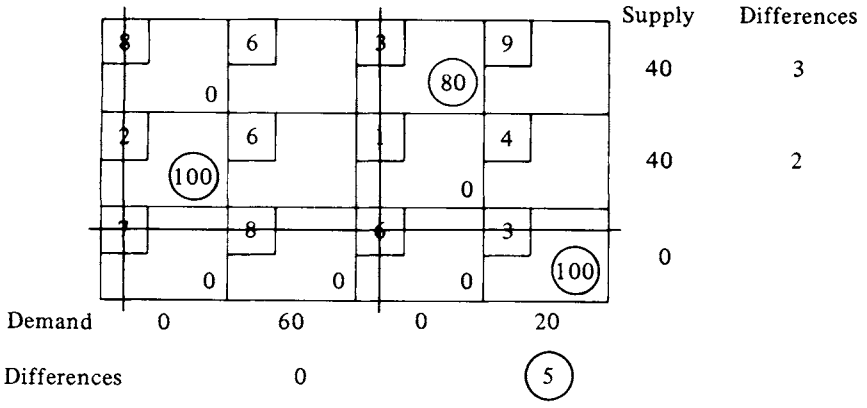
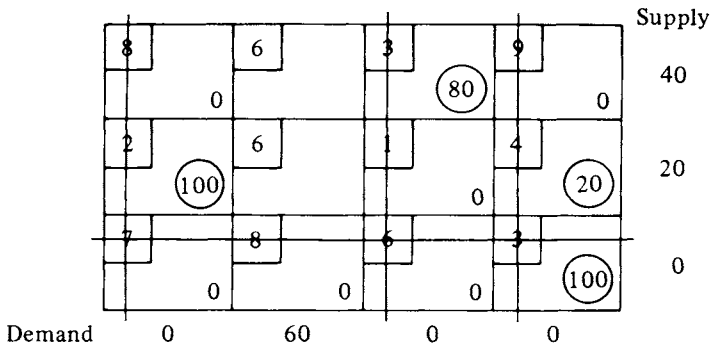
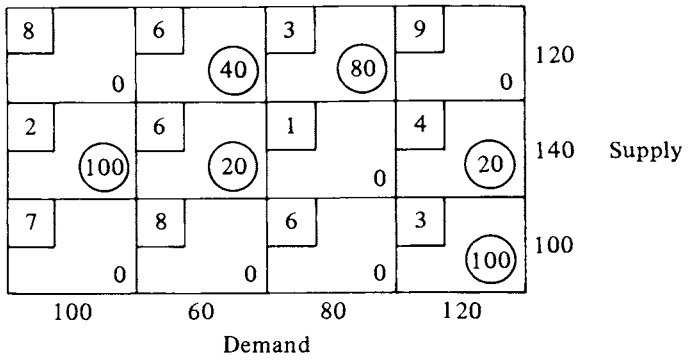


Tableau 5.28



There is now exactly one column remaining. The allocations for that column are completely determined ($x_{21} = 40$ and $x_{22} = 20$). Thus, the initial basic feasible solution is Tableau 5.29, with a cost of \$1180.

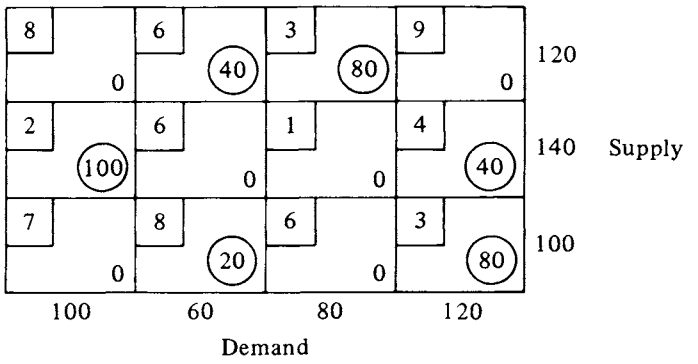
Tableau 5.29



The minimum cost method yields the initial basic feasible solution in Tableau 5.30, whose cost is \$1240. Thus, in this case the Vogel method gives a better starting solution. Applying the transportation algorithm to either of the two starting solutions given above, the reader may show that the minimum cost for this transportation problem is \$1140. △

The effectiveness of Vogel's method can be improved by using a procedure due to Roland E. Larson that is, perhaps, too complicated for hand computation but can be carried out rapidly on a computer. Instead of

Tableau 5.30



using the given costs c_{ij} for Vogel's method, we use the normalized costs c'_{ij} defined by

$$c'_{ij} = c_{ij} - \frac{1}{n}\sum_{p=1}^{n} c_{pj} - \frac{1}{m}\sum_{q=1}^{m} c_{iq}.$$

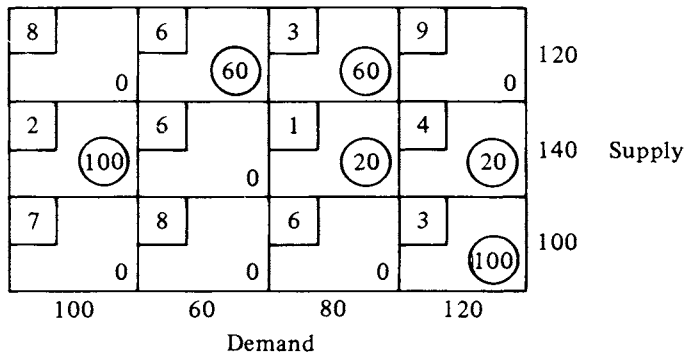
That is, we subtract from each c_{ij} the average of the costs of the row and column in which it appears. We then apply Vogel's method to the matrix $[c'_{ij}]$.

EXAMPLE 5. We consider the same transportation problem as that in Example 4. When we compute the normalized cost matrix, we obtain

$$\mathbf{C}' = \begin{bmatrix} -\frac{25}{6} & -\frac{43}{6} & -\frac{41}{6} & -\frac{17}{6} \\ -\frac{73}{12} & -\frac{37}{12} & -\frac{67}{12} & -\frac{35}{12} \\ -\frac{14}{3} & -\frac{14}{3} & -\frac{10}{3} & -\frac{25}{3} \end{bmatrix}.$$

Applying Vogel's method to $\mathbf{C}'$, we obtain the starting solution in Tableau 5.31. This solution is actually an optimal solution, and its cost is \$1140.

Tableau 5.31



△

Extensions

If the total supply exceeds the total demand in the original statement of the transportation problem, a dummy destination can be set up. That is, we create a destination with a demand equal to the difference between the total supply and the total demand. The cost of shipping to this destination from any source is 0. In fact, we have just added a slack variable to each supply constraint.

EXAMPLE 6. The problem originally given in Example 3, Section 2.1, is

Minimize $z = 5x_{11} + 7x_{12} + 9x_{13} + 6x_{21} + 7x_{22} + 10x_{23}$

subject to

$$\begin{aligned} x_{11} + x_{12} + x_{13} &\leq 120 \\ x_{21} + x_{22} + x_{23} &\leq 140 \\ x_{11} + x_{21} &\geq 100 \\ x_{12} + x_{22} &\geq 60 \\ x_{13} + x_{23} &\geq 80. \end{aligned}$$

The difference between supply and demand is 20. We create a destination with a demand equal to this amount. The constraints then become equalities, since demand equals supply. The tableau for the problem is Tableau 5.32.

Tableau 5.32

5	7	9	0	120
6	7	10	0	140
100	60	80	20	

Supply

Demand

△

5.1 EXERCISES

1. Verify that Tableau 5.12 represents the optimal solution to Example 1 by performing the necessary steps to obtain Tableau 5.12 from Tableau 5.11.

In Exercises 2–4 find an initial basic feasible solution using (a) the minimum cost rule, (b) Vogel's method, and (c) Larson's method if a hand calculator is available.

2. $\mathbf{C} = \begin{bmatrix} 5 & 2 & 3 & 6 \\ 2 & 7 & 7 & 4 \\ 1 & 3 & 6 & 9 \end{bmatrix}$, $\mathbf{s} = \begin{bmatrix} 100 \\ 80 \\ 140 \end{bmatrix}$, and $\mathbf{d} = \begin{bmatrix} 60 \\ 60 \\ 80 \\ 120 \end{bmatrix}$

3. $\mathbf{C} = \begin{bmatrix} 6 & 6 & 3 & 9 & 2 \\ 8 & 7 & 5 & 7 & 5 \\ 3 & 4 & 8 & 2 & 7 \\ 6 & 0 & 0 & 2 & 9 \end{bmatrix}$, $\mathbf{s} = \begin{bmatrix} 90 \\ 70 \\ 110 \\ 150 \end{bmatrix}$, and $\mathbf{d} = \begin{bmatrix} 100 \\ 40 \\ 100 \\ 60 \\ 120 \end{bmatrix}$

4. $\mathbf{C} = \begin{bmatrix} 4 & 6 & 7 & 5 \\ 4 & 4 & 7 & 8 \\ 5 & 3 & 6 & 5 \\ 6 & 5 & 3 & 4 \end{bmatrix}$, $\mathbf{s} = \begin{bmatrix} 100 \\ 60 \\ 50 \\ 70 \end{bmatrix}$, and $\mathbf{d} = \begin{bmatrix} 40 \\ 70 \\ 120 \\ 50 \end{bmatrix}$

5. Solve the transportation problem given in Example 3.

6. Solve the transportation problem given in Example 6.

7. Solve the transportation problem given in Exercise 3.

In Exercises 8–13 solve the given transportation problem.

8. $\mathbf{C} = \begin{bmatrix} 2 & 5 & 6 & 3 \\ 9 & 6 & 2 & 1 \\ 7 & 7 & 2 & 4 \end{bmatrix}$, $\mathbf{s} = \begin{bmatrix} 100 \\ 90 \\ 130 \end{bmatrix}$, and $\mathbf{d} = \begin{bmatrix} 70 \\ 50 \\ 30 \\ 120 \end{bmatrix}$

9. $\mathbf{C} = \begin{bmatrix} 3 & 2 & 5 & 4 \\ 6 & 5 & 7 & 8 \\ 2 & 1 & 4 & 3 \\ 4 & 3 & 5 & 2 \end{bmatrix}$, $\mathbf{s} = \begin{bmatrix} 80 \\ 60 \\ 50 \\ 100 \end{bmatrix}$, and $\mathbf{d} = \begin{bmatrix} 70 \\ 70 \\ 70 \\ 80 \end{bmatrix}$

10. $\mathbf{C} = \begin{bmatrix} 4 & 3 & 2 & 5 & 6 \\ 8 & 3 & 4 & 5 & 7 \\ 6 & 8 & 6 & 7 & 5 \\ 4 & 3 & 5 & 2 & 4 \end{bmatrix}$, $\mathbf{s} = \begin{bmatrix} 70 \\ 80 \\ 60 \\ 120 \end{bmatrix}$, and $\mathbf{d} = \begin{bmatrix} 60 \\ 50 \\ 50 \\ 70 \\ 100 \end{bmatrix}$

11. $\mathbf{C} = \begin{bmatrix} 4 & 2 & 9 & 7 \\ 7 & 8 & 5 & 6 \\ 3 & 3 & 4 & 1 \\ 7 & 5 & 2 & 6 \end{bmatrix}$, $\mathbf{s} = \mathbf{d} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$

12. $\mathbf{C} = \begin{bmatrix} 5 & 6 & 7 & 4 \\ 2 & 9 & 7 & 5 \\ 8 & 5 & 8 & 7 \end{bmatrix}$, $\mathbf{s} = \begin{bmatrix} 75 \\ 50 \\ 60 \end{bmatrix}$, and $\mathbf{d} = \begin{bmatrix} 45 \\ 50 \\ 25 \\ 50 \end{bmatrix}$

13. $\mathbf{C} = \begin{bmatrix} 6 & 4 & 3 & 5 \\ 7 & 4 & 8 & 6 \\ 8 & 3 & 2 & 5 \end{bmatrix}$, $\mathbf{s} = \begin{bmatrix} 100 \\ 60 \\ 50 \end{bmatrix}$, and $\mathbf{d} = \begin{bmatrix} 60 \\ 80 \\ 70 \\ 40 \end{bmatrix}$

14. Show that, in the $m \times n$ transportation problem, if

$$\sum_{i=1}^{m} s_i = \sum_{j=1}^{n} d_j,$$

then the constraints are equalities.

15. Show that, in the $m \times n$ transportation problem, if

$$\sum_{i=1}^{m} s_i = \sum_{j=1}^{n} d_j,$$

then one constraint is redundant. (*Hint:* First, sum all the supply constraints. Second, sum all but one of the demand constraints. Then show that the difference between the two sums is the other demand constraint.)

16. Write the formulation of the dual of the $m \times n$ transportation problem.

In Exercises 17 and 18 consider the general transportation problem in which

$$S = \sum_{i=1}^{m} s_i = \sum_{j=1}^{n} d_j.$$

17. Show that $x_{ij} = s_i d_j / S$ is a feasible solution.

18. Show that if $\mathbf{X} = [x_{ij}]$ is a feasible solution, then, for all i and j,

$$0 \leq x_{ij} \leq \min\{s_i, d_j\}.$$

Further Reading

Berge, C., and Ghouila-Houri, A. *Programming, Games and Transportation Networks*. Wiley, New York, 1965.

Larson, R. E. "Normalizing Vogel's Approximation Method." *Math. Magazine*, **45** (1972), 266–269.

Reinfeld, N. V., and Vogel, W. R. *Mathematical Programming*. Prentice-Hall, Englewood Cliffs, NJ, 1958.

5.2 THE ASSIGNMENT PROBLEM

We gave an example of the assignment problem in Section 4.1, Example 3. In this section we will discuss the simplest formulation of the problem. We assume that there are exactly as many persons available for assignment as there are jobs and that each person is to be assigned exactly one job. In the event that there are more persons available than jobs, we can create dummy jobs for the excess persons. Being assigned to a dummy job means in reality that the person is not assigned. We do not consider the case in which there are more jobs than people if all the jobs are to be completed because this violates our basic assumption that each person is assigned exactly one job.

The assignment problem, like the transportation problem, will also be considered as a minimization problem. That is, if we assign person i to job j, this assignment will cost c_{ij}. Our goal is to minimize the total cost of the assignment. Our first model for the problem is

$$\text{Minimize} \quad z = \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij}$$

subject to

$$\sum_{i=1}^{n} x_{ij} = 1, \quad j = 1,2,\ldots,n \tag{1}$$

$$\sum_{j=1}^{n} x_{ij} = 1, \quad i = 1,2,\ldots,n \tag{2}$$

$$x_{ij} = 0 \text{ or } 1, \quad i = 1,2,\ldots,n; \qquad j = 1,2,\ldots,n \tag{3}$$

We can actually remove the restriction that x_{ij} take on only 0 or 1 as a value. If we simply assume that x_{ij} must be an integer, then each of constraints (1) and (2) guarantees that x_{ij} can take on only 0 or 1 as a value. Thus, in place of (3) we write

$$x_{ij} \geq 0, \quad \text{integers}, \quad i = 1,2,\ldots,n; \qquad j = 1,2,\ldots,n. \tag{4}$$

The model defined by (1), (2), and (4) is actually a transportation problem in which the demands are all equal, the supplies are all equal, the supplies equal the demands, and the number of sources is equal to the number of destinations. Consequently, this problem could be solved using the transportation algorithm of Section 5.1. With this algorithm we will find that $2n - 1$ variables are basic. However, only n of these variables will be nonzero, so that the solution is highly degenerate. We would suspect that many iterations of the transportation algorithm would simply replace a basic variable with a zero value with another variable with a zero value. To obtain a better algorithm for the problem it will be helpful to use a slightly different point of view.

Suppose a list of available persons and a list of the jobs are both written in some fixed order. Then one possible assignment is to give job i to person i. On the other hand, if the job list is reordered, it might be possible to reduce the total cost. Assume that person i is still assigned the job in the ith position of the new job list, but since the list was shuffled, this job is no longer job i. What we need is a shuffling or permutation of the job list that yields the minimum total cost.

One way of recording a permutation of the numbers $1,2,\ldots,n$ is to write a list of these numbers in the desired order. For example, 4 2 1 3 is a permutation of 1 2 3 4. Another way of writing this permutation is to give a 4×4 matrix $\mathbf{M} = [m_{ij}]$ with entries that are zeros

and ones. If $m_{ij} = 1$, it means that i is assigned the jth position in the new order. We see that 4 2 1 3 corresponds to the matrix

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

since 1 is in third position, 2 is in second position, 3 is in fourth position, and 4 is in first position.

The set of all feasible solutions to the assignment problem is simply the set of all $n \times n$ permutation matrices. Note that each permutation matrix has the property that in each row (or column) there is precisely one entry equal to 1.

THEOREM 5.2. *If the cost matrix for an assignment problem has nonnegative entries and at least n zeros, then an optimal solution to the problem exists if n of the zeros lie in the positions of the ones of some $n \times n$ permutation matrix* **P**. *The matrix* **P** *represents an optimal assignment.*

Proof. In the situation described, the cost can never be smaller than zero, and we have found an assignment for which the cost is zero. △

This theorem provides a goal for our algorithm. We will show that we can modify the cost matrix without changing the optimal solution. The algorithm will then attempt to carry out this modification to reach a situation in which the cost matrix has a zero in each row and in each column.

THEOREM 5.3. *Suppose the matrix* $\mathbf{C} = [c_{ij}]$ *is the cost matrix for an* $n \times n$ *assignment problem. Suppose that* $\hat{\mathbf{X}} = [\hat{x}_{ij}]$ *is an optimal solution to this problem. Let* $\mathbf{C}'$ *be the matrix formed by adding* α *to each entry in the rth row. Then* $\hat{\mathbf{X}}$ *is an optimal solution to the new assignment problem defined by* $\mathbf{C}'$.

Proof. The objective function for the new problem is

$$z' = \sum_{i=1}^{n} \sum_{j=1}^{n} c'_{ij} x_{ij} = \sum_{\substack{i=1 \\ i \neq r}}^{n} \sum_{j=1}^{n} c_{ij} x_{ij} + \sum_{j=1}^{n} (c_{rj} + \alpha) x_{rj}$$

$$= \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij} + \alpha \sum_{j=1}^{n} x_{rj}$$

$$= \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij} + \alpha$$

since each row sum is 1. Therefore, the smallest value for z' will be obtained when

$$z = \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij}$$

is smallest; namely, it is obtained when $\mathbf{X} = \hat{\mathbf{X}}$. △

A statement similar to Theorem 5.3 can be made if a constant is added to some column of a cost matrix. Thus, our strategy is to modify **C** by adding constants to rows or columns.

EXAMPLE 1. Suppose the cost matrix for an assignment problem is

$$\mathbf{C} = \begin{bmatrix} 4 & 5 & 2 & 5 \\ 3 & 1 & 1 & 4 \\ 12 & 3 & 6 & 3 \\ 12 & 6 & 5 & 9 \end{bmatrix}.$$

We can begin to introduce zeros by subtracting the smallest entry in each row from that row. We obtain

$$\begin{bmatrix} 2 & 3 & 0 & 3 \\ 2 & 0 & 0 & 3 \\ 9 & 0 & 3 & 0 \\ 7 & 1 & 0 & 4 \end{bmatrix}.$$

There is no zero in column 1, but one can be forced there by subtracting 2, the smallest entry in column 1, from each entry in that column. We have

$$\begin{bmatrix} 0 & 3 & 0 & 3 \\ 0 & 0 & 0 & 3 \\ 7 & 0 & 3 & 0 \\ 5 & 1 & 0 & 4 \end{bmatrix}.$$

We now have at least one zero in each row and column. We attempt to assign the locations for the ones in a permutation matrix.

Actually, we will not produce the permutation matrix but will star the zeros in the cost matrix to indicate an assignment. We must assign the zero in position (4, 3) since it is the only zero in row 4. However, we will carry out the assignment in an order that will anticipate part of the algorithm we are developing. Starting with the first row, we assign the first zero in each row that does not belong to a previously assigned column. We say a row or column is **assigned** when some zero in it is assigned. Thus, in row 1 we assign the zero in column 1; in row 2 we assign the zero in column 2, skipping the zero in column 1 because that column was previously assigned; in row 3 we assign the zero in column 4; and in row 4 we assign the

zero in column 3. The assignment that we obtain is

$$\begin{bmatrix} 0^* & 3 & 0 & 3 \\ 0 & 0^* & 0 & 3 \\ 7 & 0 & 3 & 0^* \\ 5 & 1 & 0^* & 4 \end{bmatrix}$$

or person 1 is assigned job 1, person 2 is assigned job 2, person 3 is assigned job 4, and person 4 is assigned job 3. The cost is

$$4 + 1 + 3 + 5 = 13. \qquad \Delta$$

In Example 1 we were successful in determining an assignment for each person. However, as the next example shows, we may not always be as fortunate.

EXAMPLE 2. Suppose the cost matrix for an assignment problem is

$$\mathbf{C} = \begin{bmatrix} 4 & 1 & 3 & 4 \\ 5 & 6 & 2 & 9 \\ 6 & 5 & 8 & 5 \\ 7 & 6 & 2 & 3 \end{bmatrix}.$$

Subtracting the minimum entry in each row, we obtain

$$\begin{bmatrix} 3 & 0 & 2 & 3 \\ 3 & 4 & 0 & 7 \\ 1 & 0 & 3 & 0 \\ 5 & 4 & 0 & 1 \end{bmatrix}.$$

Now subtracting the minimum entry in each column, we obtain

$$\begin{bmatrix} 2 & 0 & 2 & 3 \\ 2 & 4 & 0 & 7 \\ 0 & 0 & 3 & 0 \\ 4 & 4 & 0 & 1 \end{bmatrix}.$$

Making the assignments row by row, we have

$$\mathbf{C}' = \begin{bmatrix} 2 & 0^* & 2 & 3 \\ 2 & 4 & 0^* & 7 \\ 0^* & 0 & 3 & 0 \\ 4 & 4 & 0 & 1 \end{bmatrix}.$$

This matrix does not represent a complete assignment; person 4 has not been given a job. Two explanations are available for this situation. Either there is no possible complete assignment for the given pattern of zeros or there is a complete assignment but the algorithm failed to find it. We investigate each of these possibilities.

First notice that any pattern of zeros in an $n \times n$ matrix has the property that all the zeros can be covered by n lines. For example, choose the n lines that each cover one column. Suppose that the zeros in the $n \times n$ matrix $\mathbf{C}$ can be covered with k lines, where $k < n$. Let a be the smallest of the uncovered entries of $\mathbf{C}$. We form a new matrix $\mathbf{C}'$ by subtracting a from the entries of each uncovered row and adding a to the entries of each covered column. Each uncovered entry of $\mathbf{C}$ has decreased by a, since it belongs to an uncovered row and an uncovered column. Each entry covered by one line has remained unchanged: either it belonged to a covered row and uncovered column and was not modified, or it belonged to an uncovered row and covered column and had a added to it and subtracted from it. Each entry in both a covered row and a covered column has increased by a. Thus $\mathbf{C}'$ has a zero entry in a position in which $\mathbf{C}$ did not have a zero and it might be possible to finish the assignment. The procedure for modifying $\mathbf{C}$ can be more simply stated: subtract a from each uncovered entry and add a to each doubly covered entry. For example, we can cover the last matrix as follows:

$$\begin{bmatrix} 2 & \not{0} & \not{2} & 3 \\ 2 & \not{4} & \not{0} & 7 \\ \not{0} & \not{0} & \not{3} & \not{0} \\ 4 & \not{4} & \not{0} & 1 \end{bmatrix}.$$

The smallest uncovered entry of $\mathbf{C}'$ is 1. Subtracting 1 from each uncovered entry and adding 1 to each doubly covered entry, we obtain the matrix

$$\begin{bmatrix} 1 & 0 & 2 & 2 \\ 1 & 4 & 0 & 6 \\ 0 & 1 & 4 & 0 \\ 3 & 4 & 0 & 0 \end{bmatrix}.$$

Now using the assignment algorithm on this last matrix, we have

$$\begin{bmatrix} 1 & 0^* & 2 & 2 \\ 1 & 4 & 0^* & 6 \\ 0^* & 1 & 4 & 0 \\ 3 & 4 & 0 & 0^* \end{bmatrix},$$

which is a complete assignment. △

We can be guided in our use of this procedure by the following theorem, proved by the graph theorist König.

THEOREM 5.4. *The maximum number of zeros that can be assigned is equal to the minimum number of lines that are needed to cover all the zeros.*

△

In Example 2, since we can cover all the zeros of matrix $\mathbf{C}'$ with three lines, it follows from Theorem 5.4 that at most three zeros can be assigned. We have determined such an assignment. It is impossible to assign four zeros with the given matrix $\mathbf{C}'$, and more zeros must be introduced using the procedure described above. The other possibility when we do not discover a complete assignment is that the row-searching algorithm has failed.

EXAMPLE 3. Consider the cost matrix for an assignment problem given by

$$\mathbf{C} = \begin{bmatrix} 4 & 2 & 9 & 7 \\ 7 & 8 & 5 & 6 \\ 3 & 3 & 4 & 1 \\ 7 & 5 & 2 & 6 \end{bmatrix}.$$

Subtracting the minimum entry in each row from that row and then the minimum entry in each column from that column, we obtain

$$\begin{bmatrix} 0 & 0 & 7 & 5 \\ 0 & 3 & 0 & 1 \\ 0 & 2 & 3 & 0 \\ 3 & 3 & 0 & 4 \end{bmatrix}.$$

Assigning the first zero entry in each row that does not lie in a previously assigned column, we get

$$\begin{bmatrix} 0^* & 0 & 7 & 5 \\ 0 & 3 & 0^* & 1 \\ 0 & 2 & 3 & 0^* \\ 3 & 3 & 0 & 4 \end{bmatrix},$$

which is not a complete assignment. However, there is a complete assignment for this matrix given by

$$\begin{bmatrix} 0 & 0^* & 7 & 5 \\ 0^* & 3 & 0 & 1 \\ 0 & 2 & 3 & 0^* \\ 3 & 3 & 0^* & 4 \end{bmatrix}.$$

Consequently, we must develop an algorithm to search it out. Before we do this, let us summarize our method of solution for an assignment problem as we now have it.

Step 1. Assuming that the cost matrix $\mathbf{C}$ has nonnegative entries and the problem is a minimization problem, subtract the smallest entry in each

row from that row and then subtract the smallest entry in each column from that column. The new matrix $\mathbf{C}'$ defines an assignment problem that has the same optimal solutions as $\mathbf{C}$, and $\mathbf{C}'$ has at least one zero in each row and column.

Step 2. For each row assign the first zero not in any previously assigned column. If n zeros have been assigned, stop; an optimal solution has been found.

Step 3. Assuming that fewer than n zeros have been assigned, determine whether reassigning some zeros will give a complete assignment. If it will, stop after the reassignment.

Step 4. If reassigning some zeros does not give a complete assignment, then find k lines ($k < n$) that cover all the zeros of $\mathbf{C}'$.

Step 5. Let a be the smallest uncovered entry in $\mathbf{C}'$. Rearrange the zeros in $\mathbf{C}'$ by subtracting a from each uncovered entry of $\mathbf{C}'$ and adding a to each doubly covered entry of $\mathbf{C}'$. Go to Step 2 to reassign all zeros.

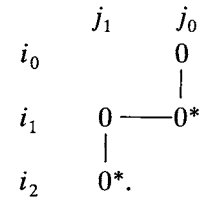
We now describe the details of Steps 3 and 4. The algorithm will require many searches of the rows and columns of $\mathbf{C}'$. All these searches are to be done from left to right or from top to bottom (in order of increasing subscripts).

Details of Step 3

Suppose that i_0 is the index of one of the rows for which, in Step 2, we fail to find a zero that can be assigned. However, there must be at least one zero in row i_0, since every row has a zero in it. Say that one of these zeros occurs in column j_0. There must be an assigned zero in column j_0, for otherwise the zero in (i_0, j_0) could be assigned. Say that this assigned zero occurs in row i_1. Starting at cell (i_0, j_0), we construct a path consisting of alternating vertical and horizontal segments, joining cells that alternately contain zeros and starred zeros. Specifically, we join

$$\begin{array}{llll} 0 & \text{in} & (i_0, j_0) & \text{to} \\ 0^* & \text{in} & (i_1, j_0) & \text{to} \\ 0 & \text{in} & (i_1, j_1) & \text{to} \\ 0^* & \text{in} & (i_2, j_1) & \ldots, \end{array}$$

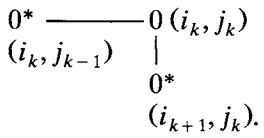
where the column indices $j_0, j_1, \ldots, j_n$ must be distinct:



We can represent this path by the sequence of cells (i_0, j_0), (i_1, j_0), (i_1, j_1), $(i_2, j_1), \ldots$. The next cell in the sequence is obtained as follows.

Case A. Suppose that we are at a 0 in (i_k, j_k). We search column j_k for a 0*. If a 0* is found, we add its cell to the sequence. If no 0* exists in column j_k, then we make the following reassignments in $\mathbf{C}'$: each 0 in the sequence from (i_0, j_0) to (i_k, j_k) is changed to a 0* and each 0* is changed to a 0. Note that we have created a 0* for row i_0 and that a 0* remains in every row that previously had one. We now repeat Step 3 for the next row in which there is no assigned zero.

Case B. Suppose, on the other hand, that we are at a 0* in cell (i_{k+1}, j_k). We search row i_{k+1} for a 0 that does not lie in a column appearing previously in the path. If a 0 is found, we add its cell to the sequence. If no 0 is found, we will not be able to modify the assignment as we did in Case A; we backtrack by labeling column j_k as **necessary** and redirecting our search. This is done by deleting the 0* at (i_{k+1}, j_k) and the 0 at (i_k, j_k) from the sequence. If $k \geq 1$, we return to the 0* at (i_k, j_{k-1}) and repeat this process with row i_k instead of row i_{k+1}. That is, we search row i_k for a 0 other than the 0 in column j_k (the necessary column) that does not lie in a column appearing previously in the path:



If $k = 0$, we search for another 0 in row i_0 that does not lie in a necessary column. If we find this 0, say in column j_0', we construct a path as described above starting at cell (i_0, j_0'). If we cannot find another suitable 0 in row i_0, a complete assignment has not been made.

There are two ways in which the construction of this sequence can be terminated. One way is when the 0's are changed to 0*'s and the 0*'s are changed to 0's. In this case we have found a 0 in row i_0 that can be assigned. The other way is when all the cells are deleted because they lie in

necessary columns. In this case, a complete assignment cannot be made with this pattern of zeros, and we must go to Step 4.

EXAMPLE 4. Consider the assignment problem whose cost matrix is

$$\mathbf{C} = \begin{bmatrix} 8 & 7 & 9 & 9 \\ 5 & 2 & 7 & 8 \\ 6 & 1 & 4 & 9 \\ 2 & 3 & 2 & 6 \end{bmatrix}.$$

We construct $\mathbf{C}'$, which has a zero in each row and column, by performing Step 1 of the algorithm. We subtract the smallest entry in each row from that row and then subtract the smallest entry in each column from that column, obtaining

$$\mathbf{C}' = \begin{bmatrix} 1 & 0 & 2 & 0 \\ 3 & 0 & 5 & 4 \\ 5 & 0 & 3 & 6 \\ 0 & 1 & 0 & 2 \end{bmatrix}.$$

We now assign zeros starting in row 1 as in Step 2. We find that row 2 is the first row that has no 0 in an unassigned column. At this point,

$$\mathbf{C}' = \begin{bmatrix} 1 & 0^* & 2 & 0 \\ 3 & 0 & 5 & 4 \\ 5 & 0 & 3 & 6 \\ 0^* & 1 & 0 & 2 \end{bmatrix}.$$

We then start constructing a sequence of 0's and 0*'s for Step 3. The first cell is (2, 2), which contains 0. We search column 2 for a 0* and find it in cell (1, 2). We search 1 for a 0 and find it in cell (1, 4). We search column 4 for a 0* and find none. Consequently we are in Case A with the sequence

$$\begin{aligned} 0 \quad &\text{in} \quad (2,2) \\ 0^* \quad &\text{in} \quad (1,2) \\ 0 \quad &\text{in} \quad (1,4), \end{aligned}$$

which represents the path

$$\begin{array}{l} (1,2)\ 0^* — 0\ (1,4). \\ \qquad\ \ | \qquad\quad . \\ (2,2)\ 0 \end{array}$$

Changing every 0 to 0* and every 0* to a 0 in the sequence, we obtain the matrix

$$\mathbf{C}' = \begin{bmatrix} 1 & 0 & 2 & 0^* \\ 3 & 0^* & 5 & 4 \\ 5 & 0 & 3 & 6 \\ 0^* & 1 & 0 & 2 \end{bmatrix},$$

where we increased the number of assignments by 1. We now repeat Step 3 with row 3.

There is no assignable 0 in row 3; therefore, we must construct a sequence starting at the 0 in cell (3, 2). The 0* in column 2 is in row 2. But there are no other 0's in row 2. We are in Case B and column 2 is necessary. Our sequence is

$$\begin{aligned} 0 \quad & \text{in} \quad (3,2) \\ 0^* \quad & \text{in} \quad (2,2). \end{aligned}$$

Since all the cells in our sequence lie in a necessary column, we must go to Step 4 to determine the necessary rows.

A row is called **necessary** if it contains a 0* in an unnecessary column. Starting with row 1, we find that its 0* is in column 4, and consequently row 1 is necessary. Row 2 has its 0* in column 2, which is a necessary column. Therefore, row 2 is not necessary. Row 3 has no 0* so it is not necessary. Row 4 has its 0* in column 1 and consequently is necessary.

Details of Step 4

Covering each necessary row and column with a line provides the k lines previously described. This procedure automatically covers all zeros of $\mathbf{C}'$.

We find that the $\mathbf{C}'$ of our example is covered as follows:

$$\mathbf{C}' = \begin{bmatrix} \cancel{1} & \cancel{0} & \cancel{2} & \cancel{0^*} \\ 3 & 0^* & 5 & 4 \\ 5 & 0 & 3 & 6 \\ \cancel{0^*} & \cancel{1} & \cancel{0} & \cancel{2} \end{bmatrix}.$$

We are now ready for Step 5. Subtract 3, the smallest uncovered entry, from each uncovered entry and add 3 to each doubly covered entry. Our new $\mathbf{C}'$, ready for Step 2, is

$$\mathbf{C}' = \begin{bmatrix} 1 & 3 & 2 & 0 \\ 0 & 0 & 2 & 1 \\ 2 & 0 & 0 & 3 \\ 0 & 4 & 0 & 2 \end{bmatrix}.$$

At the completion of the assignment procedure in Step 2 we have

$$\begin{bmatrix} 1 & 3 & 2 & 0^* \\ 0^* & 0 & 2 & 1 \\ 2 & 0^* & 0 & 3 \\ 0 & 4 & 0^* & 2 \end{bmatrix},$$

which is a complete assignment. △

We can now rewrite Step 3 in the algorithm: Assume that no zero in row i_0 has been assigned and that there is a zero in cell (i_0, j_0). Construct a sequence of vertical and horizontal segments that alternate and join 0 to 0* to 0 to 0* and so on as follows.

(A) If we are at 0 in cell (i_k, j_k), search column j_k for 0*. If 0* is found, adjoin its cell to the sequence. If 0* is not found, change each 0 in the sequence to 0* and each 0* to 0 and search for the next row without a 0*.

(B) If we are at 0* in cell (i_{k+1}, j_k), search row i_{k+1} for 0. If 0 is found, adjoin its cell to the sequence. If 0 is not found, label column j_k as necessary and delete cells (i_k, j_k) and (i_{k+1}, j_k) from the sequence. If there are more cells, we are at 0* in cell (i_k, j_{k-1}). Repeat Case B. If there are no more cells in the sequence, search row i_0 for a 0 that does not lie in a necessary column. If one is found, say in column j_0', repeat Step 3 starting at cell (i_0, j_0'). If one is not found, go to Step 4.

The algorithm that we have developed is called the **Hungarian method** in honor of the mathematicians König and Egerváry, on whose work it is based. The Hungarian method as we have described it assumes that the given assignment problem is a minimization problem. However, a maximization problem can be modified in a way that will enable the Hungarian method to produce an optimal solution. Suppose we are given the problem

$$\text{Maximize} \quad z = \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij}$$

subject to

$$\left.\begin{aligned} \sum_{j=1}^{n} x_{ij} &= 1, \quad i = 1, 2, \ldots, n \\ \sum_{i=1}^{n} x_{ij} &= 1, \quad j = 1, 2, \ldots, n \end{aligned}\right\} \tag{5}$$

$$x_{ij} \geq 0 \quad \text{integers.}$$

We can convert this problem to the minimization problem

$$\text{Minimize} \quad z = \sum_{i=1}^{n} \sum_{j=1}^{n} (-c_{ij}) x_{ij}$$

subject to the constraints in (5).

If some of the entries in the matrix $[c_{ij}]$ were positive, we now have negative entries in $[-c_{ij}]$, and the optimality criterion in Theorem 5.2 does not apply. However, we can use Theorem 5.3 and add to each entry in $[-c_{ij}]$ the negative of the smallest (most negative) entry. This new matrix will have nonnegative entries, and the assignment problem for it can be solved using the Hungarian method.

EXAMPLE 5. Suppose an assignment problem asks one to maximize the total value of the assignment for which the individual values are given by

$$[c_{ij}] = \begin{bmatrix} 3 & 7 & 4 & 6 \\ 5 & 2 & 8 & 5 \\ 1 & 3 & 4 & 7 \\ 6 & 5 & 2 & 6 \end{bmatrix}.$$

The corresponding minimization problem is

$$\text{Minimize} \quad z = \sum_{i=1}^{n} \sum_{j=1}^{n} (-c_{ij}) x_{ij}$$

subject to the constraints in (5).

The smallest entry in $[-c_{ij}]$ is -8; thus, we add 8 to each entry in $[-c_{ij}]$ to obtain

$$\begin{bmatrix} 5 & 1 & 4 & 2 \\ 3 & 6 & 0 & 3 \\ 7 & 5 & 4 & 1 \\ 2 & 3 & 6 & 2 \end{bmatrix}.$$

This matrix is now used as the cost matrix for the Hungarian method. △

5.2 EXERCISES

In Exercises 1–6 solve the assignment problem with the given cost matrix.

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

2. $\begin{bmatrix} 3 & 2 & 5 & 8 & 9 \\ 6 & 7 & 4 & 2 & 3 \\ 5 & 3 & 5 & 4 & 2 \\ 4 & 7 & 3 & 2 & 4 \\ 2 & 6 & 5 & 5 & 3 \end{bmatrix}$

3. $\begin{bmatrix} 3 & 4 & 0 & 2 & 6 & 7 \\ 4 & 6 & 4 & 5 & 3 & 6 \\ 5 & 7 & 7 & 8 & 2 & 8 \\ 0 & 8 & 8 & 4 & 6 & 4 \\ 6 & 4 & 3 & 7 & 4 & 9 \\ 7 & 5 & 5 & 0 & 6 & 7 \end{bmatrix}$

4. $\begin{bmatrix} 3 & 2 & 7 & 4 & 8 \\ 5 & 4 & 3 & 8 & 5 \\ 3 & 7 & 9 & 1 & 2 \\ 4 & 2 & 6 & 5 & 7 \\ 2 & 8 & 4 & 6 & 6 \end{bmatrix}$

5. $\begin{bmatrix} 3 & 5 & 4 & 2 & 8 & 1 \\ 8 & 3 & 6 & 6 & 4 & 3 \\ 4 & 4 & 8 & 8 & 3 & 5 \\ 3 & 8 & 7 & 4 & 9 & 7 \\ 7 & 7 & 9 & 2 & 3 & 5 \\ 9 & 7 & 2 & 7 & 5 & 8 \end{bmatrix}$

6. $\begin{bmatrix} 9 & 7 & 4 & 7 & 3 \\ 0 & 8 & 3 & 5 & 8 \\ 6 & 3 & 2 & 8 & 9 \\ 5 & 6 & 7 & 0 & 4 \\ 2 & 9 & 5 & 7 & 3 \end{bmatrix}$

7. A company leases offices along one side of a hall in a large office building. Suppose that there are 15 offices all the same size leased by the company and that they are numbered 701 through 715. Because of a recent series of job reassignments the desks in some of the offices must be moved. Specifically, the desks in 702, 705, 708, 709, and 713 must be moved to 706, 707, 712, 714, and 715, but it does not matter which desk goes to which office. Consequently, the facilities supervisor has decided to assign desks in such a way as to minimize the total distance over which they would need to be moved. What assignment should the supervisor use?

5.2 PROJECTS

1. Consider the problem of scheduling the full-time registered nurses on a surgical floor of a hospital. Adequate patient care requires that there be four RNs during the day, two RNs in the evening, and one RN at night. Assume that the scheduling is done two weeks at a time. In the two-week planning horizon each nurse must work 10 shifts.

   (a) How many nurses are required?

   (b) A schedule for a nurse is a list indicating which of the 42 shifts (why 42?) are to be worked by that nurse. How many possible schedules are there? Which of these schedules would be less desirable? What guidelines might be used for making up acceptable schedules for the nurses?

   (c) Assume that the decisions about schedules for each nurse are made in the following manner: Each nurse is given a list of 10 possible schedules following the guidelines from part (b) and asked to rank them in order of desirability. Each nurse gets a different list of schedules. Each nurse's first choice is to be assigned a weight of 10, the second a weight of 9, and so on. How might ties be handled? The choice of one schedule for each nurse from among those presented is determined by maximizing the sum of the weights of the chosen schedules. Set up a model for this situation.

   (d) Estimate the human time and the computer time involved in determining the schedule for staffing the floor in this way. (Computer time estimates can come from the size of the assignment problem.)

2. Consider the following generalizations of the desk-moving problem (see Karp and Lee in Further Reading) in Exercise 7.

   (a) Assume that there are N offices from which desks must be moved. Let $\mathbf{D} = [d_{ij}]$ be the matrix of distances from office i to office j. Suppose that the current locations are $s_1 < s_2 < \cdots < s_N$, where " < " means ordered from near to far from the elevator along the hall. Suppose that the future locations are $t_1 < t_2 < \cdots < t_N$. Show that an optimal assignment is $s_k \to t_k$, $k = 1, 2, \ldots, N$.

   (b) Suppose that, instead of being located along a hall, the offices are located around the entire perimeter of a floor of the office building. Suppose that the distance to walk completely around the floor is L and that $\mathbf{D} = [d_{ij}]$ is the matrix of shortest distances from location i to location j. What is the minimal-distance assignment?

Further Reading

Karp, R. M., and Li, S.-Y. "Two Special Cases of the Assignment Problem." *Discrete Math.* **13** (1975), 129–142.

Kuhn, H. W. "The Hungarian Method for the Assignment Problem." *Naval Res. Logistics Q.* **2** (1955), 83–97.

Kuhn, H. W. "Variants of the Hungarian Method for the Assignment Problem." *Naval Res. Logistics Q.* **3** (1956), 253–258.

5.3 GRAPHS AND NETWORKS: BASIC DEFINITIONS

A **graph** is a collection of points, called **nodes** (also called **vertices**), some of which (possibly all) are joined by lines, called **arcs**, **edges**, or **branches**. Every graph can be conveniently represented by a figure in the plane. Thus, Figure 5.1 shows a graph with seven nodes and nine arcs. We may note that the arc joining nodes 1 and 3 and the arc joining nodes 2 and 4 intersect at a point that is not a node in this representation of the graph. The arc joining nodes i and j will be denoted by (i, j).

An arc of the form (a, a) is called a **loop**. We will not allow loops in our graphs. If the nodes of a graph are numbered consecutively from 1 to n,

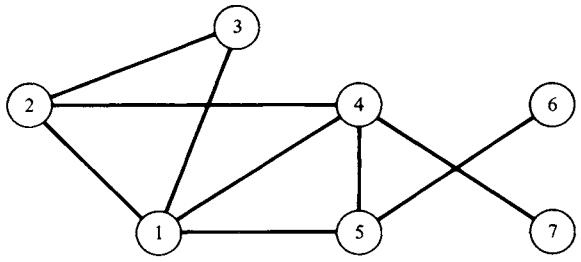


FIGURE 5.1

the graph can be represented by a matrix. The **incidence matrix** of a graph G is the matrix $\mathbf{M} = [m_{ij}]$, where

$$m_{ij} = \begin{cases} 1 & \text{if node } i \text{ is connected to node } j \\ 0 & \text{otherwise.} \end{cases}$$

For an arbitrary graph, the incidence matrix is symmetric ($\mathbf{M} = \mathbf{M}^{\mathrm{T}}$). Why? The incidence matrix for the graph in Figure 5.1 is

$$\mathbf{M} = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}.$$

A **path** between node i and node j of a graph is an ordered set of arcs

$$(i, a_1), (a_1, a_2), (a_2, a_3), \ldots, (a_r, j)$$

joining i and j. Thus, a path between node 2 and node 6 in Figure 5.1 is

$$(2,4), (4,5), (5,6).$$

Other paths between node 2 and node 6 are

$$(2,1), \quad (1,4), \quad (4,5), \quad (5,6)$$
$$(2,1), \quad (1,5), \quad (5,6)$$

and

$$(2,3), \quad (3,1), \quad (1,4), \quad (4,5), \quad (5,6).$$

A **cycle** is a path joining a node to itself. Examples of cycles in Figure 5.1 are

$$(2,3), \quad (3,1), \quad (1,2)$$

and

$$(2,4), \quad (4,5), \quad (5,1), \quad (1,2).$$

A graph is said to be **connected** if there is a path joining any two nodes of the graph. The graph shown in Figure 5.1 is connected, whereas the graph shown in Figure 5.2 is not connected.

An arc of a graph is called **directed** or **oriented** if there is a sense of direction so that one node is considered the point of origin and the other node is the point of termination. The directed arc from node i to node j will be denoted by $\overrightarrow{(i, j)}$. A graph in which every arc is directed is called a **directed graph**, a **digraph**, or an **oriented graph**. An example of a directed graph is shown in Figure 5.3.

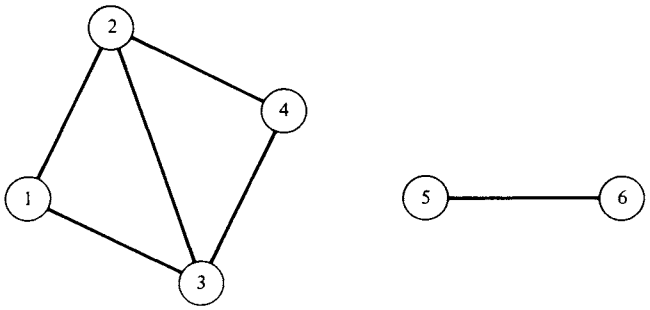


FIGURE 5.2

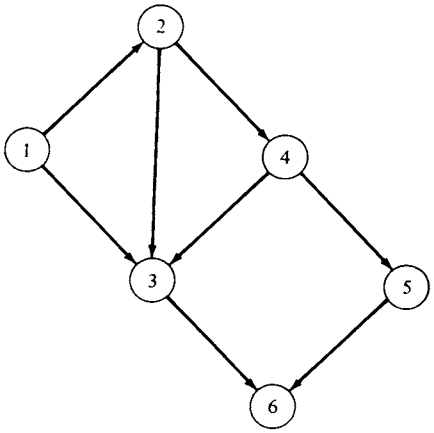


FIGURE 5.3

An incidence matrix can also be used to represent a directed graph. In this case $m_{ij} = 1$ if a directed arc connects i to j. Consequently, the incidence matrix of a digraph is usually not symmetric. The incidence matrix of the digraph in Figure 5.3 is

$$\mathbf{M} = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

For certain applications of linear programming we are interested in the idea of a network. Intuitively, a network is an interconnection of several terminals by routes between certain pairs of these terminals. Each route has a capacity, and we are interested in studying the movement or flow of

material along these routes. We can study pipeline networks, highway networks, and electrical power distribution networks, as well as many other kinds.

More formally, a **network** is a connected, directed graph for which a nonnegative number has been assigned to each ordered pair of nodes. This number is thought of as the **capacity** of the directed arc joining the two nodes. If node i is not connected to node j by an arc, the capacity c_{ij} is set to zero. The capacities represent maximum amounts that may pass along arcs of networks; specifically, they may be tons of oil/hr in a transcontinental oil pipeline network, cubic meters of water/min in a city water system, pulses/sec in a communications network, or number of vehicles/hr on a regional highway system. The nodes may represent shipping depots, relay stations, highway interchanges, or pumping stations. A network may be represented by its **capacity matrix**, which is a generalization of the incidence matrix of a graph and consists of the matrix of capacities of the arcs. The directed graph in Figure 5.3 becomes a network when we specify its capacity matrix as

$$\mathbf{C} = \begin{bmatrix} 0 & 7 & 5 & 0 & 0 & 0 \\ 0 & 0 & 2 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 8 \\ 0 & 0 & 8 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 6 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

If there is no limit on the amount passing from node i to node j in a network, the capacity c_{ij} is set equal to a very large number M.

If flow is permitted in both directions between a pair of nodes (as in a two-way street system), the nodes are connected with two directed arcs, one going in each direction, each with its own capacity. That is, a two-way street is thought of as two one-way streets.

A **flow** in a network is an assignment to each ordered pair of nodes (i, j) in the network of a nonnegative number x_{ij} that represents the amount of material moving in that directed arc. If node i is not connected to node j, then $x_{ij} = 0$. By definition, the flow may not exceed the capacity, so that we have for each i and $j, i, j = 1, 2, \ldots, n$,

$$0 \leq x_{ij} \leq c_{ij}.$$

In many networks we single out two types of nodes for special consideration. The node S is called a **source** if every arc joining S to another node is oriented so that the flow is away from S. The node T is called a **sink** if every arc joining T to another node is directed so that the flow is toward T. That is, a flow is produced at a source and it is absorbed at a sink.

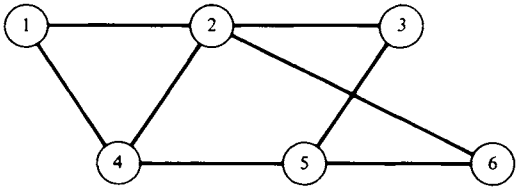
We also specify that any flow cannot cause material to accumulate at any node other than a source or a sink. That is, with the exception of a source or a sink, the flow into a node is equal to the flow out of the node. If we have a network with exactly one source and exactly one sink, number the source as node 1 and the sink as node n. Then the flow must satisfy, for each node k, $k = 2, \ldots, n-1$,

$$\sum_{i=1}^{n} x_{ik} = \sum_{j=1}^{n} x_{kj}.$$

In the next few sections we will consider certain questions about flows in networks and certain situations that can be modeled with such a mathematical structure.

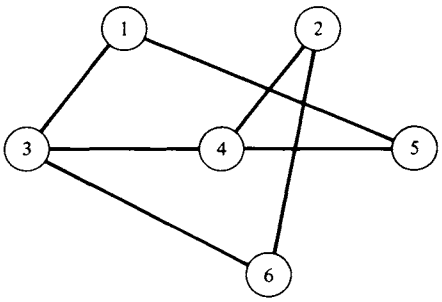
5.3 EXERCISES

1. For the graph



(a) find its incidence matrix;
(b) find three paths joining nodes 1 and 4;
(c) find two cycles from node 2.

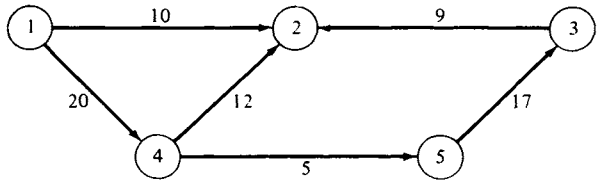
2. Follow the instructions in Exercise 1 for the following graph.



3. Sketch the graph whose incidence matrix is

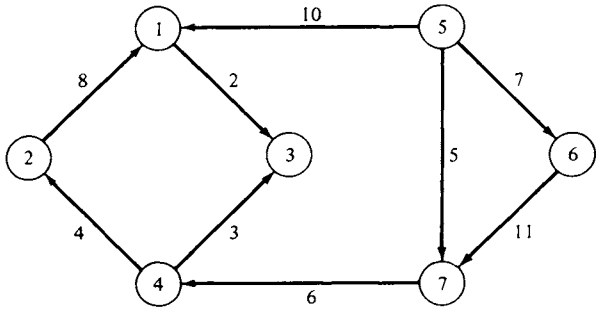
(a) $\begin{bmatrix} 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \end{bmatrix}$ (b) $\begin{bmatrix} 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 & 0 \end{bmatrix}$.

4. Find the incidence matrix for the graph in Figure 5.2.
5. For the network



where the numbers on the edges are the capacities,
(a) find the capacity matrix;
(b) find the sources;
(c) find the sinks.

6. Follow the instructions for Exercise 5 for the following network.



7. Show that node i is a source in a network if and only if the ith column of the capacity matrix of the network is zero.
8. Show that node j is a sink in a network if and only if the jth row of the capacity matrix of the network is zero.

Further Reading

Wilson, Robin J. *Introduction to Graph Theory*. Academic Press, New York, 1972.

5.4 THE MAXIMAL FLOW PROBLEM

Consider a network with n nodes that includes a single source and a single sink. For convenience we label the source node 1 and the sink node n. Let c_{ij} denote the capacity of arc (i, j). Consider a flow in this network, letting x_{ij} denote the amount of material flowing from node i to node j along arc (i, j). As we discussed in Section 5.3, the x_{ij} must satisfy

$$0 \leq x_{ij} \leq c_{ij}, \quad i, j = 1, 2, \ldots, n \tag{1}$$

and

$$\sum_{i=1}^{n} x_{ik} = \sum_{j=1}^{n} x_{kj}, \quad k = 2, 3, \ldots, n-1. \tag{2}$$

We may also write (2) as

$$\sum_{i=1}^{n} x_{ik} - \sum_{j=1}^{n} x_{kj} = 0, \quad k = 2,3,\ldots,n-1. \tag{2a}$$

The total flow starting from the source, which is to be maximized, is

$$f = \sum_{k=1}^{n} x_{1k}. \tag{3}$$

The total flow into the sink is

$$\sum_{k=1}^{n} x_{kn},$$

which by the conservation of flow is precisely the expression on the right side of (3) (verify). Thus, a mathematical formulation of the **maximal flow problem** is

$$\text{Maximize} \quad f = \sum_{k=1}^{n} x_{1k}$$

subject to

$$\sum_{i=1}^{n} x_{ik} - \sum_{j=1}^{n} x_{kj} = 0 \quad k = 2,3,\ldots,n-1 \tag{4}$$

$$0 \le x_{ij} \le c_{ij}, \quad i,j = 1,2,\ldots,n.$$

The mathematical formulation of the maximal flow problem shows that we have a linear programming problem, which could be solved by the simplex method. However, this approach is quite inefficient and in this section we present several better procedures.

Before turning to the computational procedures, we note that the maximal flow problem occurs in many applications. As a typical applied problem, consider an electrical power distribution system represented by a network that has one source and many sinks. When a brownout appears imminent at a particular location (node), that location (node) is made a sink and the goal becomes to send as much electrical power as possible to this endangered location. Thus, we have a maximal flow problem.

The following intuitive method appears to offer some hope for solving the maximal flow problem. We start at the source, and by proceeding along arcs with positive capacity we find a path from source to sink. (Why can we find this path?) The maximum amount f_1 of material that can be sent along this path is the minimum of the capacities of the arcs in the path. We now subtract f_1 from the capacity of each arc in the path just used. The capacity of at least one of the arcs in this path is now reduced to zero. We next return to the source and proceed along arcs with positive capacity

to find another path to the sink. The maximum amount f_2 of material that can be sent along this path is the minimum of the capacities of the arcs in the path. The capacity of at least one of the arcs in this path is now reduced to zero by subtracting f_2 from each capacity in the path. We continue choosing possible paths from source to sink until there are no more paths all of whose arcs have positive capacity. The total flow f is the sum of the flows

$$f = f_1 + f_2 + \cdots + f_k.$$

To illustrate this intuitive method, consider the network with source 1 and sink 6 (Figure 5.4). We first choose the path

$$1 \to 3 \to 2 \to 4 \to 6,$$

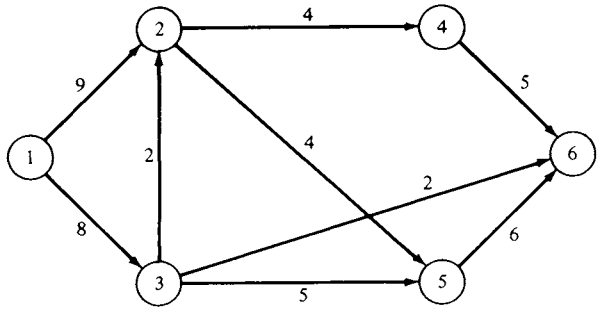


FIGURE 5.4

which has flow $f_1 = 2$. Subtracting f_1 from the capacity of each arc in this path, we obtain the network in Figure 5.5.

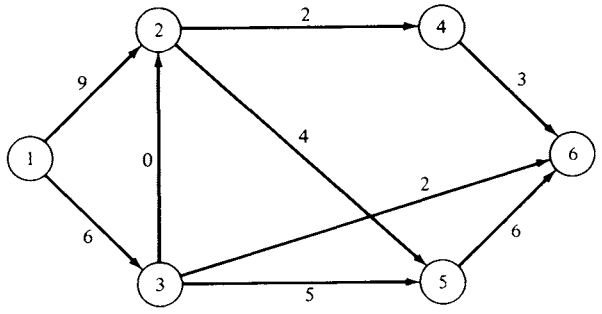


FIGURE 5.5

Next, choose the path

$$1 \to 3 \to 5 \to 6,$$

whose flow $f_2 = 5$ yields the network in Figure 5.6.

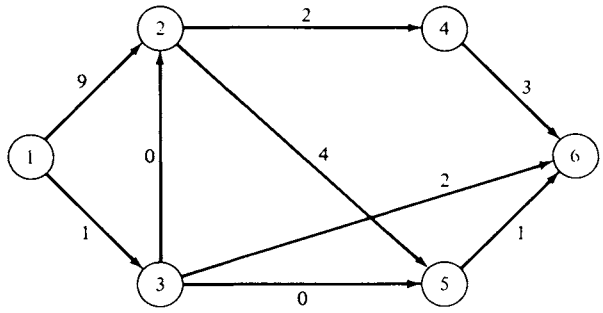


FIGURE 5.6

Now choose the path

$$1 \rightarrow 2 \rightarrow 5 \rightarrow 6$$

with flow $f_3 = 1$, obtaining the network in Figure 5.7.

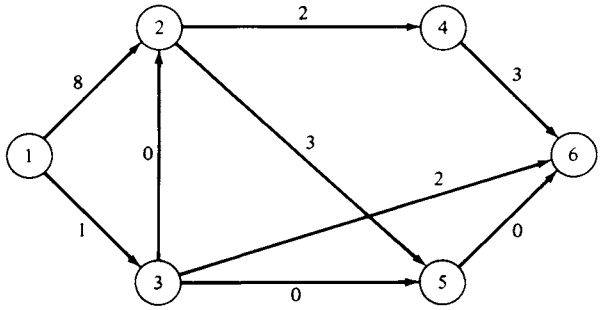


FIGURE 5.7

For the next path we have

$1 \rightarrow 2 \rightarrow 4 \rightarrow 6$ with flow $f_4 = 2$ (Figure 5.8).

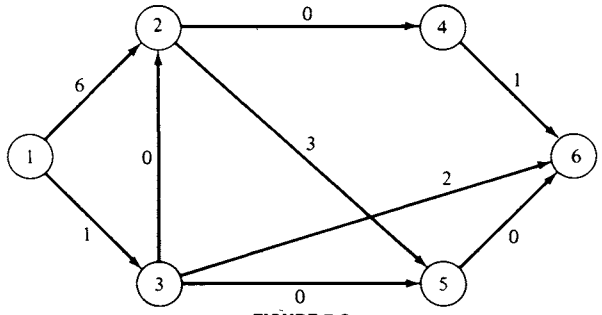


FIGURE 5.8

We now choose the path

$1 \rightarrow 3 \rightarrow 6$ with flow $f_5 = 1$ (Figure 5.9).

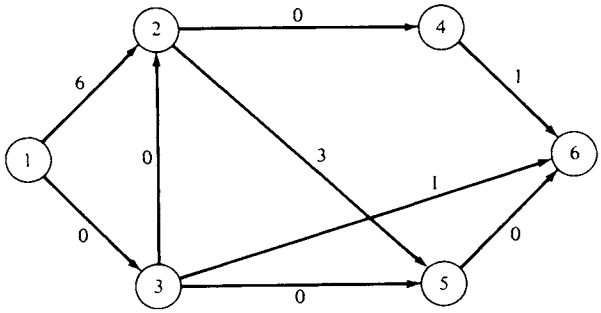


FIGURE 5.9

Since we cannot find any other paths from source to sink, all of whose arcs have positive capacities, we are finished. The total flow is

$$f = f_1 + f_2 + f_3 + f_4 + f_5 = 2 + 5 + 1 + 2 + 1 = 11.$$

However, suppose that, instead of choosing the above sequence of paths from source to sink, we choose the sequence of paths indicated below:

$1 \rightarrow 2 \rightarrow 4 \rightarrow 6$, with flow $f_1 = 4$ (Figure 5.10)

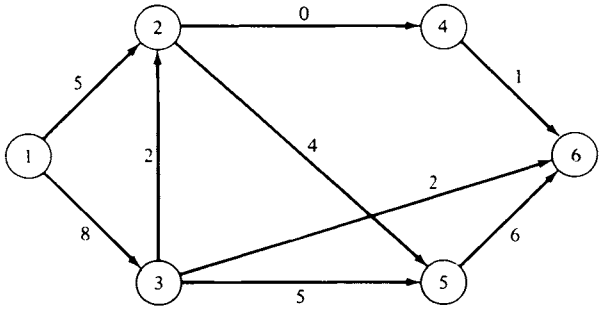


FIGURE 5.10

$1 \rightarrow 3 \rightarrow 5 \rightarrow 6$, with flow $f_2 = 5$ (Figure 5.11)

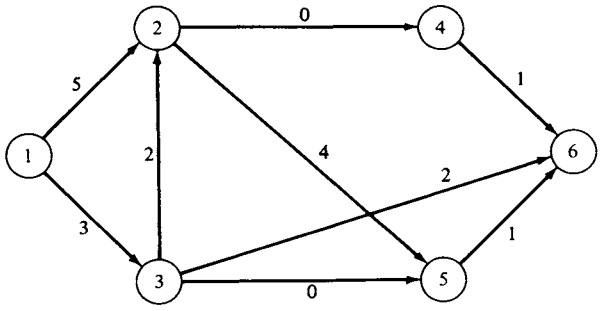


FIGURE 5.11

$1 \rightarrow 3 \rightarrow 6$, with flow $f_3 = 2$ (Figure 5.12)

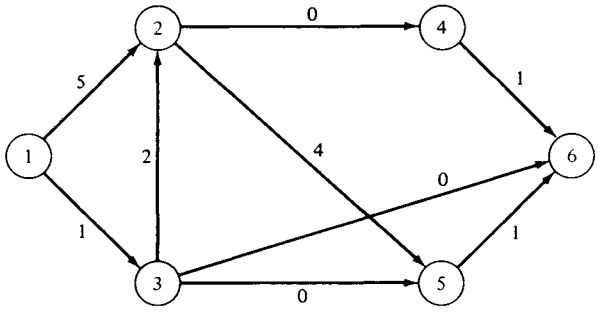


FIGURE 5.12

$1 \rightarrow 2 \rightarrow 5 \rightarrow 6$, with flow $f_4 = 1$ (Figure 5.13).

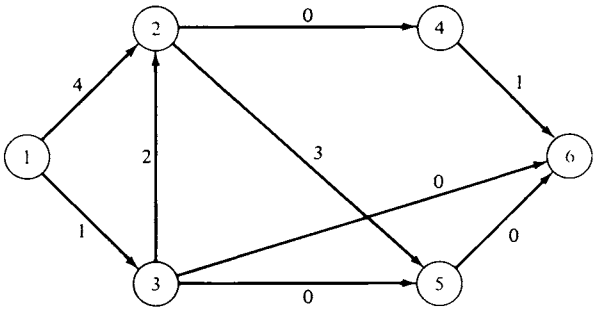


FIGURE 5.13

Since we cannot find any other paths from source to sink, all of whose arcs have positive capacities, we are finished. The total flow is

$$f = f_1 + f_2 + f_3 + f_4 = 4 + 5 + 2 + 1 = 12.$$

Thus, the intuitive method does not always yield the maximal flow. However, it can be modified to a correct algorithm by allowing fictitious flows in the opposite direction, so that an alternate path can be chosen. This principle is at the heart of the labeling procedure (sometimes called the *augmenting path method*) of Ford and Fulkerson. This procedure was developed in 1957 and was based on the earlier results of Kuhn and Egerváry (see Further Reading). In 1969 a substantial improvement in the augmenting path method was reported by Dinic, and many of the subsequent improvements in the algorithms for solving the maximum flow problem have been based on his approach. There is now great interest in investigating the use of parallel processing to obtain further improvements. Some of the initial ideas in this direction are discussed in Goldberg and Tarjan.

All these improved algorithms involved advanced ideas in combinatorial programming and complexity theory and consequently are beyond the

scope of this book. The interested reader may see the books by Chvátal and by Papadimitriou and Steiglitz for more details.

The Labeling Algorithm

We now describe the labeling procedure of Ford and Fulkerson. It attempts to construct a feasible flow from source to sink and then to augment that flow to the capacity of the path, perhaps redirecting some of the flow and changing to a path of greater capacity. We continue to number the nodes so that node 1 is the source and node n is the sink. The directed arc joining node i to j has capacity c_{ij}. If flow cannot take place from i to j, then $c_{ij} = 0$. For each arc we now define the **excess capacity**

$$d_{ij} = c_{ij} - x_{ij} + x_{ji}.$$

We begin with all flows set at value zero. Next label every directed arc with its excess capacity d_{ij}.

Step 1. Starting at the source, we form the set N_1 of all nodes that are connected to the source by an arc with positive excess capacity. We use the index k for nodes in N_1. Now label each node in N_1 with the ordered pair of numbers (e_k, p_k), where

$e_k = d_{1k}$ = excess capacity of the arc from the source to node k

p_k = node that led to node k.

Here $p_k = 1$ for each node in N_1, because we got to this node from the source. If we have labeled node n, the sink of the network, we then proceed directly to Step 5, where we increase the flow.

Step 2. Choose the node in N_1 with smallest index; say it is node k. Let N_2 denote the set of all unlabeled nodes that are joined to node k by an arc with positive excess capacity. From now on we must assume that the source is a labeled node. If there are no such unlabeled nodes, we pick the node in N_1 with the next smallest index and again form N_2. We use the index m for each node in N_2. Label each unlabeled node in N_2 with the ordered pair (e_m, p_m), where

$$e_m = \min\{d_{km}, e_k\}$$
$$p_m = k.$$

Observe that e_m is the minimum of the excess capacities of the arcs from the source to node k and from node k to node m. Also, p_m denotes the node that led to node m. We repeat for each node in N_1.

Step 3. Repeat Step 2 with N_r replacing N_{r-1}. After a finite number of steps, we arrive at one of two possibilities.

(i) The sink has not been labeled and no other nodes can be labeled.

(ii) The sink has been labeled.

Step 4. If we are in case (i), then the current flow can be shown to be maximal and we stop.

Step 5. If we are in case (ii), then we have an augmenting path whose flow we increase as follows. Suppose that the sink has the label (e_r, p_r). The first number in the label, e_r, indicates the amount by which we can increase the flow. The second number in the label, p_r, gives the node that led to the sink, making it possible to move backwards along this path to the source. Let d_{st} denote the excess capacities of the arcs in the path P. To increase the flow by e_r we now calculate the excess capacities as

$$\begin{aligned} d'_{st} &= d_{st} - e_r \\ d'_{ts} &= d_{ts} + e_r \\ d'_{ij} &= d_{ij} \quad \text{for arcs not in } P. \end{aligned}$$

Step 6. Return to Step 1.

Assuming that a maximal flow exists, the algorithm terminates after a finite number of iterations. That is, after a finite number of iterations we reach case (i).

We now calculate the net flows in each arc as follows. If $c_{ji} = 0$, so that flow cannot take place from node j to node i, then the flow in the arc from i to j is

$$x_{ij} = c_{ij} - d_{ij},$$

where d_{ij} is the most recent excess capacity calculated as described in Step 5. If both c_{ij} and c_{ji} are positive, so that flow can take place from i to j as well as from j to i, observe that

$$\begin{aligned} c_{ij} - d_{ij} &= x_{ij} - x_{ji} \\ c_{ji} - d_{ji} &= x_{ji} - x_{ij} = -(c_{ij} - d_{ij}). \end{aligned}$$

Hence, $c_{ij} - d_{ij}$ and $c_{ji} - d_{ji}$ cannot both be positive. We let

$$\left.\begin{aligned} x_{ij} &= c_{ij} - d_{ij} \\ x_{ji} &= 0 \end{aligned}\right\} \quad \text{if} \quad c_{ij} - d_{ij} \geq 0$$

$$\left.\begin{aligned} x_{ji} &= c_{ji} - d_{ji} \\ x_{ij} &= 0 \end{aligned}\right\} \quad \text{if} \quad c_{ji} - d_{ji} \geq 0.$$

We illustrate the labeling method with the network considered in the intuitive method. Start with all flows set at value zero. Figure 5.14 shows the given network with each arc labeled with its excess capacity.

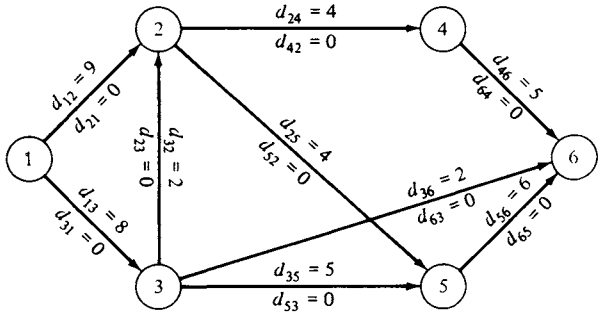


FIGURE 5.14

Step 1. Starting at the source, node 1, we find all nodes that are connected to it by an arc with positive excess capacity. These are nodes 2 and 3. Thus,

$$e_2 = d_{12} = 9, \quad e_3 = d_{13} = 8$$
$$p_2 = 1, \quad p_3 = 1.$$

We now label nodes 2 and 3 with the respective ordered pairs (9, 1) and (8, 1), shown in Figure 5.15.

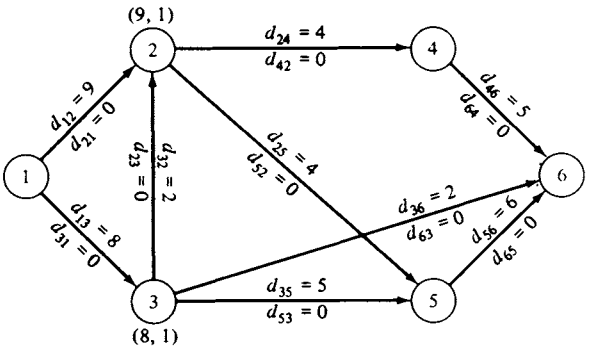


FIGURE 5.15

Step 2. Starting from node 2, we find all unlabeled nodes that are joined to node 2 by an arc with positive excess capacity. These are nodes 4 and 5. The label on node 4 is (e_4, p_4), where

$$e_4 = \min\{d_{24}, e_2\}$$
$$= \min\{4, 9\} = 4$$
$$p_4 = 2.$$

Similarly, the label on node 5 is (e_5, p_5), where

$$\begin{aligned} e_5 &= \min\{d_{25}, e_2\} \\ &= \min\{4, 9\} = 4 \\ p_5 &= 2. \end{aligned}$$

We proceed in the same manner from node 3. The only unlabeled node that can be reached from node 3 is node 6, the sink. This node is labeled (e_6, p_6), where

$$\begin{aligned} e_6 &= \min\{d_{36}, e_3\} \\ &= \min\{2, 8\} = 2 \\ p_6 &= 3. \end{aligned}$$

The network in Figure 5.16 shows the labels obtained in Step 2. Observe that we have labeled the sink.

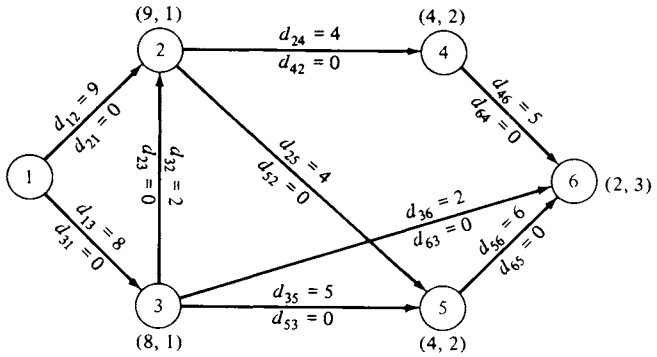


FIGURE 5.16

Since we have labeled the sink, we proceed to Step 5. We can increase the flow from its initial value of 0 by $e_6 = 2$ units to total flow of 2 units. We now move backward to the source and calculate new excess capacities as follows. Since $p_6 = 3$, we go back to node 3 and let

$$\begin{aligned} d'_{36} &= d_{36} - e_6 = 2 - 2 = 0 \\ d'_{63} &= d_{63} + e_6 = 0 + 2 = 2. \end{aligned}$$

Since $p_3 = 1$, we go back to node 1, the source, and let

$$\begin{aligned} d'_{13} &= d_{13} - e_6 = 8 - 2 = 6 \\ d'_{31} &= d_{31} + e_6 = 0 + 2 = 2. \end{aligned}$$

The network in Figure 5.17 shows the new excess capacities. We now return to Step 1 and relabel the nodes. In Figure 5.17 we have also indicated these new labels.

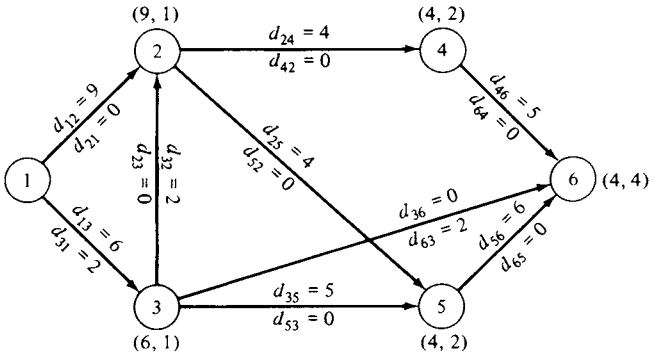


FIGURE 5.17

The sink has again been labeled. We can increase the flow by 4 units to a total flow of 6 units. We move backward along the path

$$1 \rightarrow 2 \rightarrow 4 \rightarrow 6$$

and calculate new excess capacities, which are indicated in the network in Figure 5.18. In this network we have also indicated the newest labels on the nodes.

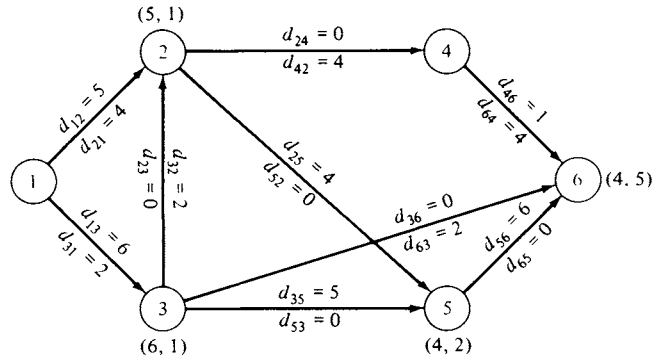


FIGURE 5.18

The sink has been labeled and evidently we can increase the flow by 4 units to a total flow of 10 units. We move backward along the path

$$1 \rightarrow 2 \rightarrow 5 \rightarrow 6$$

and calculate new excess capacities, which are indicated in the network in Figure 5.19. In this network we have also indicated the newest labels on the nodes.

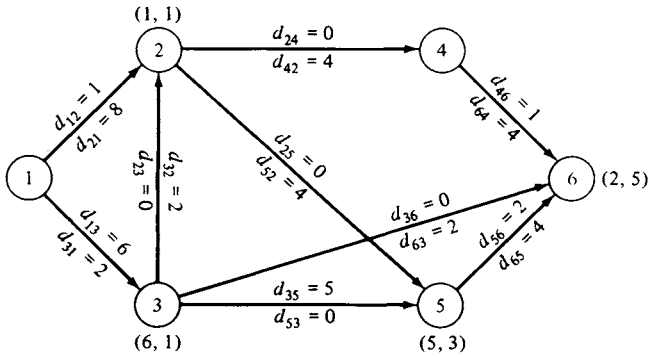


FIGURE 5.19

We have labeled the sink and can now increase the flow by 2 units to a total of 12 units. We move backward along the path

$$1 \rightarrow 3 \rightarrow 5 \rightarrow 6$$

and calculate new excess capacities, which are indicated in the network in Figure 5.20. In this network we have also indicated the newest labels on the nodes.

At this point, the sink has not been labeled and no other nodes can be labeled. The current flow of 12 units is maximal and we stop. The net flow in each arc is shown in the network in Figure 5.21. △

We shall now show that if the above procedure yields case (i), then the current flow is optimal. We must start with another definition. A **cut** in a network is a set of directed arcs with the property that every path from the source to the sink contains at least one arc from the set. Since the number of directed arcs in a network is finite, the number of cuts is also finite.

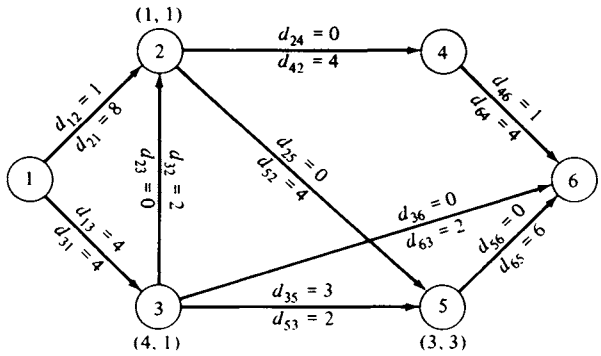


FIGURE 5.20

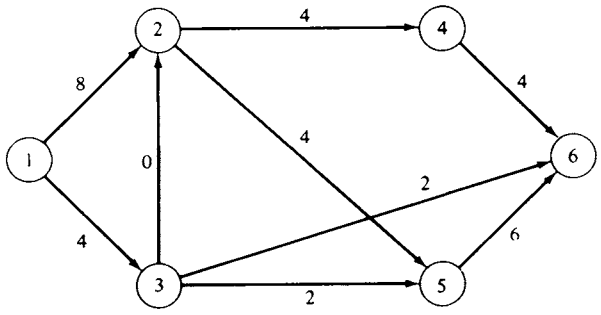


FIGURE 5.21

From this finite set of cuts we shall soon single out one cut. We define the **capacity** of a cut as the sum of the capacities of its directed arcs.

EXAMPLE 1. Consider the network in Figure 5.22. The set of directed arcs

$$\{(\overrightarrow{4,6}), (\overrightarrow{2,5}), (\overrightarrow{3,2}), (\overrightarrow{1,3})\}$$

is a cut with capacity 19. The set of directed arcs

$$\{(\overrightarrow{4,6}), (\overrightarrow{3,6}), (\overrightarrow{3,5})\}$$

is not a cut because the path

$$1 \to 2 \to 5 \to 6$$

contains no directed arc from the set.

For a given path, the flow along this path cannot exceed the smallest of the capacities of its arcs. Hence, for any cut the flow along this path cannot exceed the capacity of whichever of its arcs belongs to the cut. Now a maximal flow consists of a sum of flows along various paths from the source to the sink. Therefore, for any cut a maximal flow cannot exceed

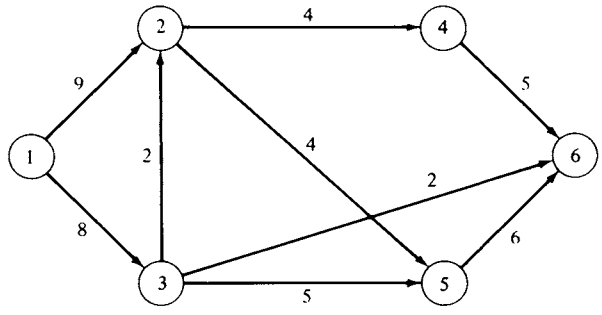


FIGURE 5.22

the sum of the capacities of the arcs of the various paths that belong to the cut. Hence, a maximal flow cannot exceed the capacity of the cut. △

Suppose that we are now in case (i) of the labeling algorithm: the sink has *not* been labeled and no other nodes can be labeled. The set N of all nodes of the network can be partitioned into two disjoint subsets: N_L, the labeled nodes, and N_U, the unlabeled nodes.

Let A be the set of directed arcs that join nodes in N_L to nodes in N_U. We first show that A is a cut in the network. Suppose it is not a cut, so that there is a path from source to sink that does not contain any directed arc from A. It then follows that all the nodes in this path belong either to N_L or to N_U (why?). Since by definition the source is labeled and the sink is unlabeled, we have a contradiction to our assumption that A is not a cut. Therefore, it must be one.

We next show that the capacity of cut A is equal to the maximal flow in the network. From Equation (2a) and the definition of the excess capacities, we can write

$$\sum_{j=1}^{n} (c_{ij} - d_{ij}) = 0, \quad i = 2, \ldots, n-1. \tag{5}$$

When $i = 1$, we obtain from the definition of excess capacity

$$\sum_{j=2}^{n} (c_{1j} - d_{1j}) = \sum_{j=2}^{n} x_{1j} \tag{6}$$

since $x_{j1} = 0$ for $j = 1, 2, \ldots, n$ (why?). Thus, the sum in (6) gives the total flow. We now combine (5) and (6) and obtain

$$\sum_{i \in N_L} \sum_{j=1}^{n} (c_{ij} - d_{ij}) = \sum_{j=2}^{n} x_{1j} = \text{total flow} \tag{7}$$

since the source, node 1, belongs to N_L. Consider the left side of (7). If i and j both belong to N_L, then $c_{ij} - d_{ij}$ and $c_{ji} - d_{ji}$ both occur on the left side of (7) and cancel each other out. Thus, only the terms for which j belongs to N_U are left in the sum. If j is in N_U, then $d_{ij} = 0$. Hence, the left side of (7) becomes

$$\sum_{i \in N_L} \sum_{j \in N_U} c_{ij}$$

and thus (7) can be written as

$$\sum_{i \in N_L} \sum_{j \in N_U} c_{ij} = \text{total flow}. \tag{8}$$

Now the left side of (8) is the capacity of the cut A, since A consisted of exactly those arcs joining nodes in N_L to nodes in N_U. Thus, for this particular cut, the total flow is exactly equal to its capacity.

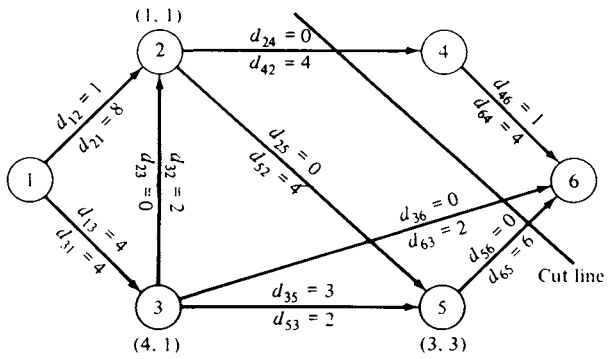


FIGURE 5.23

Since we have already shown that the maximal flow of a network cannot exceed the capacity of any cut, we conclude that for the cut A, just constructed, the flow is maximal. Thus, the labeling algorithm described earlier does yield a maximal flow. Moreover, similar reasoning shows that cut A has the minimum capacity among all cuts. The results established here can also be stated as the max flow–min cut theorem.

THEOREM 5.5 (MAX FLOW–MIN CUT THEOREM). *The maximum flow in a network is the minimum of the capacities of all cuts in the network.* △

EXAMPLE 2. The cut A defined by the network in Figure 5.20 is $\{(\overline{2,4}),(\overline{3,6}),(\overline{5,6})\}$. It takes the name *cut* from the graphical representation in Figure 5.23. The cut line is drawn through precisely those arcs that belong to cut A. △

When using the labeling algorithm it is possible that an augmented path will contain one or more arcs that must be traversed backward from their direction in the original network. The following example illustrates this situation.

EXAMPLE 3. Consider the network in Figure 5.24. We wish to find the maximal flow. We apply the Labeling Algorithm and obtain the labels shown in Figure 5.25.

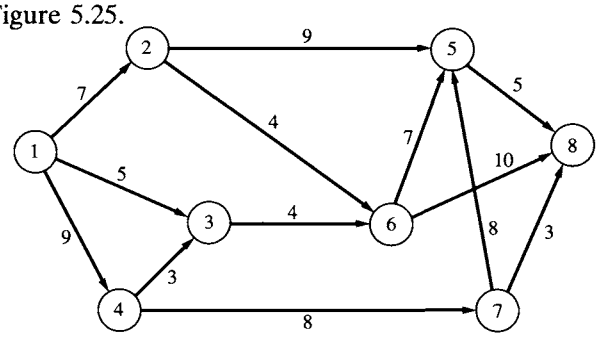


FIGURE 5.24

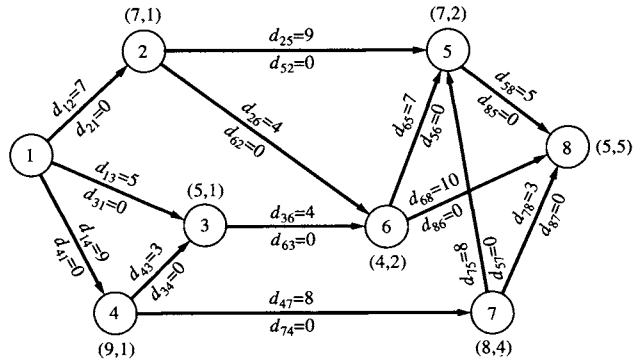


FIGURE 5.25

Since the sink has been labeled, we can increase the flow by 5 units from its initial value of 0. We move backward along the path

$$1 \rightarrow 2 \rightarrow 5 \rightarrow 8$$

and calculate the new excess capacities. These are shown in Figure 5.26 along with the new labels on the nodes.

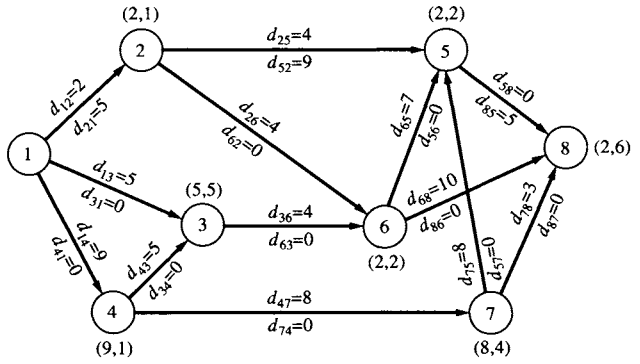


FIGURE 5.26

Again the sink has been labeled, and we can increase the flow by 2 units to a total of 7 units. We move backward along the path

$$1 \rightarrow 2 \rightarrow 6 \rightarrow 8$$

and calculate the new excess capacities. These are shown in Figure 5.27 along with the new labels on the nodes.

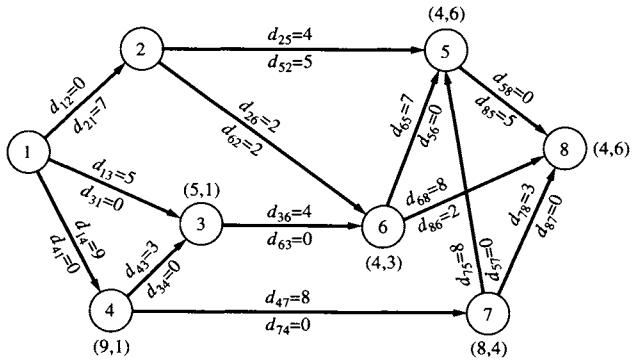


FIGURE 5.27

Again the sink has been labeled, and we can increase the flow by an additional 4 units to a total of 11 units. We move backward along the path

$$1 \rightarrow 3 \rightarrow 6 \rightarrow 8$$

and calculate the new excess capacities. These are shown in Figure 5.28 along with the new labels on the nodes.

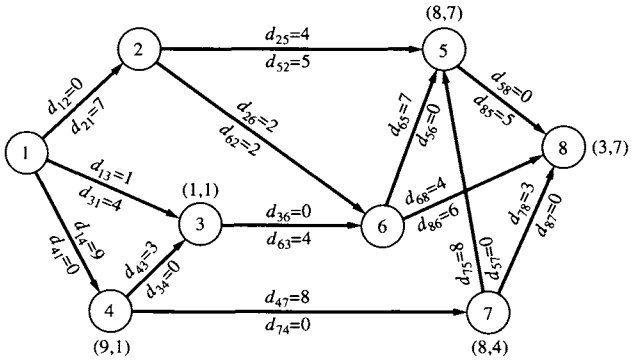


FIGURE 5.28

Also in this iteration of the Labeling Algorithm, the sink has been labeled. The augmenting path is

$$1 \rightarrow 4 \rightarrow 7 \rightarrow 8,$$

and we can increase the flow by 3 units to a total of 14 units. We calculate the new excess capacities as shown in Figure 5.29, along with the new labels on the nodes.

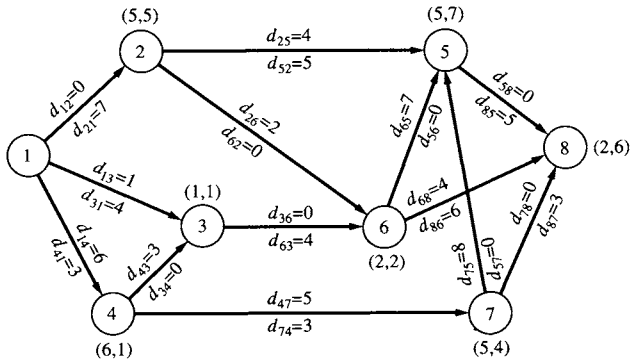


FIGURE 5.29

Again the sink has been labeled, and we can increase the flow by 2 units to a total of 16 units. The augmenting path is

$$1 \rightarrow 4 \rightarrow 7 \rightarrow 5 \rightarrow 2 \rightarrow 6 \rightarrow 8.$$

Note that the arc $5 \rightarrow 2$ in this path is the reverse of the arc $2 \rightarrow 5$ included in the original network. Thus, we must be careful with the indices in the excess capacity calculations. We have

$$\begin{aligned} d'_{14} &= d_{14} - e_8 & d'_{41} &= d_{41} + e_8 \\ d'_{47} &= d_{47} - e_8 & d'_{74} &= d_{74} + e_8 \\ d'_{75} &= d_{75} - e_8 & d'_{57} &= d_{57} + e_8 \\ d'_{52} &= d_{52} - e_8 & d'_{25} &= d_{25} + e_8. \end{aligned}$$

Note that in calculating d'_{25} we have to *increase* the excess capacity of a *forward* arc.

$$\begin{aligned} d'_{26} &= d_{26} - e_8 & d'_{62} &= d_{62} + e_8 \\ d'_{68} &= d_{68} - e_8 & d'_{86} &= d_{86} + e_8 \end{aligned}$$

The new capacities and new labels computed from them are shown in Figure 5.30.

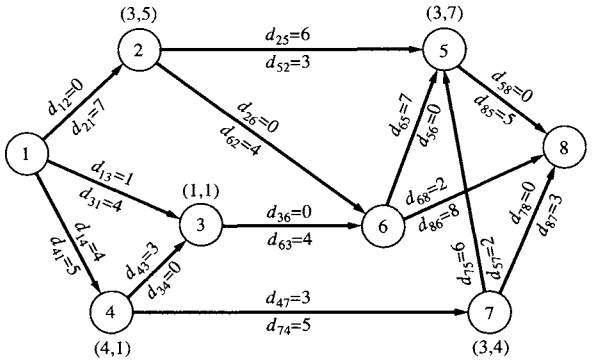


FIGURE 5.30

In this case the sink is not labeled and no other nodes can be labeled. We have found a maximal flow, as shown in Figure 5.31.

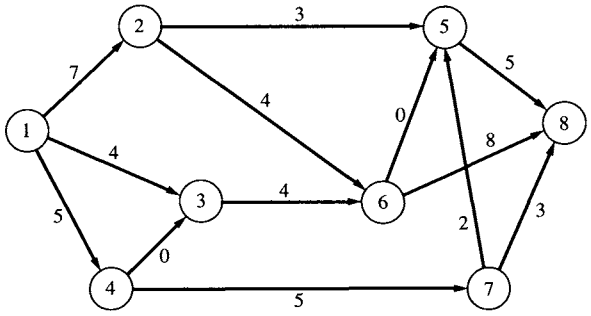


FIGURE 5.31

The minimum cut for this network can be computed by first partitioning the set of nodes $N = \{1, 2, \ldots, 8\}$ into the subsets N_L and N_U of labeled and unlabeled nodes, respectively, using the labels obtained in the last iteration of the Labeling Algorithm. We have $N_L = \{1, 2, 3, 4, 5, 7\}$ and $N_U = \{6, 8\}$ Cut A is the set of directed arcs in the network leading from nodes in N_L to nodes in N_U. We have

$$A = \{\overline{(2,6)}, \overline{(3,6)}, \overline{(5,8)}, \overline{(7,8)}\}.$$

The arcs in the cut set are shown in Figure 5.32 with gaps. Observe that the capacity of this cut is

$$4 + 4 + 5 + 3 = 16 = \text{maximal flow}.$$

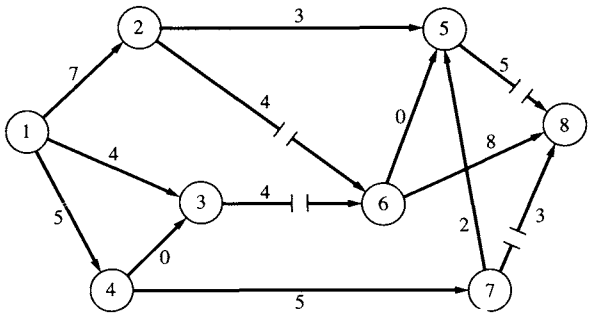


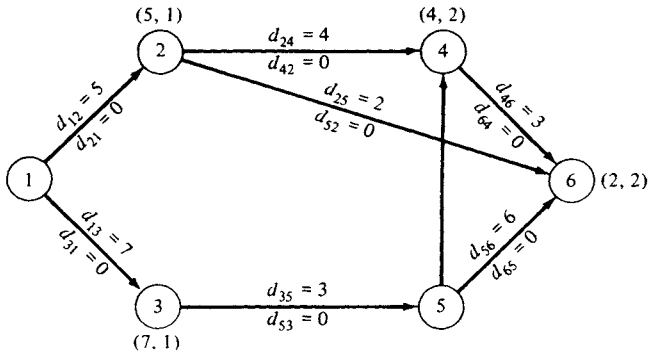
FIGURE 5.32

△

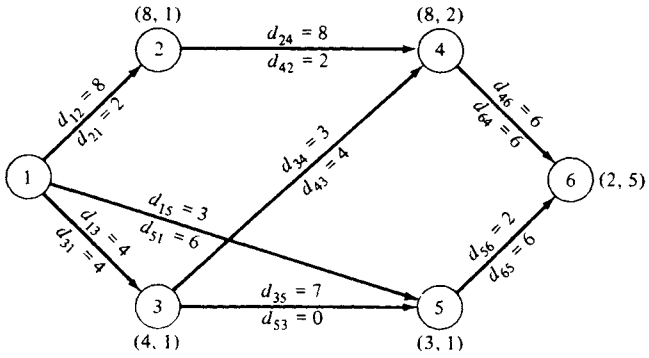
5.4 EXERCISES

In Exercises 1 and 2, a network that has been labeled with the labeling algorithm is given. In each case the sink has been labeled. Following the appropriate steps of the labeling algorithm, adjust the excess capacities and relabel the network.

1.

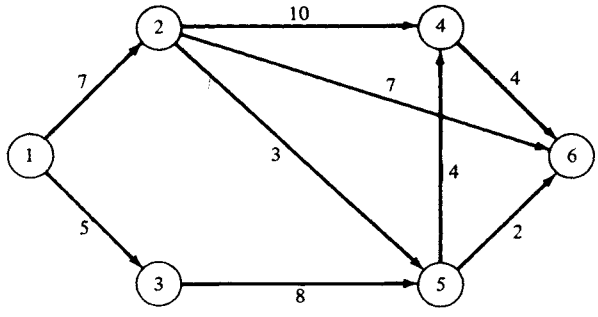


2.

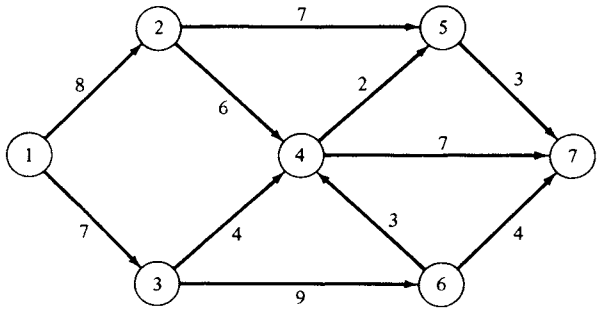


In Exercises 3–6, find the maximal flow in the given network using the labeling algorithm.

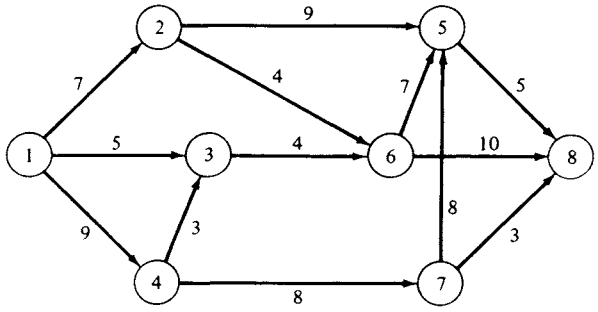
3.



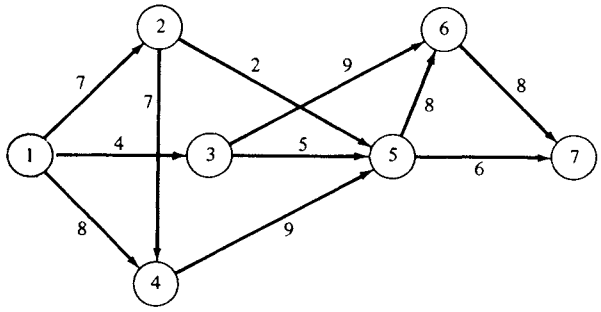
4.



5.



6.



7. (a) Model the following situation as a maximal flow problem. There are an equal number, say n, of boys and girls at a school dance. It is also known for each boy–girl pair whether they are friends (this friendship relation could be given by an $n \times n$ matrix). Find the maximum number of friendly pairs that can be formed.

(b) Solve the above problem for the friendship relation matrix

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

(c) Solve the problem in part (a) for the friendship relation matrix

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

(d) Find a condition on the friendship relation matrix that will guarantee that each boy and girl is dancing with a friend.

8. (a) Model the following situation as a maximal flow problem. Each day at the CPR Clothing Factory there are a large number of tasks to be assigned to the various machines. Assume that any task when assigned to a machine will occupy the machine for the entire day. Of course, only some tasks can be assigned to each machine—some machines are cutting machines; others are pleaters; and others can be set for basting, straight stitching, or top stitching. For a particular day suppose the list of tasks and possible machines is as given in the accompanying table.

	Machine					
Task	1	2	3	4	5	6
1		√				
2	√					
3	√	√				
4	√	√				
5			√	√		
6				√	√	
7				√		√
8			√		√	√
9					√	
10			√			√

(b) How many tasks can be completed?

9. Show that the flow out of the source of a network is equal to the flow into the sink.

5.4 PROJECT

There are many situations in which one is concerned about flow along one route. If there are no delays, the maximum flow is easily determined as the time available divided by time per unit flow. The interesting cases occur when there are delays at certain points and limited waiting facilities at these delay points. Examples of such a situation are trains moving on a single track with sidings at certain points and a production line with certain slowly executed operations that cause bottlenecks. Of course, a railroad siding can hold only a certain number of trains. Likewise, we assume that the holding areas on the production line can contain only a limited number of items.

These situations can be modeled as maximal flow problems. To construct the model, we must determine the network and the meanings of its nodes, arcs, and capacities. As a first step, let P be the duration of the period in which we want to maximize the flow. Divide P into k equal units. Then we may examine the state of the system at any time $0, 1, \ldots, k$. For example, a train schedule would probably have $P = 24$ hr and $k = 240$, meaning that we know the status of each train every 0.1 hr.

Assume there are r delay points. Let n_i $(i = 1, 2, \ldots, r)$ be the capacity of the ith delay point. Let t_0 be the time needed to go from the beginning of the route to the first delay point. Let t_i $(i = 1, 2, \ldots, r-1)$ be the time needed to go from the ith delay point to the $(i+1)$st delay point. Finally, let t_r be the time needed to go from the rth delay point to the end of the route.

(a) Assuming there are no delays, what is the maximal flow in the route during a period of length P?

(b) What is the earliest arrival time at the end of the route, assuming that a unit of flow starts at the beginning of the time period?

(c) What is the latest departure time from the beginning of the route that will allow a unit of flow to arrive at the end of the route before the time period is up?

To construct the maximal flow model we will need a network that has a different set of nodes for each of the $k + 1$ specified times in the period under consideration. We will also need a decision function that governs whether a unit of flow can move from one delay point to the next. In the case of the train example, this function will be based in part on information about express train schedules and the need for a local train to be able to move from one siding to the next without being overtaken by an express. Let this function be

$$\delta_{ij} = \begin{cases} 1 & \text{if at time } j \text{ the unit of flow can safely move} \\ & \text{from delay point } i \text{ to delay point } i+1 \\ 0 & \text{otherwise.} \end{cases}$$

(d) Formulate the maximal flow model of this situation.

(e) Assume that

$$k = 13, \quad r = 3$$
$$t_0 = 2, \quad t_1 = 3, \quad t_2 = 1, \quad t_3 = 2$$
$$n_1 = 2, \quad n_2 = 1, \quad n_3 = 3$$

$$\delta = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{bmatrix}.$$

Find the maximal flow along this route for the given period of time.

Further Reading

Chvátal, Vašek. *Linear Programming*. Freeman, New York, 1980.

Dinic, E. A. "Algorithm for Solution of a Problem of Maximum Flow in Networks with Power Estimation." *Soviet Math. Doklady.* **11** (1970), 1277–1280.

Ford, L. R., Jr., and Fulkerson, D. R. *Flows in Networks*. Princeton Univ. Press, Princeton, NJ, 1962.

Goldberg, Andrew V., and Tarjan, Robert E. "A New Approach to the Maximum-Flow Problem." *J. Assoc. Comput. Mach.* **35** (1988), 921–940.

Papadimitriou, Christos H., and Steiglitz, Kenneth. *Combinatorial Optimization: Algorithms and Complexity*. Prentice-Hall, Englewood Cliffs, NJ, 1982.

5.5 THE SHORTEST ROUTE PROBLEM

In many applied problems it is necessary to find the shortest path from a given node in a network to another node in the network. These include problems in the optimal distribution of goods (along highways or railroads), routing of communication signals, and transporting people in metropolitan areas. Another very interesting application of the **shortest route problem** is the problem of optimally replacing equipment that deteriorates with age. A number of these applications will be considered in the exercises.

Many algorithms have been developed for the solution of the shortest route problem. The algorithm that we now present is *not* the most efficient one; however, it does have the advantage of being one of the simplest to describe and understand. Other algorithms are described in materials listed under Further Reading.

Once we have designated our origin, the algorithm sweeps out to find the shortest route to each node in the network; in particular, it gives the shortest route from the origin to the designated destination node. The nodes are examined in order of distance from the origin.

The set of nodes of the network is divided into two disjoint subsets, N_r and N_u, as follows:

N_r consists of the origin and all nodes that have been *reached* from the origin by a shortest route.

N_u consists of all nodes that have not yet been reached from the origin by a shortest route.

Initially, N_r contains only the origin; we transfer nodes from N_u to N_r in a systematic fashion to be described below. The algorithm terminates when

N_u is empty. Of course, from a practical point of view, the algorithm can be stopped as soon as the destination node is in N_r.

To start the algorithm we find the node closest to the origin and place this node in N_r. In case of ties we place all nodes that are closest to the origin in N_r. This completes the first stage of the algorithm.

At all subsequent stages of the algorithm we will need to find the node (or nodes) in N_u that is (are) closest to the origin and move it (them) to N_r. We now describe how to find such a node. Let j be a node in N_u and assume that j is at least as close to the origin as any other node in N_u. We claim that j is connected to a node in N_r by a directed arc (a path that does not go through any other nodes). Certainly j is connected to a node (the origin) in N_r by a path (why?). If this path contains a node in N_u other than j, then that node is closer to the origin, contradicting the choice of j. Thus, in this path the arc from j must go to a node in N_r. Hence, the candidates for node j are all those nodes in N_u that are connected to some node in N_r by a directed arc.

Thus, we select node j in N_u as follows:

For each node i in N_r (including the origin) that is connected to a node k in N_u by a directed arc, form $s' = s + d$, where s is the length of the shortest route from the origin to i (we keep track of this number for each node we add to N_r) and d is the length of the directed arc from i to k.

Select j as the node for which s' is minimal, put j in N_r, and record its distance to the origin. In case of ties, choose all the tying nodes and place them in N_r. This completes the algorithm. This second step of selecting node j is repeated until the destination node is placed in N_r.

We shall illustrate the algorithm and describe a way of keeping track of the necessary information by using the network in Figure 5.33. Its origin is node 1, and its destination is node 8. The number on each arc represents the length of the arc rather than the capacity as for a flow problem.

Starting with the origin, we list each node across the top of the page. Under each node we list all the nodes that are connected to this node by a

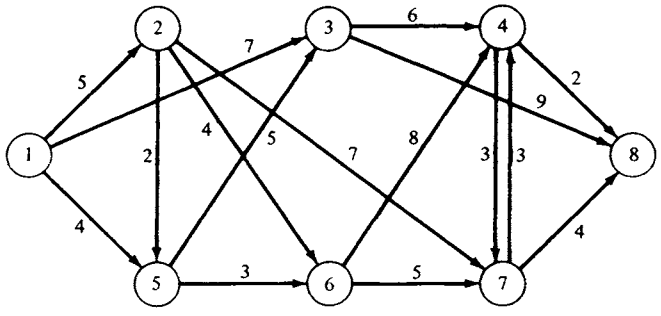


FIGURE 5.33

directed arc that *leaves* this node. Next to each node in the list we indicate the length of the directed arc that joins the node at the top to that node in the list. For example, the list shows that node 3 is connected to nodes 4 and 8 by directed arcs of lengths 6 and 9, respectively (Table 5.2).

TABLE 5.2

1	2	3	4	5	6	7	8
5–4	5–2	4–6	8–2	6–3	7–5	4–3	
2–5	6–4	8–9	7–3	3–5	4–8	8–4	
3–7	7–7						

Initially, $N_r = \{1\}$. We now find the node closest to the origin. The candidates are nodes 5, 2, and 3. The closest one is node 5. In the list, we circle the entry 5–4 under node 1 and mark the distance 4 next to node 5, indicating that the distance from the origin to node 5 is 4. We also cross out all node 5's in every column, since we have now found the shortest route to node 5, and the other directed arcs leading to node 5 will not be used. At this point, $N_r = \{1, 5\}$, and our modified list is shown in Table 5.3.

TABLE 5.3

1	2	3	4	5–4	6	7	8
(5–4)	~~5–2~~	4–6	8–2	6–3	7–5	4–3	
2–5	6–4	8–9	7–3	3–5	4–8	8–4	
3–7	7–7						

The nodes in N_u that are connected to nodes in N_r by directed arcs can be found by reading the uncircled and not-crossed-off entries in the columns labeled by nodes in N_r. Thus, nodes 2, 3, and 6 are candidates for the next node to be added to N_r. The distance from the origin to node 6 via node 5 can be obtained by adding the number, 4, next to node 5 (the distance from the origin to node 5) to the length of the directed arc $(\overline{5, 6})$, which is 3. The distances from the origin to nodes 2 and 3 come from the table. Since the smallest of these distances is 5, we choose node 2 to be put in N_r. We circle 2–5 under 1 in the list, make the distance 5 next to node 2, indicating that the shortest distance to node 2 is 5, and cross out all arcs leading *into* node 2 (there are none). At this point, $N_r = \{1, 2, 5\}$, and our modified list is shown in Table 5.4.

TABLE 5.4

1	2–5	3	4	5–4	6	7	8
(5–4)	~~5–2~~	4–6	8–2	6–3	7–5	4–3	
(2–5)	6–4	8–9	7–3	3–5	4–8	8–4	
3–7	7–7						

Using the technique described before, we find that nodes 3, 6, and 7 are the next candidates for the node to be put in N_r. The length of the routes associated with these nodes are given in Table 5.5.

TABLE 5.5

Route	*Length*
$1 \rightarrow 3$	7
$1 \rightarrow 2 \rightarrow 6$	$5 + 4 = 9$
$1 \rightarrow 2 \rightarrow 7$	$5 + 7 = 12$
$1 \rightarrow 5 \rightarrow 6$	$4 + 3 = 7$
$1 \rightarrow 5 \rightarrow 3$	$4 + 5 = 9$

We select both nodes 3 and 6 to be reached along routes with length 7. We circle 3–7 under node 1 and 6–3 under node 5, mark the distance 7 next to nodes 3 and 6 at the top of the columns, and cross out all node 3's and node 6's in every column. At this point, $N_r = \{1, 2, 3, 5, 6\}$, and our modified list is shown in Table 5.6. Since the first and fifth columns have no other available directed arcs, we can ignore them, so we place a check mark over these columns.

TABLE 5.6

√ 1	2–5	3–7	4	√ 5–4	6–7	7	8
(5–4)	~~5–2~~	4–6	8–2	(6–3)	7–5	4–3	
(2–5)	~~6–4~~	8–9	7–3	~~3–5~~	4–8	8–4	
(3–7)	7–7						

The next candidates for inclusion in N_r are nodes 7, 4, and 8. The lengths of the routes associated with these nodes are given in Table 5.7.

TABLE 5.7

Route	*Length*
$1 \rightarrow 2 \rightarrow 7$	$5 + 7 = 12$
$1 \rightarrow 3 \rightarrow 4$	$7 + 6 = 13$
$1 \rightarrow 3 \rightarrow 8$	$7 + 9 = 16$
$1 \rightarrow 5 \rightarrow 6 \rightarrow 7$	$7 + 5 = 12$
$1 \rightarrow 5 \rightarrow 6 \rightarrow 4$	$7 + 8 = 15$

We must choose node 7, mark its distance as 12 (along two different paths), and circle 7–7 in column 2 and 7–5 in column 6. We also cross out all other occurrences of node 7 in any of the columns. Our list is now as shown in Table 5.8.

TABLE 5.8

√ 1	√ 2–5	3–7	4	√ 5–4	6–7	7–12	8
(5–4)	~~5–2~~	4–6	8–2	(6–3)	(7–5)	4–3	
(2–5)	~~5–4~~	8–9	~~7–8~~	~~3–5~~	4–8	8–4	
(3–7)	(7–7)						

After the next stage the list becomes as shown in Table 5.9 (verify).

TABLE 5.9

√ 1	√ 2–5	3–7	4–13	√ 5–4	√ 6–7	7–12	8
(5–4)	~~5–2~~	(4–6)	8–2	(6–3)	(7–5)	~~4–8~~	
(2–5)	~~5–4~~	8–9	~~7–8~~	~~3–5~~	~~4–8~~	8–4	
(3–7)	(7–7)						

Verify that the final list is Table 5.10.

TABLE 5.10

√ 1	√ 2–5	√ 3–7	√ 4–13	√ 5–4	√ 6–7	√ 7–12	8–15
(5–4)	~~2–5~~	(4–6)	(8–2)	(6–3)	(7–5)	~~8–15~~	
(2–5)	~~4–6~~	~~6–3~~	~~2–5~~	~~3–7~~	~~4–6~~	~~6–3~~	
(3–7)	(7–7)						

From this list we see that the shortest route to the destination (node 8) has length 15. The route can be found by looking in the circled numbers and working back from node 8. Among the circles find 8; it occurs in column 4. Thus, the last arc in the route is $4 \rightarrow 8$. Now find a 4 among the circles; it is in column 3. Thus, the route goes $3 \rightarrow 4 \rightarrow 8$. After another step, we find the shortest route is

$$1 \rightarrow 3 \rightarrow 4 \rightarrow 8.$$

The list also tells us that to get to node 6, for example, the shortest route has length 7. It is

$$1 \rightarrow 5 \rightarrow 6.$$

Equipment Replacement Problem

Another situation that can be modeled as a shortest route problem is the question of when equipment that deteriorates with age should be replaced. Over a period of several years we would expect the price of the equipment to increase gradually and the cost of maintenance for the equipment to increase rapidly as it ages. Certainly this situation is familiar to every car owner.

Consider a factory that must occasionally replace a piece of equipment that deteriorates with age. Assume that they are using a planning horizon of 5 years. At the beginning of Year 1 they will purchase a new piece of equipment that will be replaced after every j years. Their problem is to determine j so that the combined maintenance costs and purchase prices will be a minimum. The purchase prices and maintenance costs are given in Table 5.11.

TABLE 5.11

	Purchase price				
Beginning of year	1	2	3	4	5
Price ($10,000)	17	19	21	25	30
	Maintenance costs				
Age (years)	0–1	1–2	2–3	3–4	4–5
Cost ($10,000)	3.8	5.0	9.7	18.2	30.4

We now construct a network that will model the equipment replacement problem. The shortest path in the network will represent the optimal replacement scheme. Each node of the network will represent the beginning of a year in the planning horizon. We must also include a node to represent the end of the last year in the planning horizon. Each node is connected to *every* subsequent node by an arc. The arcs represent replacement strategies. For example, the arc connecting node 1 to node 4 represents the strategy of purchasing a new machine at the beginning of Year 1 and then replacing it at the beginning of Year 4. The length of each arc represents the total cost (purchase price plus maintenance costs) of the corresponding replacement strategy. For example, the length of arc $(\overline{1,4})$ is $17 + (3.8 + 5.0 + 9.7) = 20.5$. Notice that this arc represents owning a piece of equipment for 3 years, so that there is a maintenance cost for each of the years. All the arcs for this network and the computations for their lengths are given in Table 5.12.

TABLE 5.12

$(\overline{1,2})$	$17 + 3.8$	$= 20.8$
$(\overline{1,3})$	$17 + (3.8 + 5.0)$	$= 25.8$
$(\overline{1,4})$	$17 + (3.8 + 5.0 + 9.7)$	$= 35.5$
$(\overline{1,5})$	$17 + (3.8 + 5.0 + 9.7 + 18.2)$	$= 53.7$
$(\overline{1,6})$	$17 + (3.8 + 5.0 + 9.7 + 18.2 + 30.4)$	$= 84.1$
$(\overline{2,3})$	$19 + 3.8$	$= 22.8$
$(\overline{2,4})$	$19 + (3.8 + 5.0)$	$= 27.8$
$(\overline{2,5})$	$19 + (3.8 + 5.0 + 9.7)$	$= 37.5$
$(\overline{2,6})$	$19 + (3.8 + 5.0 + 9.7 + 18.2)$	$= 55.7$
$(\overline{3,4})$	$21 + 3.8$	$= 24.8$
$(\overline{3,5})$	$21 + (3.8 + 5.0)$	$= 29.8$
$(\overline{3,6})$	$21 + (3.8 + 5.0 + 9.7)$	$= 39.5$
$(\overline{4,5})$	$25 + 3.8$	$= 28.8$
$(\overline{4,6})$	$25 + (3.8 + 5.0)$	$= 33.8$
$(\overline{5,6})$	$30 + 3.8$	$= 33.8$

The network that models this equipment replacement problem is shown in Figure 5.34.

Any route through the network will represent a 5-year plan for equipment replacement. For example, the route $1 \rightarrow 3 \rightarrow 4 \rightarrow 6$ represents purchasing a new piece of equipment at the beginning of Year 1, of Year 3, and of Year 4. The cost of this strategy is $25.8 + 24.8 + 33.8 = 84.4$, or \$844,000. The reader may use the shortest route algorithm to show that the optimal replacement strategy is to purchase a new piece of equipment at the beginning of Years 1 and 3. The cost of this strategy is \$653,000.

In general, if the planning horizon encompasses n time units, the network will have $n + 1$ nodes. The length of the arc joining node i to

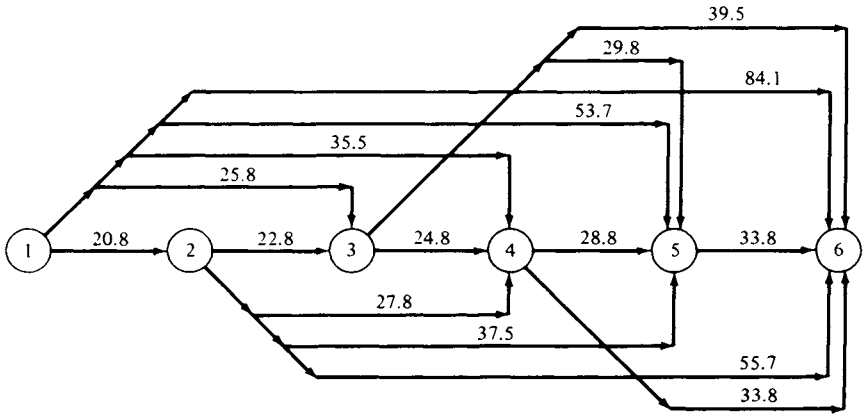


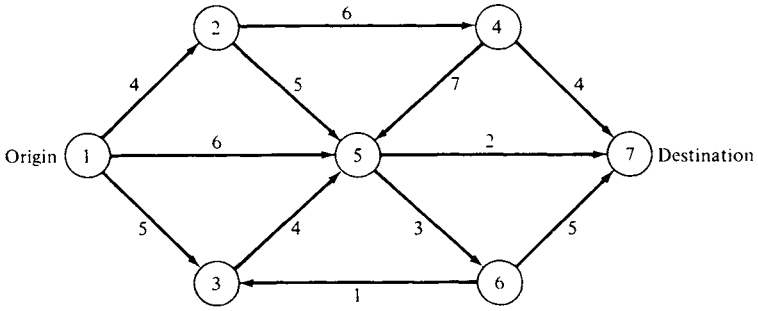
FIGURE 5.34

node j $(i < j)$ will be the sum of the purchase price at the beginning of year i and the maintenance costs for the first $j - i$ time periods.

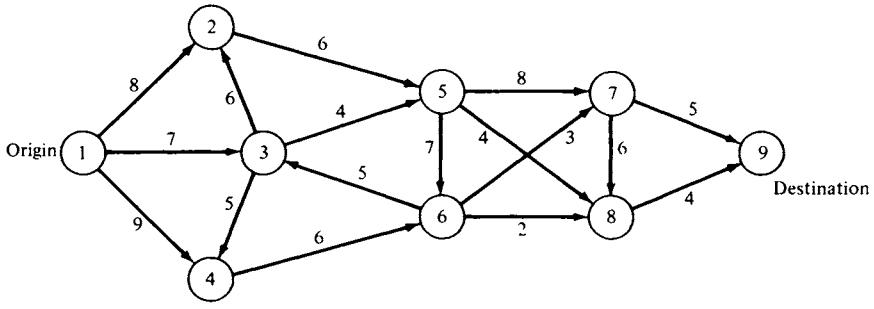
5.5 EXERCISES

In Exercises 1–4 find the shortest path between the indicated origin and the indicated destination.

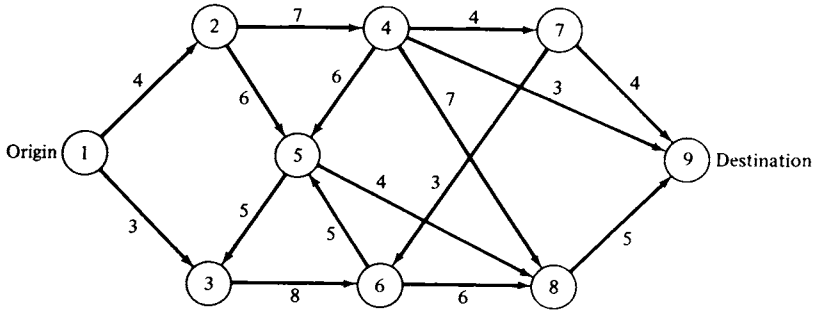
1.



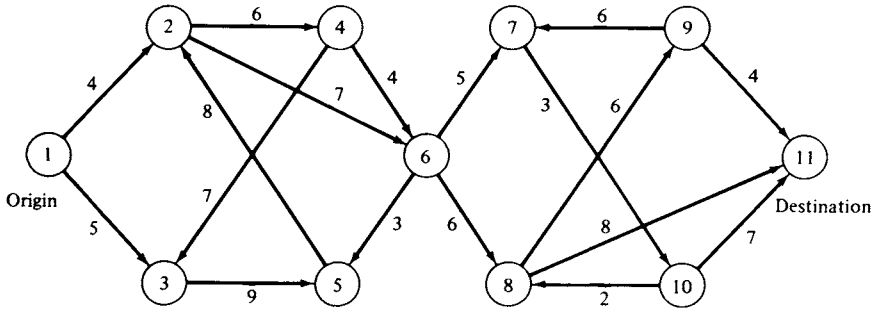
2.



3.



4.



In Exercises 5 and 6 we give tables of purchase prices and maintenance costs for pieces of equipment. Develop a replacement policy for the equipment.

5.

	Year					
	1	2	3	4	5	6
Purchase price ($10,000)	5	6	8	9	11	12
Age (years)	0–1	1–2	2–3	3–4	4–5	5–6
Maintenance cost ($10,000)	3.0	3.5	5.5	8.5	12.0	18.0

6.

	Year									
	1	2	3	4	5	6	7	8	9	10
Purchase price ($10,000)	3.0	3.5	4.1	4.9	6.0	6.5	6.7	6.7	7.5	8.0
Age (years)	0–1	1–2	2–3	3–4	4–5	5–6	6–7	7–8	8–9	9–10
Maintenance cost ($10,000)	1.0	1.5	2.1	2.8	4.0	7.0	20.0	7.0	8.0	9.5

5.5 PROJECTS

1. Charlie Anderson lives in the town of Hatboro in eastern Montgomery County. Frequently he travels to northeast Philadelphia to visit his parents. The network of roads connecting the two points is shown in Figure 5.35. Node 1 is Charlie's home and node 45 is his destination. This network is a combination of rural roads, urban streets, and expressways. The rural roads are in the area from Hatboro to the outskirts of Philadelphia (nodes 1 through 20). These roads have long stretches without intersections, few traffic signals and stop signs, and generally light traffic. The urban streets begin near the city limits of Philadelphia (nodes 21 through 45). They are generally more congested and have many traffic signals. The expressway joins nodes 21, 44, and 45.

   Charlie has computed average times for traveling over each arc of the network. These times along with the length of each arc are given in Table 5.13.

   (a) Compute the route Charlie should use to travel the shortest distance when going from Hatboro to northeast Philadelphia.

   (b) Compute the route Charlie should use to get to northeast Philadelphia most quickly.

   (c) Compare your answers in parts (a) and (b) with the route $1 \rightarrow 3 \rightarrow 4 \rightarrow 6 \rightarrow 10 \rightarrow 17 \rightarrow 19 \rightarrow 23 \rightarrow 29 \rightarrow 34 \rightarrow 38 \rightarrow 41 \rightarrow 42 \rightarrow 43 \rightarrow 45$. Discuss

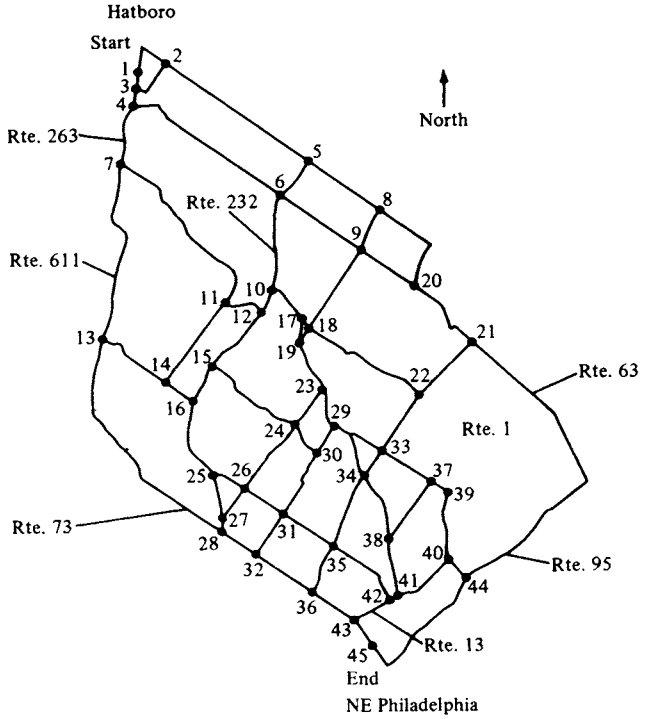


FIGURE 5.35

TABLE 5.13 Distances and Times between Nodes

Nodes	*Distance (miles)*	*Time (sec)*	*Nodes*	*Distance (miles)*	*Time (sec)*
1–2	1.5	380	22–33	1.2	220
1–3	0.7	190	23–24	0.8	120
2–5	3.1	360	23–29	0.8	140
3–2	1.4	210	24–26	1.6	220
3–4	0.7	200	24–30	0.6	175
4–6	3.6	380	25–26	0.6	170
4–7	1.1	230	25–27	0.8	170
5–6	0.9	120	26–27	0.6	160
5–8	1.6	140	26–31	0.8	190
6–9	1.8	225	27–28	0.3	135
6–10	1.8	220	28–32	0.7	180
7–11	3.5	450	29–30	0.6	120
7–13	3.1	420	29–33	0.8	150
8–9	0.9	150	29–34	0.6	180
8–20	2.0	275	30–31	1.4	225
9–18	1.7	255	31–32	0.9	180
9–20	1.2	230	31–35	1.3	320
10–12	0.4	85	32–36	1.3	320
10–17	0.8	170	33–34	0.5	100
11–12	0.8	110	33–37	1.0	140
11–14	1.8	235	34–35	1.4	240
12–15	1.5	220	34–38	1.3	220
13–14	1.5	280	35–36	1.3	230
13–28	4.8	750	35–42	1.5	240
14–16	0.6	110	36–43	0.9	190
15–16	0.8	140	37–38	1.4	210
15–24	2.2	340	37–39	0.4	80
16–25	1.6	230	38–41	1.0	140
17–18	0.2	60	39–40	1.3	240
17–19	0.4	95	40–41	1.2	305
18–19	0.3	80	40–44	0.5	55
18–22	2.6	400	41–42	0.2	70
19–23	1.6	160	42–43	0.7	240
20–21	1.3	260	43–45	0.5	100
21–22	1.6	250	44–45	3.3	520
21–44	6.5	480			

what penalties Charlie incurs by not using either the minimum distance or the minimum time routes.

2. On the basis of published figures for new car prices and costs for maintenance, develop an optimal replacement policy for the time period 10 years ago until the present.

Further Reading

Djiskstra, E. W. "A Note on Two Problems in Connection with Graphs." *Numerische Math.* **1** (1959), 269–271.

Dreyfus, S. E. "An Appraisal of Some Shortest-Path Algorithms," *Perspectives on Optimization: A Collection of Expository Articles* (A. M. Geoffrion, Ed.), pp. 197–238 Addison-Wesley, Reading, MA, 1972.

5.6 THE CRITICAL PATH METHOD

Scheduling models are an important area of study in operations research. We have examined some of these models in the examples and exercises in previous sections. The assignment problem gave one form of a scheduling model, as we showed in Project 1 of Section 5.2. This project discussed an idealized method of scheduling nurses on a floor of a hospital. The traveling salesman problem occurred as a scheduling model for a job shop. In this situation we were minimizing total setup time. The literature has extensive discussions of the airline flight crew problem. Some references to this problem and other scheduling problems are given under Further Reading.

In this section we discuss yet another model for scheduling. This model, the **critical path method**, or **CPM**, was originally conceived by researchers at E. I. duPont de Nemours Company and Remington Rand in a collaborative effort. John W. Mauchly, James E. Kelley, and Morgan Walker had the lead roles in its development. It was tested in early 1958 on scheduling the construction of a $10 million chemical plant in Louisville, Kentucky. Its advantage in this situation was the small effort needed to incorporate design changes into the schedule.

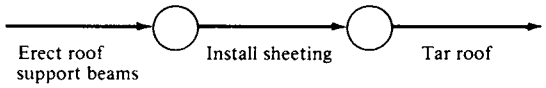
Modifications of the original model have been made, but the basic concepts remain the same. The model is based on a network that represents the activities and completion times of a process. CPM is used in a variety of settings, including larger construction projects, complicated maintenance procedures, installation of computer systems, and production of motion pictures. A related planning process called PERT (Program Evaluation and Review Technique) is widely used for government activities, especially in the Department of Defense. We will limit our discussion to the network model for CPM.

It is typical of construction projects that many unrelated activities can be performed simultaneously with the only constraint being that certain activities must precede others. For example, the schedule for building a house cannot call for tarring the roof before the supporting beams have been erected and covered with sheeting. Aside from these mechanical constraints, the other important constraint is meeting the scheduled completion date. CPM models the schedule of the activities of a project and indicates an order in which the individual activities should be performed.

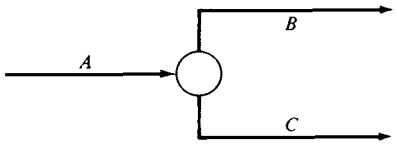
The nodes in the CPM network represent *times*, but not measured in days from the beginning of the project. Rather the times are completions of activities such as *excavation completed* or *plumbing roughed in*. In some

of the literature these times are called **events**. The arcs of the network represent **activities** such as *install floor* or *paint walls*. The source node is the beginning of the project and the sink node is the end of the project, and these can be the only source and sink, respectively, for the network.

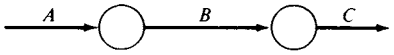
Instead of the usual specification giving a criterion for connecting two nodes with an arc, it is easier for CPM networks to give the dual specification: a criterion for having two arcs join at a node with one arc entering the node and one leaving. Two such arcs representing activities A_i and A_j are joined at a node with A_i entering and A_j leaving if activity A_i must be completed before activity A_j can begin. For example, part of a CPM network might be



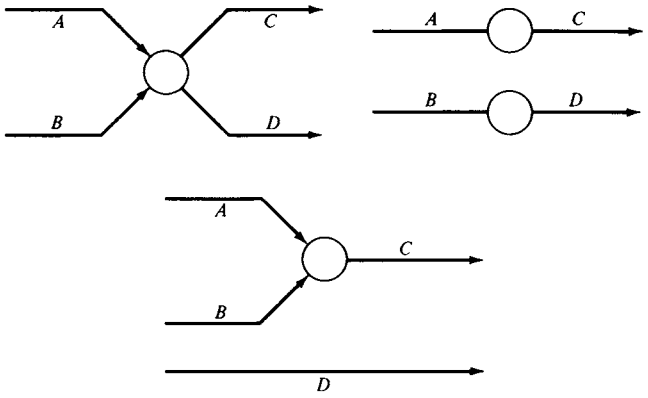
In joining arcs one must be careful not to produce sequences in the network that do not fit with reality. If activities B and C must follow activity A, but are independent, they should be diagrammed as



and not as

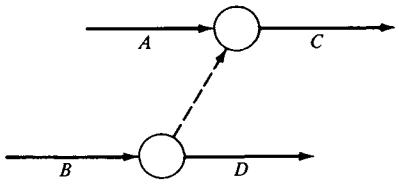


A slightly more complicated situation occurs when there are four activities, A, B, C, and D, where A and B both precede C but only B precedes D. None of the following correctly diagrams this situation (why?).



This dilemma is solved by introducing a **dummy activity** or **logical restraint**. It is an arc of the network, usually shown as a dashed rather than a

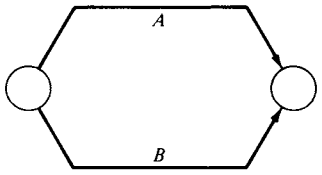
solid line, which represents no work. Using this device we can correctly represent the above situation as



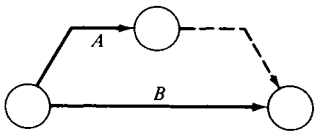
It is also important that no loops be introduced in a CPM network. A loop would mean that an activity must both precede and follow another—a logical impossibility. Loops are easily detected in a small network. In large networks with thousands of nodes, loops may be introduced by errors in the input data. Consequently, a first step in a CPM algorithm is to check the network for loops.

Once a correct precedence network has been established for a problem, the key step in the CPM process must take place. An estimate for the time necessary to complete each activity must be given. The time estimates should be as good as possible. However, after the critical path is found, the planner can reexamine his or her estimates for the path to see if any are unrealistic.

Typically the events of the projects are numbers but not necessarily in any particular order. In this way more events can be added to a network without having to renumber all previous events. The activities are specified by giving the pairs of nodes that are joined by their representing arcs. To ensure that each activity has a unique designation, dummy activities are inserted in case two activities connect the same two nodes. Thus,



becomes



EXAMPLE 1. Consider the several-weekend project faced by a suburban homeowner who wants to replace his small metal storage shed with a larger, more solidly constructed one, building the newer one on the same site as the old shed. Since he has no other place to store tools, he plans to build the new shed around the old one and then to demolish the old shed.

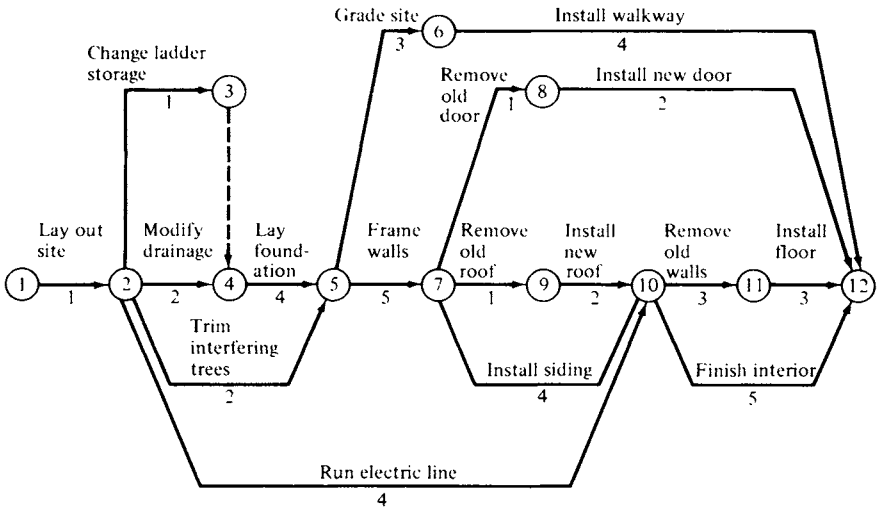


FIGURE 5.36

He has drawn the precedence network shown in Figure 5.36. The number below each activity represents the time (hr) estimate for that activity.

After the network representing the project activities has been determined and checked for its accuracy, the critical path algorithm can be started. The first step of the algorithm is to compute the early event time for each node. The **early event time** $T_E(j)$ for node j represents the soonest that node j can be reached once all activities on all paths leading to the node have been completed. Mathematically it can be computed by modifying the shortest path algorithm to compute the longest path.

In this example, we give the details of computing the early event times for the first six nodes. All the early event times are shown in square boxes in Figure 5.37. For the source node we have $T_E(1) = 0$, representing the beginning of the project. Node 2 has only one arc leading into it, so that $T_E(2) = T_E(1) + 1 = 1$, that is, $T_E(2)$ is the sum of the activity time and the previous event time. Likewise, $T_E(3) = T_E(2) + 1 = 2$.

Since node 4 has two arcs leading into it, we must choose the *longer* of the arcs to compute $T_E(4)$. Remember that $T_E(j)$ represents the earliest that the event can occur after *all* activities leading into it are completed. Thus,

$$\begin{aligned} T_E(4) &= \max\{T_E(2) + 2, T_E(3) + 0\} \\ &= \max\{3, 2\} \\ &= 3. \end{aligned}$$

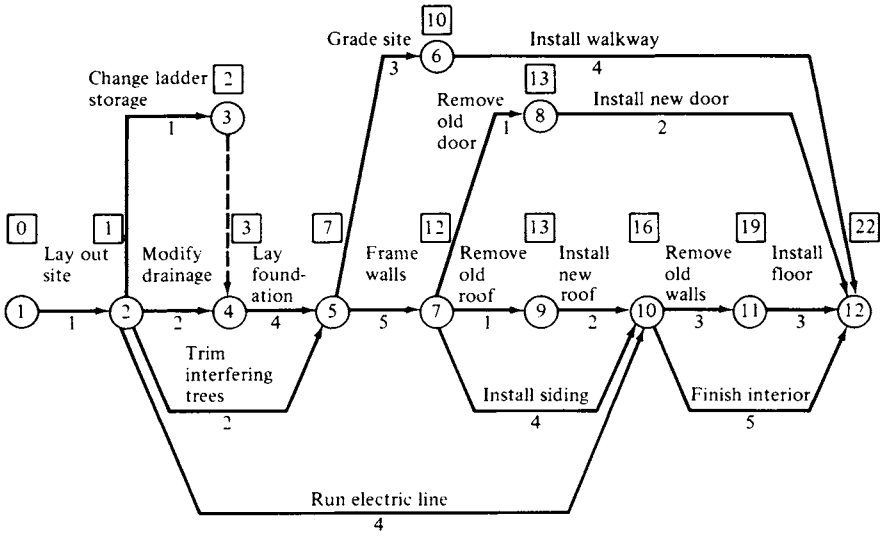


FIGURE 5.37

Likewise, node 5 has two arcs coming into it. Therefore,

$$\begin{aligned} T_E(5) &= \max\{T_E(2) + 2, T_E(4) + 4\} \\ &= \max\{3, 7\} \\ &= 7. \end{aligned}$$

We also find $T_E(6) = T_E(5) + 3 = 10$. Thus, in general,

$$T_E(j) = \max_i \{T_E(i) + \text{length}(\overline{i,j})\},$$

where the maximum is taken over all arcs leading into node j.

The early event time computed for the node that represents project completion (the sink) is the total amount of time necessary to complete the project. If this time is unaceptable, the project manager can at this point investigate ways of speeding up some of the activities. He or she can choose the activities to concentrate on after the critical path (or several of them) is (are) known for the project.

The second step of the algorithm is to compute the late event time for each node. The **late event time** $T_L(j)$ for node j is the latest time by which node j can be reached to still have the project completed in the shortest possible time. Again in this example, we give the details of computing the late event times for nodes 7 through 12. All late event times are shown in diamonds below the early event times in Figure 5.38.

For the sink node we have $T_L(12) = 22$, the total project time. For node 11, since the activity of laying the floor takes 3 units, the start of floor

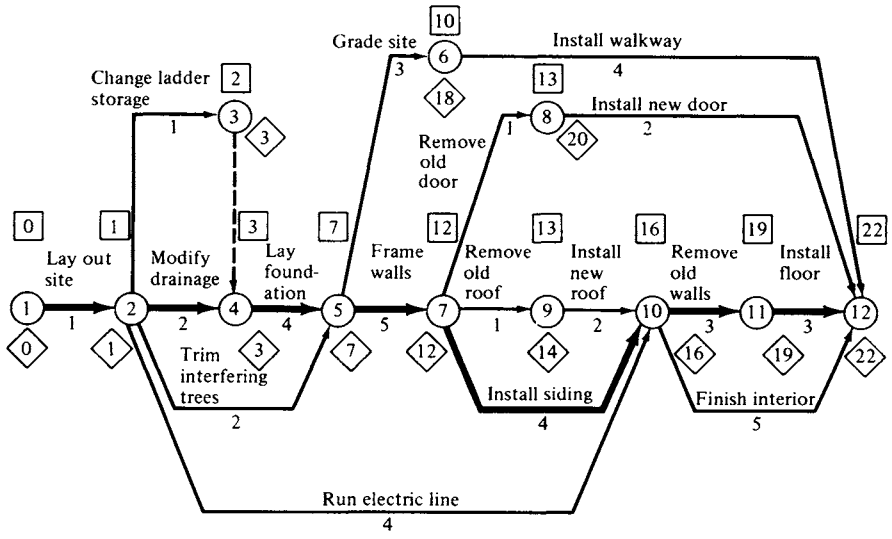


FIGURE 5.38

laying must occur at 19 units. That is,

$$T_L(11) = T_L(12) - 3 = 22 - 3 = 19.$$

There are two paths from node 10 to node 12. We must use the longer of the two to compute $T_L(10)$. We have

$$\begin{aligned} T_L(10) &= \min\{T_L(11) - 3, T_L(12) - 5\} \\ &= \min\{16, 17\} \\ &= 16. \end{aligned}$$

For nodes 8 and 7, $T_L(9) = T_L(10) - 2 = 14$ and $T_L(8) = T_L(12) - 2 = 20$, since only one arc leaves each of these nodes. For node 7, we find that

$$\begin{aligned} T_L(7) &= \min\{T_L(8) - 1, T_L(9) - 1, T_L(10) - 4\} \\ &= \min\{19, 13, 12\} \\ &= 12. \end{aligned}$$

In general, we have

$$T_L(j) = \min_i \{T_L(i) - \text{length}(\overrightarrow{j,i})\},$$

where i runs through all nodes for which there is an arc from node j to node i. The fact that $T_L(1)$ must be 0 is a good check on the accuracy of the event time calculations.

Recall that for any node j, $T_L(j)$ is the *latest* time by which all the activities leading from node j must be started to have the project finished on schedule. Also, $T_E(j)$ is the *soonest* time by which all the activities

leading from node j can be started. These activities could not be started earlier because at least one activity leading to node j had not been completed. The difference $T_L(j) - T_E(j)$ is called the **float time** or **slack time**. Any event that has zero float time is called a **critical event**. The next activity must be started without delay. A **critical path** is a path in the network that passes through only critical events and whose length is the value of T_E at the project-completion node. There may be more than one critical path in a network. The activities along a critical path are the ones whose lengths are determining the total length of the project.

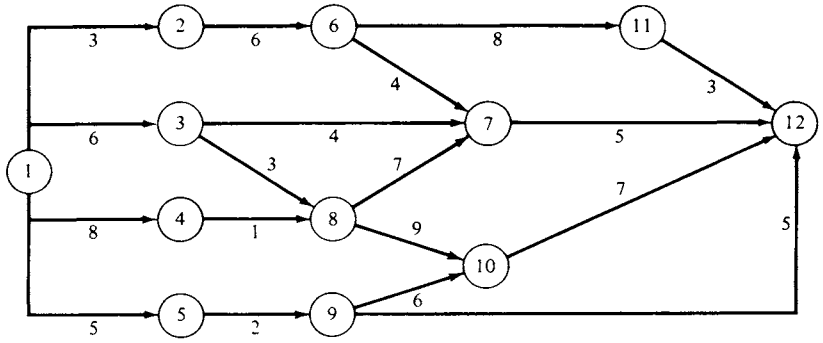
In our example there is only one critical path, namely, $1 \rightarrow 2 \rightarrow 4 \rightarrow 5 \rightarrow 7 \rightarrow 10 \rightarrow 11 \rightarrow 12$. This path is indicated by the heavy lines in Figure 5.38. △

5.6 EXERCISES

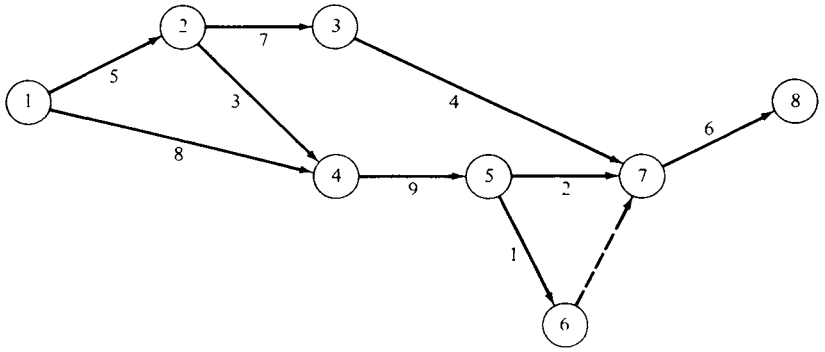
For the precedence networks given in Exercises 1–4,

(a) find the early event times;
(b) find the late event times;
(c) find a critical path.

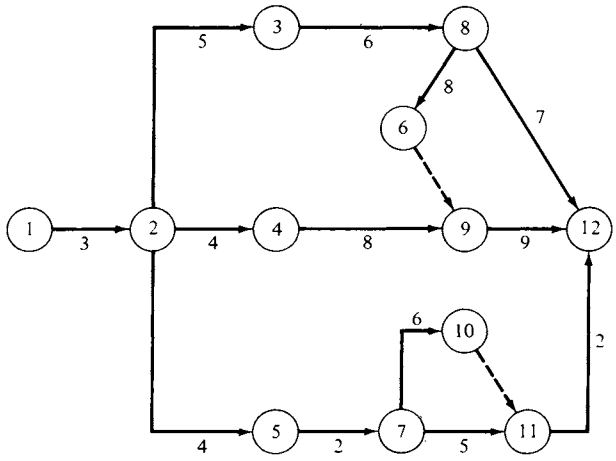
1.



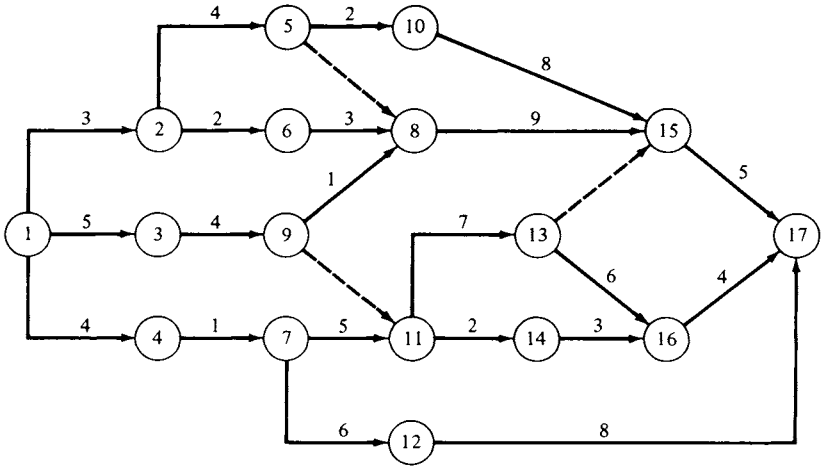
2.



3.

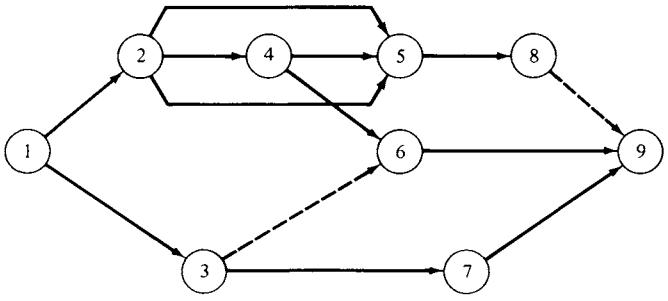


4.

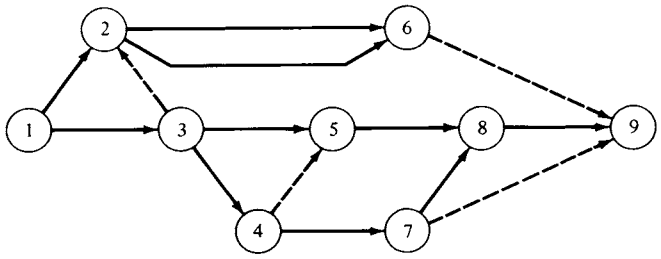


The networks given in Exercises 5 and 6 are to be models for certain projects. However, they are incorrectly constructed. Find and correct, if possible, the errors in the networks.

5.



6.



In Exercises 7 and 8 lists of activities and activity times are given. Also, we give the logical constraints on each activity. Construct a precedence network for each list of activities and find a critical path in the network.

7. A_1 precedes A_2, A_3, A_4, A_5, and A_6.
A_{10} follows A_2.
A_{11} follows A_3.
A_7 follows A_5.
A_8 follows A_4 and precedes A_{12}.
A_9 follows A_4 and A_7.
A_6 and A_9 precede A_{13}.
A_{12} follows A_{10} and A_{11}.
A_{12} and A_{13} terminate at the same time.

Activity	*Time*	*Activity*	*Time*
A_1	7	A_7	8
A_2	5	A_8	6
A_3	8	A_9	11
A_4	3	A_{10}	2
A_5	1	A_{11}	5
A_6	2	A_{12}	7
		A_{13}	9

8. A_1, A_2, A_3, and A_{17} start at the same time.
A_1 precedes A_4 and A_5.
A_6 follows A_2.
A_3 precedes A_7.
A_{10} follows A_5.
A_{13} follows A_6, A_7, and A_{10}.
A_4 precedes A_8 and A_9.
A_{12} follows A_8.
A_9 precedes A_{15} and A_{16}.
A_{11} follows A_{17}.
A_{14} follows A_{12}.
A_{11} precedes A_{18}.
A_{19} follows A_{15}.
A_{13}, A_{14}, A_{16}, A_{18}, and A_{19} terminate at the same time.

Activity	Time	Activity	Time
A_1	6	A_{11}	4
A_2	7	A_{12}	3
A_3	5	A_{13}	6
A_4	4	A_{14}	8
A_5	8	A_{15}	7
A_6	9	A_{16}	5
A_7	3	A_{17}	4
A_8	7	A_{18}	6
A_9	6	A_{19}	7
A_{10}	9		

Further Reading

Antell, J. M., and Woodhead, R. W. *Critical Path Methods in Construction Practice*, 2nd ed. Wiley, New York, 1970.

Moder, J. J., and Phillips, C. R. *Project Management with CPM and PERT*, 2nd ed. Van Nostrand, New York, 1970.

Shaffer, L. R., Ritter, J. B., and Meyer, W. L. *The Critical Path Method*. McGraw-Hill, New York, 1965.

5.7 COMPUTER ASPECTS (OPTIONAL)

In this chapter we have described several linear programming problems that because of their specialized structure are more efficiently solved using their own algorithms rather than using the simplex algorithm. In fact, all these models can be framed in terms of networks, and some commercial simplex optimizers can recognize that the coefficient matrix of such a model is a network even if the user had not envisioned it as such. The algorithm that is most widely available for general network problems is the **out-of-kilter algorithm**, which solves the minimum cost flow problem in a capacitated network.

The **minimum cost flow problem** can be mathematically formulated as follows. Consider a network with n nodes. With each node we associate a number b_i, indicating the availability of a single commodity at node i. If $b_i > 0$, there is b_i of the commodity available at node i, and node i is called a **source**; if $b_i < 0$, there is a demand for b_i of the commodity at node i, and node i is called a **sink**; and if $b_i = 0$, node i is called an **intermediate** or **transshipment** node. Let u_{ij} and l_{ij} denote the upper and lower capacities of arc $(\overline{i, j})$. Let c_{ij} denote the cost of shipping one unit from node i to node j along arc $(\overline{i, j})$. Since flow can be neither created nor destroyed at any node, we obtain the conservation of flow equations

$$\sum_{j=1}^{n} x_{ij} - \sum_{k=1}^{n} x_{ki} = b_i, \quad i = 1, 2, \ldots, n.$$

Of course, we are interested in minimizing the total cost

$$\sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij}.$$

Thus, a mathematical formulation of the minimum cost flow problem is

$$\text{Minimize} \quad \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij}$$

subject to

$$\sum_{j=1}^{n} x_{ij} - \sum_{k=1}^{n} x_{ki} = b_i, \quad i = 1, 2, \ldots, n$$

$$0 \leq l_{ij} \leq x_{ij} \leq u_{ij}, \quad \begin{cases} i = 1, 2, \ldots, n \\ j = 1, 2, \ldots, n. \end{cases}$$

When this problem is represented by a network diagram, it is convenient to label each arc with the triple numbers $[c_{ij}, u_{ij}, l_{ij}]$. If an arc has no upper bound on its capacity, only the cost of shipping one unit of goods along the arc is given.

The out-of-kilter algorithm finds, for a given amount of flow, the path that is cheapest. We examined a special case of this problem as we discussed the shortest route algorithm. There the amount of flow was one unit and each arc had an upper capacity of one unit and a lower capacity of zero units.

All the models that we have discussed in this chapter can be transformed to minimum cost flow models. Thus, the out-of-kilter algorithm could be used to solve any of them. As an example of the types of network computer codes available, we discuss the features of a typical code for the out-of-kilter algorithm. However, a description of the algorithm itself is beyond the scope of this book.

Input

The network is specified by giving a list of arcs. Each arc is specified by giving the names of the nodes it joins in the order "from–to." For each arc the user must also specify the upper bound on the capacity (using a large number if the capacity is unbounded), the lower bound on the capacity, the cost, and the initial flow. In many cases the initial flow is chosen to be zero. The computer code may be designed to handle only integer flow problems, so that the cost, initial flow, and capacities must be restricted to integer variables.

Since some codes do not have built-in routines for input verification, the user may have to check the inputs to make sure that:

1. there are no dangling nodes in the network;
2. there is only one source and one sink;
3. the initial flow is conserved at each interior node, namely, at node j,

$$\sum_p x_{pj} = \sum_q x_{jq};$$

4. the upper bound on each arc is greater than or equal to the lower bound for that arc.

Output

The code will have output options that allow the user to save the problem and restart it after making modifications to the network. The printed output will include a listing of the active arcs at an optimal solution along with the flow for each of these arcs. The output may also include a list of the $z_{ij} - c_{ij}$, which are marginal costs for increasing the flow one unit along the arcs $(\overrightarrow{i, j})$.

EXAMPLE 1. Consider the transportation problem (Example 1, Section 5.1) with a cost matrix and demand and supply vectors as follows:

$$\mathbf{C} = \begin{bmatrix} 5 & 7 & 9 & 6 \\ 6 & 7 & 10 & 5 \\ 7 & 6 & 8 & 1 \end{bmatrix}, \quad \mathbf{s} = \begin{bmatrix} 120 \\ 140 \\ 100 \end{bmatrix}, \quad \text{and} \quad \mathbf{d} = \begin{bmatrix} 100 \\ 60 \\ 80 \\ 120 \end{bmatrix}.$$

A minimum cost flow network that models this problem is given in Figure 5.39. Note that each source is a node and each destination is a node. Each source node is connected to every destination node by an arc. The cost of each arc is that specified by the matrix **C**. Also included are two additional nodes, a **supersource** and a **supersink**. These nodes serve to place the supply and demand constraints on the sources and destinations. The supersource is connected to each source by an arc that has cost zero, lower bound zero, and upper bound equaling the supply capacity of the source. A flow from the supersource to a source can be thought of as the process of manufacturing the goods at the source. The conservation of flow at each source guarantees that no more than the available supply may be shipped from that source.

Likewise, each destination is connected to the supersink by an arc with upper bound equaling the demand at the destination, lower bound zero, and cost zero.

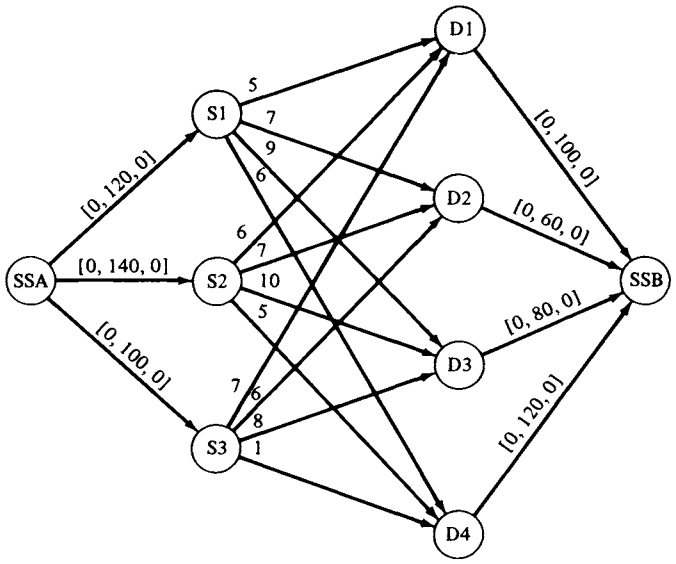


FIGURE 5.39

We show in Table 5.14 the input necessary for solving this problem using the Unisys UKILT 1100 Out-of-Kilter Network Optimization System. There is one additional arc required for this system, connecting the supersink to the supersource. The system expects all nodes to have arcs leading both in and out of them. Such a network is called a **circularization network**. The cost of the new arc is zero, and the lower and upper bounds are equal to the total supply. This ensures that the total supply is used and that the total demand is met. The nodes are labeled by numbers and the arcs, by the pair of nodes that they join. The initial flows are all chosen to be zero. Table 5.15 shows the output of UKILT 1100 for this problem.

Besides the out-of-kilter algorithm, there are implementations of the various specialized algorithms for particular settings. For example, the critical path method is typically used outside operations research settings, so that easily used codes for this algorithm have been developed.

CPM Codes

The CPM codes will produce charts and tables that can be used in the field by the project supervisors and foremen. Much of the effort in writing a CPM code goes into making this output meaningful to those who must use it.

There are about 100 commercially available CPM codes. These are owned either by larger corporations that do many complicated scheduling

TABLE 5.14

	From	To	Cost	Upper bound	Lower bound	Initial flow
1	BEGIN					
2	TRANSPORTATION					
3	ARCS					
4	S1	D1	5	1000	0	0
5	S1	D2	7	1000	0	0
6	S1	D3	9	1000	0	0
7	S1	D4	6	1000	0	0
8	S2	D1	6	1000	0	0
9	S2	D2	7	1000	0	0
10	S2	D3	10	1000	0	0
11	S2	D4	5	1000	0	0
12	S3	D1	7	1000	0	0
13	S3	D2	6	1000	0	0
14	S3	D3	8	1000	0	0
15	S3	D4	1	1000	0	0
16	SSA	S1	0	120	0	0
17	SSA	S2	0	140	0	0
18	SSA	S3	0	100	0	0
19	D1	SSB	0	100	0	0
20	D2	SSB	0	60	0	0
21	D3	SSB	0	80	0	0
22	D4	SSB	0	120	0	0
23	SSB	SSA	0	360	360	0
24	END					
25	SOLVE					
26	OUTPUT 11					
27	REPORT 11					
28	STOP					

tasks or by consulting firms. Experience has shown that networks representing more than 250 activities should certainly be processed on a computer. In fact, the break-even point between hand and computer processing may drop to as low as 100 activities if the network is extremely complicated. The larger CPM codes can handle problems with as many as 10,000 activities. The most popular CPM/project management codes are now run on PCs. They provide very easy to understand graphical displays and easy to use editing features. Some examples of such software are Super-Project, Time-Line, and Microsoft Project.

Besides using the CPM algorithm on a given network, a typical code does a substantial amount of input verification. It will check for loops, dangling activities, and duplicate activities. Most codes also have provisions for saving the network in computer-readable form (e.g., on disk or tape) so that changes in the network can be easily made.

TABLE 5.15

```
                    TITLE TRANSPORTATION
          NUMBER OF NODES:          9
           NUMBER OF ARCS:         20
               TOTAL COST:       1900

************************
*                      *
*  THE SOLUTION IS OPTIMAL  *
*                      *
************************
   TRANSPORTATION
ARC NUMBER FROM NODE TO NODE COST MARG COST UPPER BOUND LOWER BOUND FLOW
        1  S1        D1        5         0        1000           0  100
        2  S1        D2        7         1        1000           0    0
        3  S1        D3        9         0        1000           0   20
        4  S1        D4        6         2        1000           0    0
        5  S2        D1        6         0        1000           0    0
        6  S2        D2        7         0        1000           0   60
        7  S2        D3       10         0        1000           0   60
        8  S2        D4        5         0        1000           0   20
        9  S3        D1        7         5        1000           0    0
       10  S3        D2        6         4        1000           0    0
       11  S3        D3        8         2        1000           0    0
       12  S3        D4        1         0        1000           0  100
       13  SSA       S1        0        -1         120           0  120
       14  SSA       S2        0         0         140           0  140
       15  SSA       S3        0        -4         100           0  100
       16  D1        SSB       0        -4         100           0  100
       17  D2        SSB       0        -3          60           0   60
       18  D3        SSB       0         0          80           0   80
       19  D4        SSB       0        -5         120           0  120
       20  SSB       SSA       0        10         360         360  360
```

5.7 EXERCISES

1. Why is a supersink necessary in modeling a transportation problem as a minimum cost flow problem? What aspect of the problem could it represent?
2. Model the assignment problem as a minimum cost flow problem. Assume that there are n persons and n jobs.
3. Model the shortest route problem as a minimum cost flow problem.
4. Model the critical path method as a minimum cost flow problem.

5.7 PROJECTS

1. **The transshipment problem.** In many applications of the transportation problem we encounter a situation in which a node serves merely as a transshipment

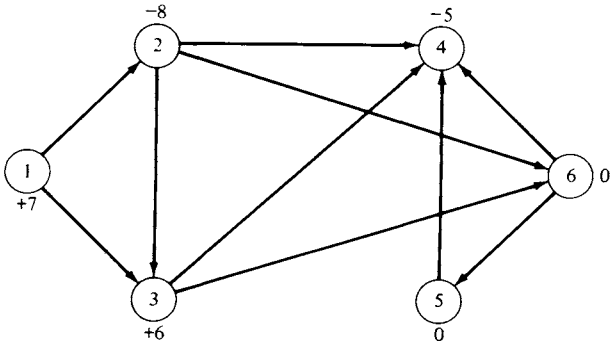


FIGURE 5.40

point. That is, there is neither supply nor demand at the point, but goods are shipped through the point. Consider the network in Figure 5.40, where the numbers beside the nodes are the b_i described in this Section. The nodes can be divided into five classes.

Pure source—all arcs lead away from the node, $b_i > 0$
Pure sink—all arcs lead into the node, $b_i < 0$
Transshipment source—arcs leading into and out of the node, $b_i > 0$
Tranship ment sink—arcs leading into and out of the node, $b_i < 0$
Pure transshipment point—arcs leading into and out of the node, $b_i = 0$

(a) Classify the nodes in the network in Figure 5.40.

Every transshipment problem can be formulated as a minimum cost flow problem. This formulation may require the introduction of a supersource or supersink. A supersource will be necessary when there is more than one pure source in the network. This supersource is connected to each pure source with an arc whose capacity has an upper bound equal to the supply at the pure source. Likewise, a supersink is necessary when there is more than one pure sink. A minimum cost flow problem specifies a particular value of flow that must be sent from the supersource to the supersink. Because there must be conservation of flow at each node, the sum of the amounts available at the sources of a transshipment problem must equal the sum of the amounts required at the sinks before the problem can be formulated as a minimum cost flow problem. If these sums are not equal, an additional pure source or pure sink must be added as a dummy to the network to compensate for this difference.

(b) Why does a dummy sink node have to be connected only to the source node and not to the transshipment nodes or other sink nodes?
(c) Classify the nodes of the network in Figure 5.41.
(d) Formulate the network in Figure 5.41 as a minimum cost flow problem.

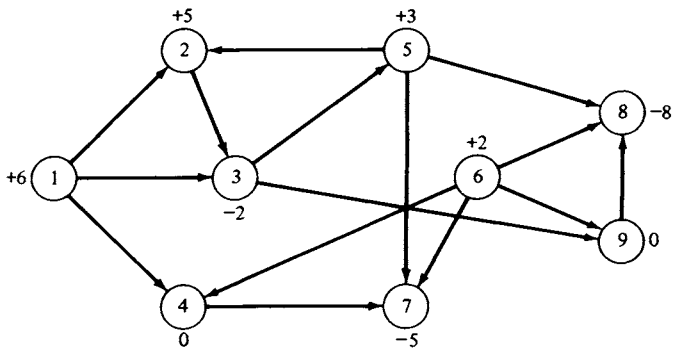


FIGURE 5.41

2. If your computer center has a network code, formulate and run the following.
   (a) Figure 5.4, Section 5.4
   (b) Example 2, Section 5.1
   (c) Example 1, Section 5.1
   (d) The shortest route example in Section 5.5

APPENDIX

A

Karmarkar's Algorithm

IMPLEMENTATIONS OF THE simplex method, and later the revised simplex method, have been used as the primary techniques for solving linear programming problems since the simplex method was first discovered by Dantzig. The experience of solving a huge number of linear programming problems over a number of years led to the conjecture that the number of iterations of the simplex method is proportional to $\frac{3}{2}n$, where n is the number of variables in the problem. Furthermore, the number of operations per iteration is proportional to mn, where m is the number of constraints. Such an estimate indicates that the simplex algorithm is a **polynomial time** algorithm because its running time is proportional to a polynomial in n. However, in 1972 Klee and Minty provided a particular linear programming problem that interacted badly with the rules for choosing entering and departing variables and consequently had a running time proportional to $2^{n/2} - 1$. Thus the simplex algorithm is not a polynomial algorithm after all. Note that the problem used to disprove the conjecture was tailored to the particular pivoting strategy used by the simplex method. Jeroslow later showed that for any choice of pivoting

strategy, an equivalent problem could be constructed that had the same bad behavior. Although problems such as this have not arisen in applications, there is no guarantee that such a problem will not occur.

In the mid-1980s a new approach to solving linear programming problems was proposed. It was based on work done in investigating algorithms to follow paths in the interior of the convex set of feasible solutions. Although these algorithms were first devised in the 1950s, they did not initially achieve the performance of the simplex method. In 1979 Khachiyan proved that by using the appropriate interior path algorithm, one can guarantee that a linear programming problem will be solved in polynomial time. This proof renewed interest in interior path methods.

Instead of following the edges of the convex set from extreme point to extreme point, eventually reaching an optimal solution, the interior path methods burrow through the interior of the set to find an optimal solution. Because they are not constrained to the directions of the edges, nor their length, it is reasonable to assume that the interior path methods might be faster than the edge-following method. However, there has been no clear demonstration of the superiority of interior path methods. Most of the users of sophisticated linear programming software for frequent solutions of large problems still use software based on the edge-following simplex algorithm.

The most successful of the interior path methods has been that due to Karmarkar. We will give a description of the main points of the algorithm without presenting the technical details. A full description of the algorithm is quite complex and involves geometric ideas that are beyond the scope of this book. A more complete description may be found in Nering and Tucker (Further Reading).

To apply Karmarkar's method to a general linear programming problem, we first convert the problem to a minimization one of the form:

$$\begin{aligned} &\text{Minimize} \quad z = \mathbf{c}^{\mathrm{T}}\mathbf{x} \\ &\text{subject to} \\ &\qquad \mathbf{A}\mathbf{x} = \mathbf{b} \\ &\qquad \mathbf{x} \geq \mathbf{0}. \end{aligned}$$

Let $\mathbf{x}_0$ be a feasible solution to this problem. For the Karmarkar method, $\mathbf{x}_0$ will be chosen to be in the interior of the set of feasible solutions. That is, $\mathbf{x}_0$ will be chosen to not lie on an edge or hyperplane forming the boundary of the set of feasible solutions. As contrasted to the simplex method in which most of the components of a basic feasible solution are 0, all the components of $\mathbf{x}_0$ will be nonzero. We want to calculate a new feasible solution $\mathbf{x}_1$ that is closer to the optimal solution of the problem. Let $\mathbf{x}_1 = \mathbf{x}_0 + \Delta\mathbf{x}$. We investigate how to choose $\Delta\mathbf{x}$ to meet this criterion. The important aspects of the choice are the direction of $\Delta\mathbf{x}$

as a vector and its magnitude, or step size, to ensure that $\mathbf{x}_1$ remains within the set of feasible solutions. Note that the objective function at the new feasible solution $\mathbf{x}_1$ is given by

$$\mathbf{c}^T\mathbf{x}_1 = \mathbf{c}^T\mathbf{x}_0 + \mathbf{c}^T\Delta\mathbf{x}.$$

Since this is a minimization problem, the quantity $\mathbf{c}^T\Delta\mathbf{x}$ should be negative and as large as possible in absolute value, so that the value of the objective function at $\mathbf{x}_0$, $\mathbf{c}^T\mathbf{x}_0$, is reduced as much as possible. Also

$$\mathbf{A}\mathbf{x}_1 = \mathbf{A}\mathbf{x}_0 + \mathbf{A}\Delta\mathbf{x}.$$

If $\mathbf{x}_1$ is to be feasible, we must have $\mathbf{A}\mathbf{x}_1 = \mathbf{b}$, so, by substitution, $\mathbf{b} = \mathbf{b} + \mathbf{A}\Delta\mathbf{x}$ or $\mathbf{A}\Delta\mathbf{x} = \mathbf{0}$. This means that $\Delta\mathbf{x}$ must be chosen to lie in the null space of $\mathbf{A}$ (see Example 5, Section 0.4). Finally, $\mathbf{x}_1$ must be nonnegative, which means that

$$\mathbf{0} \leq \mathbf{x}_1 = \mathbf{x}_0 + \Delta\mathbf{x}$$

or

$$\Delta\mathbf{x} \geq -\mathbf{x}_0.$$

We now develop a way to construct a vector in the null space of $\mathbf{A}$ from any given vector $\mathbf{z}$. This process will allow us to choose $\Delta\mathbf{x}$ so that $\mathbf{c}^T\Delta\mathbf{x}$ has the proper value. We claim that for any vector $\mathbf{z}$, the vector

$$\mathbf{P}\mathbf{z} = \left(\mathbf{I} - \mathbf{A}^T(\mathbf{A}\mathbf{A}^T)^{-1}\mathbf{A}\right)\mathbf{z} = \mathbf{z} - \mathbf{A}^T(\mathbf{A}\mathbf{A}^T)^{-1}\mathbf{A}\mathbf{z}$$

lies in the null space of $\mathbf{A}$; that is, $\mathbf{A}(\mathbf{P}\mathbf{z}) = \mathbf{0}$ since

$$\mathbf{A}(\mathbf{P}\mathbf{z}) = \mathbf{A}\mathbf{z} - \mathbf{A}\mathbf{A}^T(\mathbf{A}\mathbf{A}^T)^{-1}\mathbf{A}\mathbf{z} = \mathbf{A}\mathbf{z} - \mathbf{A}\mathbf{z} = \mathbf{0}.$$

If we let $\mathbf{z} = -\mathbf{c}$ and choose $\Delta\mathbf{x} = -\mathbf{P}\mathbf{c}$ then $\mathbf{c}^T\Delta\mathbf{x}$ is as large negative as possible.

We have now found the correct direction $(-\mathbf{P}\mathbf{c})$. In computing the magnitude of the step, we want $\mathbf{x}_1$ to lie in the interior of the set of feasible solutions so we must guarantee that we do not step as far as the boundary. This guarantee comes by choosing $\mathbf{x}_1$ to be

$$\mathbf{x}_0 - 0.98s\,\frac{\mathbf{P}\mathbf{c}}{|\mathbf{P}\mathbf{c}|},$$

where s, the distance from $\mathbf{x}_0$ to the boundary, can be computed from the inequality $\Delta\mathbf{x} \geq -\mathbf{x}_0$ and where $-\mathbf{P}\mathbf{c}/|\mathbf{P}\mathbf{c}|$ is a vector of length 1 in the direction of $-\mathbf{P}\mathbf{c}$.

The two features of this algorithm that we have overlooked are rescaling and a stopping condition. In practice, the general minimization problem is transformed to a restricted form that has an obvious feasible solution for the initial value and whose objective function minimum is

known to be zero. After each iteration of the algorithm, variables are rescaled to bring the current feasible solution to a standard form and to allow some freedom of movement near the boundary of the set of feasible solutions. We can stop the iterations when the objective function is within a preset distance from zero. We will then have to recover the solution to the linear programming problem as it was originally presented by undoing the various transformations that we performed on the given problem.

Examples of Karmarkar's algorithm that clearly show its workings and its power in a simple setting are impossible to construct. The iterations of the algorithm require careful attention to numerical computations, some of which are quite complex. The following example constrains the set of feasible solutions to two dimensions, yet it highlights the important properties of the algorithm.

EXAMPLE 1. Consider the linear programming problem,

$$\begin{aligned} &\text{Maximize} \quad z = 20x + 30y \\ &\text{subject to} \\ &-3x + y \leq 1 \\ &-x + y \leq 3 \\ &-x + 2y \leq 8 \\ &y \leq 6 \\ &x + 2y \leq 18 \\ &0.9x + 0.9y \leq 12 \\ &x \leq 9 \\ &x - y \leq 6 \\ &x - 2y \leq 4 \\ &x - 3y \leq 3 \\ &x \geq 0, \quad y \geq 0. \end{aligned}$$

By adding a slack variable to each constraint and converting the problem to a minimization problem, we can cast it in the form necessary for the Karmarkar algorithm. We obtain the problem,

$$\begin{aligned} &\text{Minimize} \quad z = \mathbf{c}^{\mathrm{T}}\mathbf{x} \\ &\text{subject to} \\ &\mathbf{A}\mathbf{x} = \mathbf{b} \\ &\mathbf{x} \geq \mathbf{0}, \end{aligned}$$

where

$$\mathbf{A} = \begin{bmatrix} -3 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 2 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 2 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0.9 & 0.9 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & -3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{b}^T = [1 \quad 3 \quad 8 \quad 6 \quad 18 \quad 12 \quad 9 \quad 6 \quad 4 \quad 3]$$

and

$$\mathbf{c}^T = [-20 \quad -30 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0].$$

Since the problem has not been rescaled, an initial feasible solution for the Karmarkar algorithm looks rather unintuitive, but can be constructed from knowing that the point (1, 1) lies in the set of feasible solutions of the original maximization problem. We simply compute the values of the slack variables and find that

$$\mathbf{x}_0^T = [1 \quad 1 \quad 3 \quad 3 \quad 7 \quad 5 \quad 15 \quad 10.2 \quad 8 \quad 6 \quad 5 \quad 5]$$

We will step in the direction $-\mathbf{Pc}$ from $\mathbf{x}_0$. Using the definition of $\mathbf{P}$ given above and software that can perform numerical matrix manipulations, we find that the unit vector in the direction $-\mathbf{Pc}$ is

$$\mathbf{u} = -\frac{\mathbf{Pc}}{|\mathbf{Pc}|} = \begin{bmatrix} 0.2106943871859121 \\ 0.1863525550415169 \\ 0.4457306065162191 \\ 0.0243418321443944 \\ -0.1620107228971102 \\ -0.1863525550415162 \\ -0.5833994972689248 \\ -0.3573422480046798 \\ -0.2106943871859076 \\ -0.0243418321443999 \\ 0.1620107228971091 \\ 0.3483632779386140 \end{bmatrix}.$$

We let $\Delta\mathbf{x} = s\mathbf{u}$ and solve the system of inequalities $s\mathbf{u} \geq -\mathbf{x}_0$ for s. We find that $s = 25.71136943075831$. Thus, the new feasible solution for this

problem is

$$\mathbf{x}_1 = \mathbf{x}_0 + 0.98s\mathbf{u} = \begin{bmatrix} 6.308896401405732 \\ 5.695551799297400 \\ 14.231137404919790 \\ 3.613344602108312 \\ 2.917792802811221 \\ 0.304448200702618 \\ 0.300000000000000 \\ 1.195996619367340 \\ 2.691103598594382 \\ 5.386655397891551 \\ 9.082207197188751 \\ 13.777758996485850 \end{bmatrix}.$$

These steps are repeated until we obtain a new feasible solution that is within a preset distance of the current solution.

This example was solved using a version of Karmarkar's algorithm that included rescaling at each step. The interior path followed by this version is shown in Figure A.1. △

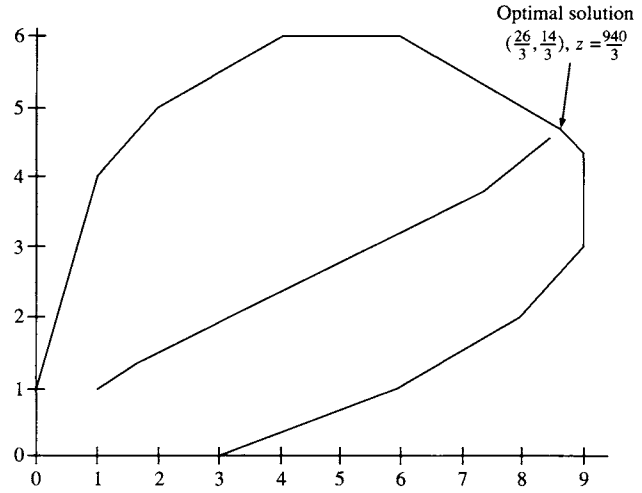


FIGURE A.1

Further Reading

Karmarkar, N. "A New Polynomial–Time Algorithm for Linear Programming." *Combinatorica* **4** (1984), 373–395.

Nering, Evar D., and Tucker, Albert W. *Linear Programs and Related Problems*. Academic Press, New York, 1993.

Rockett, A. M., and Stevenson, J. C. "Karmarkar's Algorithm." *Byte* (September 1987), 156.

APPENDIX

B

Microcomputer Software

SINCE THE FIRST edition of this book appeared, major developments have taken place in the availability of software that can be effectively used in a linear programming course. As has already been noted in the text, one of the fortuitous events in the history of linear programming was the prodigious growth of computing power coupled with its precipitous cut in cost. Thus, as early as the 1960s powerful programs were developed to solve sizable linear programming problems on large computers.

Nowadays, we are in the fortunate position of having a number of software packages for solving linear programming problems that are inexpensive, are easy to use, and will run on a personal computer. In this appendix we provide some information on the following software packages: LINDO, MPSIII/pc, and the MATLAB optimization toolbox, programs that solve a problem presented to them in mathematical form; LINGO and PAM, programs that formulate a mathematical model of the problem and then proceed to solve it; and finally WHAT'S BEST!, a spreadsheet-based program that solves a mathematically presented program. These programs

solve linear, integer, and mixed-integer programming problems and run on a variety of platforms, including PCs, Macintoshes, and various work stations. They are each available in a number of different versions, whose capabilities differ according to the platform. For example, standard LINDO will handle at most 200 variables, whereas extended LINDO will handle at most 100,000 variables. Keep in mind that to run the MATLAB optimization toolbox it is necessary to have MATLAB itself; similarly, to run WHAT'S BEST! it is necessary to have a spreadsheet that it supports, which includes the most widely used four or five spreadsheet programs.

LINDO, LINGO, and WHAT'S BEST! are available from LINDO Systems, 1415 North Dayton Avenue, Chicago 60622; telephone, (800) 441-2378. MATLAB and the MATLAB optimization toolbox are available from the The Math-Works, Inc., Cochituate Place, 24 Prime Park Way, Natick, Massachusetts 01760; telephone, (508) 653-2997. MPSIII/pc and PAM are available from Ketron Management Science, 2200 Wilson Boulevard, No. 220, Arlington, Virginia 22201; telephone, (703) 558-8700.

Further Reading

Moré, Jorge J., and Wright, Stephen J. *Optimization Software Guide*. SIAM, Philadelphia, 1993.

APPENDIX

C

SMPX

INTRODUCTION

There are two main ways to use software to learn about linear programming and its extensions. One method uses a small version of a substantial linear programming code to find the solution to a problem and to interpret the solution in light of the scenario that was being modeled. This version should have good input and output capabilities but can hide all of the simplex algorithm. It is the answer that is most important.

Alternatively, one can use a specially tailored version of the simplex algorithm as an experimental tool for understanding the subtleties of the algorithm, the sequence of steps taken as the problem is solved, and the consequences of the choices and decisions that are hidden in the problem-solving code.

Using both types of software will contribute to understanding both the algorithmic process and its application to realistic problems. LINDO is an example of applications-oriented software; a copy can be obtained by mailing the order card that accompanies this book. SMPX has been created by Evar D. Nering to allow experimentation with the simplex

algorithm: it is not recommended for use in solving applied problems. A disk with this program and some examples from the text is included with this book. A brief description of how to get started conducting experiments with SMPX follows. More information on the details of the software is available through the help system of SMPX.

SMPX

SMPX is a program for an IBM compatible personal computer and is designed for instructional purposes to support learning the principles of linear programming and the simplex algorithm. The program can handle only small tableaux, up to 10 by 10, that can be displayed on a computer screen. It also displays intermediate results that are useful to someone learning about the theory of linear programming but are unnecessary for using the simplex algorithm in an applied problem.

The user interface of SMPX is similar in appearance to the user interface that characterizes Microsoft Windows. However, the program runs under DOS and the screen is a character-oriented display. Commands can be entered either with the keyboard and function keys or by using a mouse to activate buttons and menu items. The arrow keys and tab keys are used to navigate among the buttons and menus. A linear programming problem is described to SMPX by using its equivalent tableau and marking variables that are artificial. The software automatically provides slack variables, but it assumes that all constraints are written with $\leq$ signs.

Several files of examples and exercises taken from the book are included on the disk. These can be used as you explore the use of the software. You can add to these files or create other files of problems as you continue through the book and expand your investigations. The software allows you to create new problems either by starting from a template with the correct number of variables and constraints or by modifying an existing problem.

GETTING STARTED

The simplest way to use SMPX is to create a directory on your hard drive called \LP. Then copy all the files from the disk to your hard drive. Start the program by giving its name as a command. Once the program is running you can explore the choices on the menu bar, including HELP, which starts the extensive context-sensitive help system.

When you start SMPX, the first action you will usually take is to open a problem file. You can scan its contents in the preview window and then select a problem to solve in the work window.

To stop work and exit from SMPX use the command Alt-X.

Answers to Odd-Numbered Exercises

CHAPTER 0

Section 0.1, page 9

1. $a = 4, \quad b = 2, \quad c = 9, \quad d = -1$

3. (a) Not possible (b) Not possible

(c) $\begin{bmatrix} 2 & 3 \\ 3 & 1 \\ 1 & 2 \end{bmatrix}$ (d) $\begin{bmatrix} 4 & 4 & 4 \\ 2 & 4 & 4 \\ 1 & 11 & 4 \end{bmatrix}$

5. $\mathbf{AB} = \begin{bmatrix} 9 & 11 \\ 10 & 2 \end{bmatrix}, \quad \mathbf{BA} = \begin{bmatrix} -5 & 1 \\ 12 & 16 \end{bmatrix}$

7. $\mathbf{AC} = \mathbf{BC} = \begin{bmatrix} -3 & -12 \\ -8 & -32 \end{bmatrix}$

9. $\begin{aligned} 3x_1 - 2x_2 + 5x_3 + 4x_4 &= 1 \\ 4x_1 + 2x_2 + x_3 &= -3 \\ 3x_1 + 4x_2 - 2x_3 + x_4 &= 5 \end{aligned}$

Section 0.2, page 20

1. $\begin{bmatrix} 1 & 0 & 0 & 11 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & -4 \end{bmatrix}$ 3. $\begin{bmatrix} 1 & 0 & -\frac{7}{3} & 2 & -\frac{13}{3} \\ 0 & 1 & 2 & -1 & 3 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$

5. (a) $x = 1, \quad y = 2, \quad z = -2$
   (b) No solution

7. (a) No solution
   (b) $x = 3, \quad y = -2, \quad z = 0, \quad w = 1$

9. (a) No solution
   (b) $x = \frac{7}{3} - \frac{7}{9}r, \quad y = \frac{4}{3} + \frac{8}{9}r, \quad z = r$, where r is any real number.

11. (a) $x = 0, \quad y = 0, \quad z = 0$
    (b) $x = -r, \quad y = 0, \quad z = 0, \quad w = r$, where r is any real number.

13. (i) $a = -3$
    (ii) $a =$ any real number except 3 or -3
    (iii) $a = 3$

Section 0.3, page 27

1. $\begin{bmatrix} \frac{4}{5} & -\frac{1}{5} \\ -\frac{3}{5} & \frac{2}{5} \end{bmatrix}$

3. (a) $\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$ (b) $\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$ (c) $\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & -3 & 1 \end{bmatrix}$

5. (a) $\begin{bmatrix} -\frac{5}{7} & \frac{3}{7} \\ \frac{4}{7} & -\frac{1}{7} \end{bmatrix}$ (b) No inverse (c) $\begin{bmatrix} \frac{14}{9} & -\frac{1}{3} & -\frac{1}{9} \\ -\frac{2}{3} & 0 & \frac{1}{3} \\ -\frac{5}{9} & \frac{1}{3} & \frac{1}{9} \end{bmatrix}$

7. (a) $\begin{bmatrix} \frac{2}{5} & -\frac{3}{10} \\ \frac{1}{5} & \frac{1}{10} \end{bmatrix}$ (b) $\begin{bmatrix} \frac{1}{2} & 0 & -\frac{1}{2} \\ \frac{7}{4} & -\frac{3}{2} & -\frac{5}{4} \\ -1 & 1 & 1 \end{bmatrix}$ (c) No inverse

9. (a) Does not exist

   (b) $\begin{bmatrix} 4 & -2 & -\frac{3}{2} \\ -\frac{13}{7} & 1 & \frac{9}{14} \\ -\frac{12}{7} & 1 & \frac{11}{14} \end{bmatrix}$

   (c) Does not exist

Section 0.4, page 32

3. (b) and (c)
5. (a) and (b)

Section 0.5, page 41

1. (a) and (b)
3. (a) and (b)
5. (c): $\begin{bmatrix}1\\1\\2\end{bmatrix}+\begin{bmatrix}1\\2\\-1\end{bmatrix}+\begin{bmatrix}3\\4\\-2\end{bmatrix}=\begin{bmatrix}5\\7\\-1\end{bmatrix}$

   (d): $\begin{bmatrix}2\\1\\3\end{bmatrix}+2\begin{bmatrix}1\\2\\-1\end{bmatrix}=\begin{bmatrix}4\\5\\1\end{bmatrix}$
7. (a)
9. (a)
11. (c): $\begin{bmatrix}-3\\0\\1\end{bmatrix}=\begin{bmatrix}0\\1\\0\end{bmatrix}-2\begin{bmatrix}3\\2\\1\end{bmatrix}+3\begin{bmatrix}1\\1\\1\end{bmatrix}$
13. (a) $\begin{bmatrix}1\\0\\0\end{bmatrix}$ (b) $\begin{bmatrix}1\\2\\-2\end{bmatrix}$ (c) $\begin{bmatrix}0\\1\\2\end{bmatrix}$ (d) $\begin{bmatrix}-1\\2\\4\end{bmatrix}$
19. 2

CHAPTER 1

Section 1.1, page 60

1. Let x = amount of PEST (in kilograms) and
   y = amount of BUG (in kilograms).

   Minimize $z = 3x + 2.5y$
   subject to

$$30x + 40y \geq 120$$
$$40x + 20y \leq 80$$
$$x \geq 0, \quad y \geq 0.$$

   To change to standard form, change the objective function and the first constraint.
3. Let x = number of Palium pills prescribed per day and
   y = number of Timade pills prescribed per day.

Minimize $z = 0.4x + 0.3y$
subject to

$$\begin{aligned} 4x + \quad 2y &\geq 10 \\ 0.5x + 0.5y &\leq \ 2 \\ x \geq 0, \quad y &\geq 0. \end{aligned}$$

To change to standard form, change the objective function and the first constraint.

5. Let x = number of kilograms of Super and
y = number of kilograms of Deluxe.

Maximize $z = 20x + 30y$
subject to

$$\begin{aligned} 0.5x + 0.25y &\leq 120 \\ 0.5x + 0.75y &\leq 160 \\ x \geq 0, \quad y &\geq 0. \end{aligned}$$

This model is in standard form.

7. Let x_1 = number of bags of Regular Lawn (in thousands)
x_2 = number of bags of Super Lawn (in thousands), and
x_3 = number of bags of Garden (in thousands).

Maximize $z = 300x_1 + 500x_2 + 400x_3$
subject to

$$\begin{aligned} 4x_1 + 4x_2 + 2x_3 &\leq 80 \\ 2x_1 + 3x_2 + 2x_3 &\leq 50 \\ x_1 \geq 0, \quad x_2 \geq 0, \quad x_3 &\geq 0. \end{aligned}$$

This model is in standard form.

9. Let x_1 = number of books in paperback binding,
x_2 = number of books in bookclub binding, and
x_3 = number of books in library binding.

Maximize $z = 0.5x_1 + 0.8x_2 + 1.2x_3$
subject to

$$\begin{aligned} 2x_1 + 2x_2 + \ 3x_3 &\leq 420 \\ 4x_1 + 6x_2 + 10x_3 &\leq 600 \\ x_1 \geq 0, \quad x_2 \geq 0, \quad x_3 &\geq 0. \end{aligned}$$

This model is in standard form.

11. Let x_{ij} = amount of the ith ingredient in the jth mixture (in kilograms), where

Ingredient 1 = Sunflower seeds
Ingredient 2 = Raisins
Ingredient 3 = Peanuts
Mixture 1 = Chewy
Mixture 2 = Crunchy
Mixture 3 = Nutty

$$\text{Maximize} \quad 2\sum_{i=1}^{3} x_{i1} + 1.6\sum_{i=1}^{3} x_{i2} + 1.2\sum_{i=1}^{3} x_{i3} - \sum_{j=1}^{3} x_{1j} - 1.5\sum_{j=1}^{3} x_{2j} - 0.8\sum_{j=1}^{3} x_{3j}$$

subject to

$$\sum_{j=1}^{3} x_{1j} \le 100$$

$$\sum_{j=1}^{3} x_{2j} \le 80$$

$$\sum_{j=1}^{3} x_{3j} \le 60$$

$$0.6x_{11} - 0.4x_{21} + 0.6x_{31} \le 0$$
$$-0.2x_{11} - 0.2x_{21} + 0.8x_{31} \le 0$$
$$-0.4x_{12} + 0.6x_{22} + 0.6x_{32} \le 0$$
$$0.8x_{13} - 0.2x_{23} - 0.2x_{33} \le 0$$
$$0.6x_{13} + 0.6x_{23} - 0.4x_{33} \le 0$$
$$x_{ij} \ge 0, \quad i = 1,2,3; \quad j = 1,2,3$$

Section 1.2, page 68

1. Maximize $z = [20 \quad 30]\begin{bmatrix} x \\ y \end{bmatrix}$

   subject to

$$\begin{bmatrix} 0.4 & 0.3 \\ 0.2 & 0.4 \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} \le \begin{bmatrix} 18 \\ 14 \end{bmatrix}$$

$$\begin{bmatrix} x \\ y \end{bmatrix} \ge \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

3. Maximize $z = 3x + 2y + 3v - 2w$

   subject to

$$\begin{bmatrix} 2 & 6 & 2 & -4 \\ -2 & -6 & -2 & 4 \\ 3 & 2 & -5 & 1 \\ -3 & -2 & 5 & -1 \\ 6 & 7 & 2 & 5 \end{bmatrix}\begin{bmatrix} x \\ y \\ v \\ w \end{bmatrix} \le \begin{bmatrix} 7 \\ -7 \\ 8 \\ -8 \\ 4 \end{bmatrix}$$

$$\begin{bmatrix} x \\ y \\ v \\ w \end{bmatrix} \ge \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

5. Maximize $z = \begin{bmatrix} -3 & -2 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ u \\ v \end{bmatrix}$

subject to

$$\begin{bmatrix} 2 & 1 & 1 & 0 \\ 3 & -2 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ u \\ v \end{bmatrix} = \begin{bmatrix} 4 \\ 6 \end{bmatrix}$$

$$\begin{bmatrix} x \\ y \\ u \\ v \end{bmatrix} \geq \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

7. Maximize $z = \begin{bmatrix} 3 & 2 & 3 & -2 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ v \\ w \\ u \end{bmatrix}$

subject to

$$\begin{bmatrix} 2 & 6 & 2 & -4 & 0 \\ 3 & 2 & -5 & 1 & 0 \\ 6 & 7 & 2 & 5 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ v \\ w \\ u \end{bmatrix} = \begin{bmatrix} 7 \\ 8 \\ 4 \end{bmatrix}$$

$$\begin{bmatrix} x \\ y \\ v \\ w \\ u \end{bmatrix} \geq \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

9. Maximize $z = \begin{bmatrix} 20 & 30 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ u \\ v \end{bmatrix}$

subject to

$$\begin{bmatrix} 0.5 & 0.25 & 1 & 0 \\ 0.5 & 0.75 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ u \\ v \end{bmatrix} = \begin{bmatrix} 120 \\ 160 \end{bmatrix}$$

$$\begin{bmatrix} x \\ y \\ u \\ v \end{bmatrix} \geq \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

11. (a) $\begin{bmatrix} 2 & 2 \\ 5 & 3 \end{bmatrix} \begin{bmatrix} 2 \\ 1 \end{bmatrix} = \begin{bmatrix} 6 \\ 13 \end{bmatrix} \leq \begin{bmatrix} 8 \\ 15 \end{bmatrix}; \quad \begin{bmatrix} 2 \\ 1 \end{bmatrix} \geq \begin{bmatrix} 0 \\ 0 \end{bmatrix}; \quad z = 340$

$\begin{bmatrix} 2 & 2 \\ 5 & 3 \end{bmatrix} \begin{bmatrix} 1 \\ 3 \end{bmatrix} = \begin{bmatrix} 8 \\ 14 \end{bmatrix} \leq \begin{bmatrix} 8 \\ 15 \end{bmatrix}; \quad \begin{bmatrix} 1 \\ 3 \end{bmatrix} \geq \begin{bmatrix} 0 \\ 0 \end{bmatrix}; \quad z = 420$

(b) $\begin{bmatrix} 2 & 2 \\ 5 & 3 \end{bmatrix}\begin{bmatrix} 3 \\ 1 \end{bmatrix} = \begin{bmatrix} 8 \\ 18 \end{bmatrix} \leq \begin{bmatrix} 8 \\ 15 \end{bmatrix}$

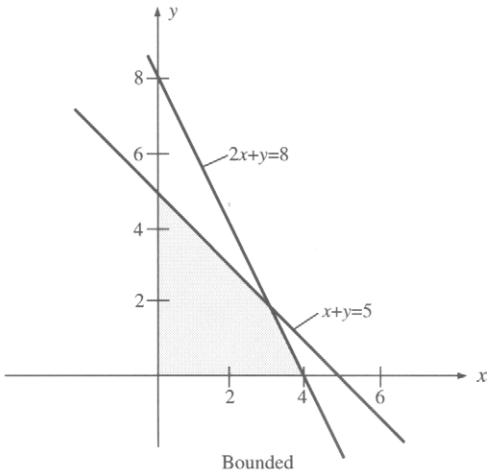
$\begin{bmatrix} 2 & 2 \\ 5 & 3 \end{bmatrix}\begin{bmatrix} 2 \\ 2 \end{bmatrix} = \begin{bmatrix} 8 \\ 16 \end{bmatrix} \leq \begin{bmatrix} 8 \\ 15 \end{bmatrix}$

$\begin{bmatrix} -2 \\ 3 \end{bmatrix} \geq \begin{bmatrix} 0 \\ 0 \end{bmatrix}$

13. (a) $x = 2, \quad y = 3, \quad u = 3, \quad v = 4$
(b) Impossible

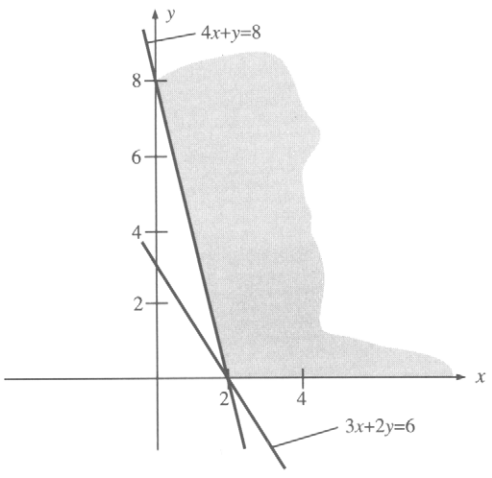
Section 1.3, page 81

1.



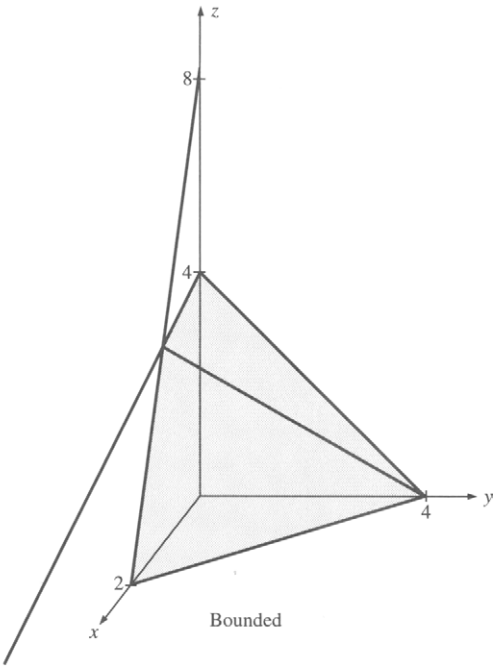
Bounded

3.



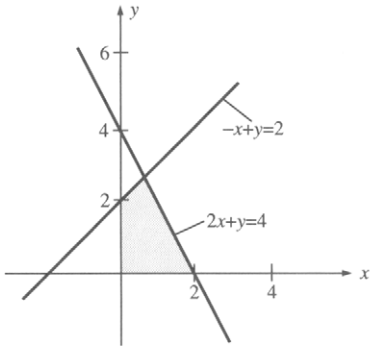
Unbounded

5.

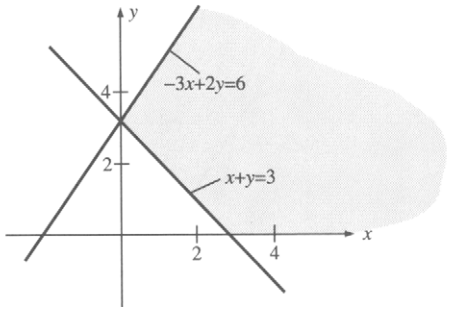


Bounded

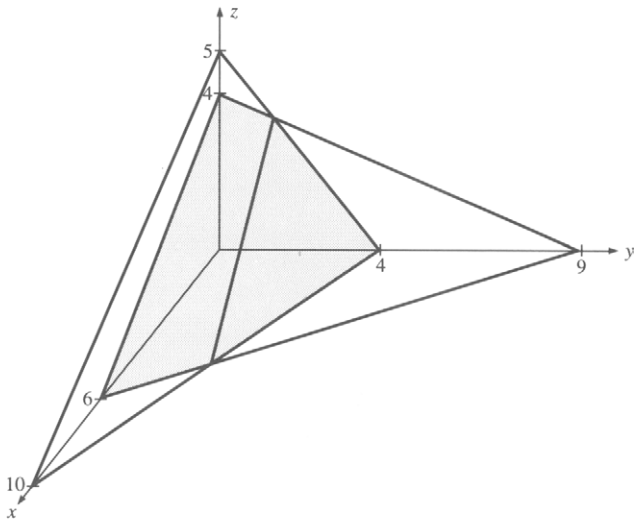
7.



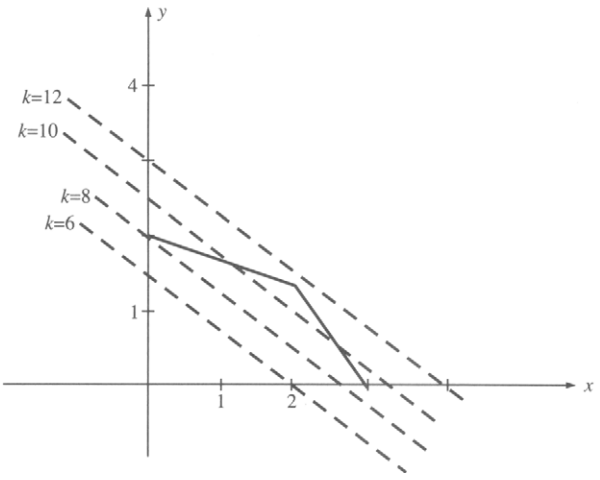
9.



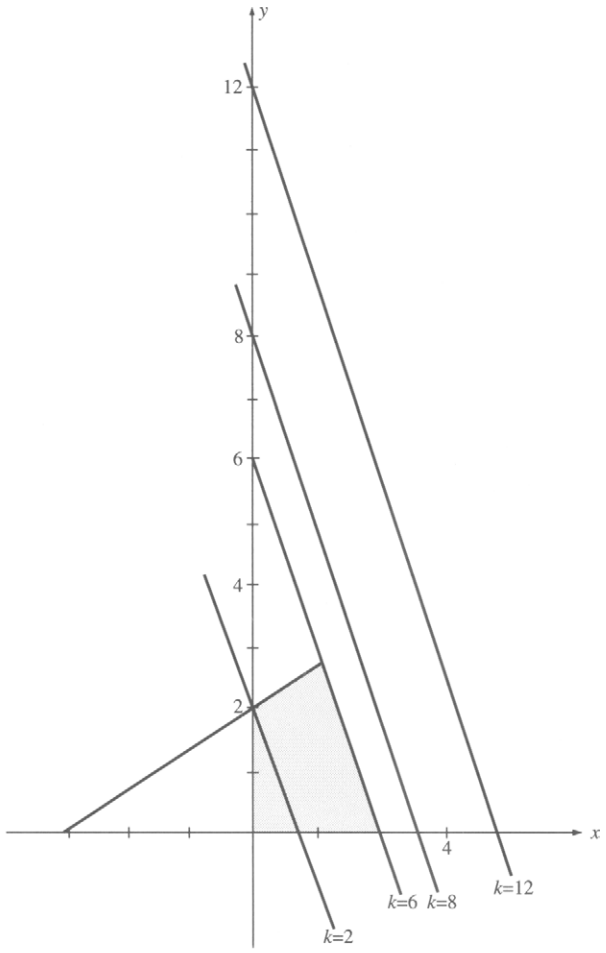
11.



13.



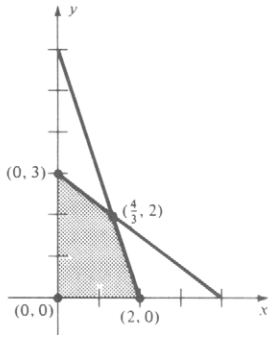
15.



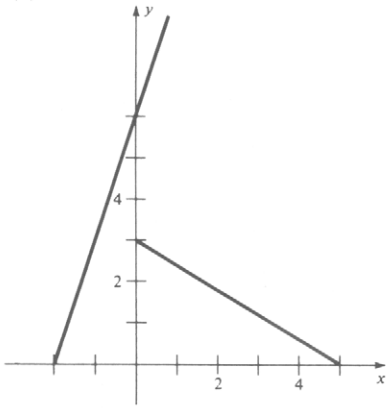
17. Not convex 19. Convex 21. Convex 23. Convex

Section 1.4, page 90

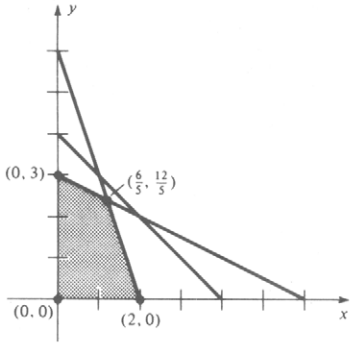
1. (a) $(0,0)$, $(2,0)$, $(\frac{4}{3},2)$, $(0,3)$ (b) $(0,3)$; $z = 6$



3. (a) None exists (b) Does not exist

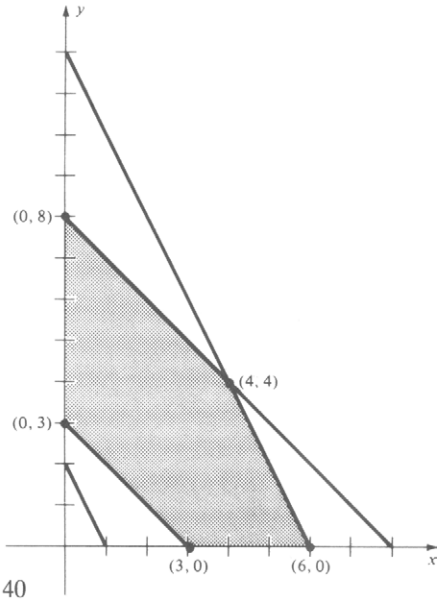


5. (a) $(0,0)$, $(2,0)$, $(\frac{6}{5}, \frac{12}{5})$, $(0,3)$



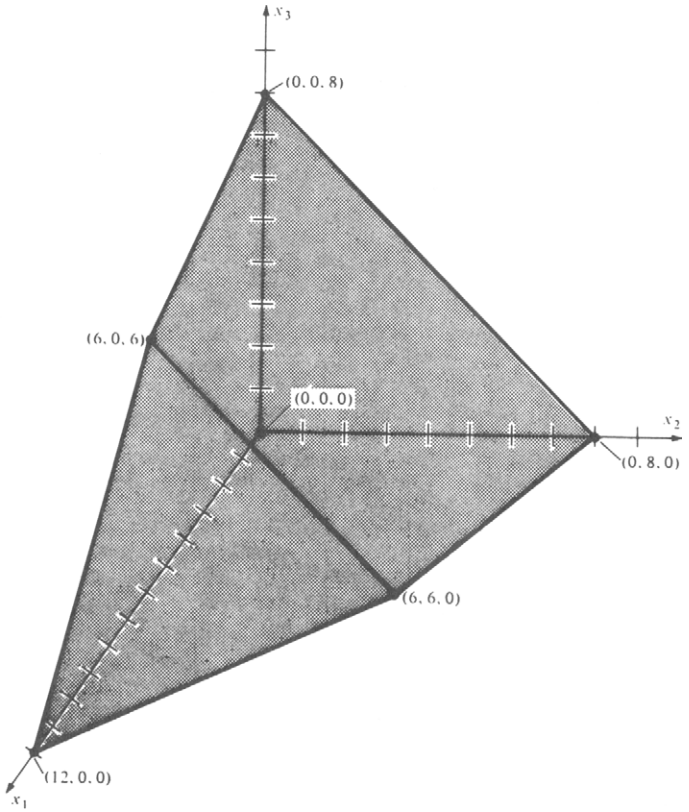
(b) $(0,0)$; $z = 0$

7. (a) $(3,0)$, $(6,0)$, $(4,4)$, $(0,8)$, $(0,3)$



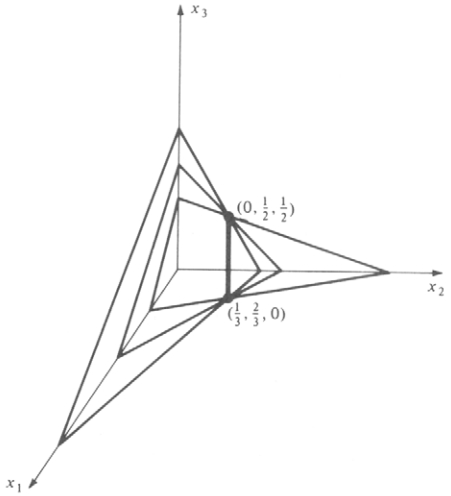
(b) $(0,8)$; $z = 40$

9. (a) $(0,0,0)$, $(0,0,8)$, $(6,0,6)$, $(12,0,0)$, $(6,6,0)$, $(0,8,0)$



(b) $(6,6,0)$; $z = 36$

11. (a) $(\frac{1}{3}, \frac{2}{3}, 0)$, $(0, \frac{1}{2}, \frac{1}{2})$



(b) $(\frac{1}{3}, \frac{2}{3}, 0)$; $z = 3$

Section 1.5, page 99

1. (i) (a), (c), (e) (ii) (a), (c) (iii) (a), (b), (c)
   (iv) (a) basic variables are x_2, x_4, x_5
   (c) basic variables are x_1, x_4, and one of x_2, x_3, x_5

3. Let x_1 = number of glazed doughnuts per day and
   x_2 = number of powdered doughnuts per day.

   Maximize $z = 0.07x_1 + 0.05x_2$
   subject to

$$\begin{aligned} x_1 + x_2 &\leq 1400 \\ x_1 &\leq 1000 \\ x_2 &\leq 1200 \\ x_1 &\geq 600 \\ x_1 \geq 0, \quad x_2 &\geq 0 \end{aligned}$$

   Extreme points: (600, 0), (1000, 0), (1000, 400) (600, 800)
   Optimal solution: (1000, 400); z = \$90

5. Let x_1 = number of glazed doughnuts per day
   x_2 = number of powdered doughtnuts per day, and
   x_3, x_4, x_5, x_6 = slack variables.

   Maximize $z = 0.07x_1 + 0.05x_2$
   subject to

$$\begin{aligned} x_1 + x_2 + x_3 &= 1400 \\ x_1 + x_4 &= 1000 \\ x_2 + x_5 &= 1200 \\ x_1 - x_6 &= 600 \end{aligned}$$

   At the optimal solution,

   $x_3 = 0$ = additional number of doughnuts per day that could be baked;
   $x_4 = 0$ = additional number of doughnuts per day that could be glazed;
   $x_5 = 800$ = additional number of doughnuts per day that could be dipped; and
   $x_6 = 400$ = number of glazed doughnuts over the required number.

   The basic variables are x_1, x_2, x_5, and x_6.

7. (a) Basic if x_2 and x_4 are taken as nonbasic variables; basic if x_1 and x_2 are taken as nonbasic variables; and not basic if x_1 and x_4 are taken as nonbasic variables
   (b) Not basic
   (c) Not basic

9. (a) Maximize $z = 4x_1 + 2x_2 + 7x_3$
subject to

$$\begin{aligned} 2x_1 - x_2 + 4x_3 + x_4 &= 18 \\ 4x_1 + 2x_2 + 5x_3 + x_5 &= 10 \\ x_j \geq 0, \quad j &= 1,2,\ldots,5 \end{aligned}$$

(b)

x_1	x_2	x_3	x_4	x_5	Basic variables	Optimal
0	0	0	18	10	x_4, x_5	No
0	0	2	10	0	x_3, x_4	Yes
0	5	0	23	0	x_2, x_4	No
$\frac{5}{2}$	0	0	13	0	x_1, x_4	No

CHAPTER 2

Section 2.1, page 119

1.

	x	y	u	v	
u	3	5	1	0	8
v	2	7	0	1	12
	-2	-5	0	0	0

3. (a) x_2 (b) x_1 (c) No finite optimal solution

5. Using x_2 as the entering variable,

	x_1	x_2	x_3	x_4	
x_2	$\frac{1}{2}$	1	0	$\frac{1}{2}$	$\frac{3}{2}$
x_3	$-\frac{1}{4}$	0	1	$-\frac{5}{4}$	$\frac{3}{4}$
	2	0	0	-2	$\frac{23}{2}$

Using x_4 as the entering variable,

	x_1	x_2	x_3	x_4	
x_4	1	2	0	1	3
x_3	1	$\frac{5}{2}$	1	0	$\frac{9}{2}$
	4	4	0	0	$\frac{35}{2}$

7.

	x_1	x_2	x_3	x_4	
x_1	1	1	5	0	4
x_4	0	1	7	1	10
	0	3	13	0	19

9. (a) $x_1 = 20, \quad x_2 = 0, \quad x_3 = 0, \quad u = 6, \quad v = 12, \quad w = 0$
Basic variables: x_1, u, v
(b)

	x_1	x_2	x_3	u	v	w	
x_1	1	0	-8	$-\frac{5}{2}$	0	13	5
x_2	0	1	2	$\frac{1}{2}$	0	-2	3
v	0	0	-5	-1	1	7	6
	0	0	7	$\frac{5}{2}$	0	-7	27

(c) $x_1 = 5, \quad x_2 = 3, \quad x_3 = 0, \quad u = 0, \quad v = 6, \quad w = 0$
Basic variables: x_1, x_2, v

11. $\begin{bmatrix} 0 \\ 0 \end{bmatrix}; \quad z = 0$

13. $x = \frac{38}{23}, \quad y^+ = \frac{12}{23}, \quad y^- = 0; \quad z = \frac{136}{23}$

15. Make 0 kg Super blend and $\frac{640}{3}$ kg Deluxe blend.
Profit = \$64.00

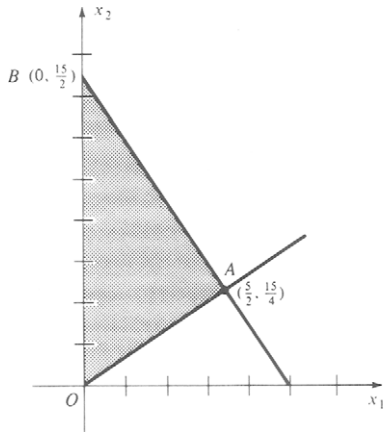
17. Make 0 books in either paperback or library bindings and 100 books in book-club binding. Profit = \$80.00

19. No finite optimal solution

21. $[3 \quad 0 \quad 0 \quad 0]^T; \quad z = 15$

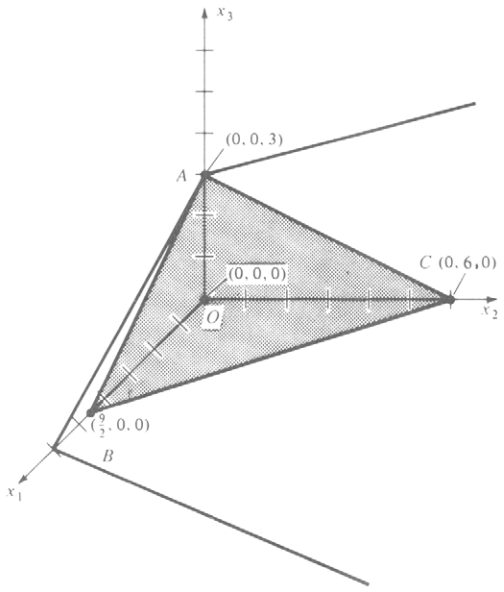
Section 2.2, page 130

1. $[0 \quad \frac{15}{2}]^T; \quad z = \frac{75}{2}$



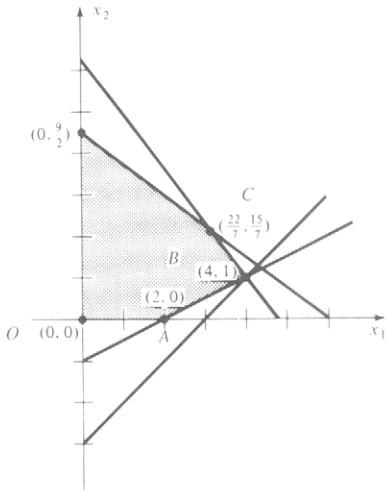
The simplex algorithm examines the following extreme points: O, O, A, B.

3. $[0 \quad 0 \quad 3]^T$; $z = 15$



The simplex algorithm examines the following extreme points: O, A, A.

5. $[\frac{22}{7} \quad \frac{15}{7}]^T$; $z = \frac{207}{7}$



The simplex algorithm examines the following extreme points: O, A, B, C.

9. $[4 \quad 1 \quad 0 \quad 4 \quad 1 \quad 0 \quad 0]^T$; $z = -2$

Section 2.3, page 150

1. (a)

	x_1	x_2	x_3	y_1	y_2	
y_1	1	2	7	1	0	4
y_2	1	3	1	0	1	5
	-2	-5	-8	0	0	-9

(b)

	x_1	x_2	x_3	y_1	y_2	
y_1	1	2	7	1	0	4
y_2	1	3	1	0	1	5
	$-1-2M$	$-5M$	$-3-8M$	0	0	$-9M$

3. (a)

	x_1	x_2	x_3	x_4	x_5	y_1	y_2	
y_1	1	3	2	-1	0	1	0	7
y_2	2	1	1	0	-1	0	1	4
	-3	-4	-3	1	1	0	0	-11

(b)

	x_1	x_2	x_3	x_4	x_5	y_1	y_2	
y_1	1	3	2	-1	0	1	0	7
y_2	2	1	1	0	-1	0	1	4
	$3-3M$	$-2-4M$	$-3M$	M	M	0	0	$-11M$

5.

	x_1	x_2	x_3	x_4	x_5	y_1	y_2	
x_2	$-\frac{1}{4}$	1	0	$-\frac{1}{4}$	$\frac{3}{4}$	$\frac{1}{4}$	$-\frac{3}{4}$	$\frac{1}{2}$
x_3	$\frac{3}{4}$	0	1	$-\frac{1}{4}$	$-\frac{1}{4}$	$\frac{1}{4}$	$-\frac{3}{4}$	$\frac{3}{2}$
	0	0	0	0	0	1	1	0

7. (a)

	x_1	x_2	x_3	x_4	x_5	
x_2	-1	1	$\frac{3}{10}$	0	$-\frac{1}{2}$	$\frac{3}{5}$
x_4	$\frac{1}{2}$	0	$\frac{1}{5}$	1	$-\frac{1}{10}$	$\frac{7}{10}$
	-3	0	$-\frac{7}{10}$	0	$-\frac{1}{2}$	$\frac{3}{5}$

(b) No finite optimal solution

9. An artificial variable y_1 has a nonzero value at the end of Phase 1. There are no feasible solutions.
11. Invest \$70,000 in bond AAA and \$30,000 in stock BB. Return = \$7600
13. No feasible solutions
15. Make no PEST and 3 kg BUG. Profit = \$7.50
17. No feasible solutions
19. Invest \$40,000 in the utilities stock, \$0 in the electronic stock, and \$160,000 in the bond. Return = \$11,600
21. No finite optimal solution
23. $[0 \quad 2 \quad 0 \quad 4 \quad 0 \quad 1]^T$; $z = 12$

CHAPTER 3

Section 3.1, page 165

1. Maximize $z' = 8w_1 + 12w_2 + 6w_3$
subject to
$w_1 + 2w_2 + 2w_3 \leq 3$
$4w_1 + 3w_2 + w_3 \leq 4$
$w_1 \geq 0, w_2 \geq 0, w_3 \geq 0$

3. Minimize $z' = 8w_1 + 7w_2 + 12w_3$
subject to
$3w_1 + 5w_2 + 4w_3 \geq 3$
$2w_1 + w_2 \geq 2$
$w_1 + 2w_2 + w_3 \geq 5$
$4w_2 - 2w_3 \geq 7$
$w_1 \geq 0, w_3 \geq 0, w_2$ unrestricted

5. Minimize $z' = 18w_1 + 12w_2$
subject to

$$3w_1 + 2w_2 \geq 3$$
$$3w_1 + 2w_2 = 1$$
$$w_1 + 4w_2 \geq 4$$
$$w_1 \geq 0, \quad w_2 \text{ unrestricted}$$

7. (a) Primal problem:

Maximize $z' = -2x_1 + 3x_2 - x_4$
subject to

$$\begin{aligned} x_1 + 2x_2 + x_3 &\leq 7 \\ x_1 + 4x_2 - x_4 &= 5 \\ -x_2 - x_3 - 5x_4 &\leq -3 \\ x_1 \geq 0, \quad x_2 \geq 0, \quad x_3 \geq 0, \quad x_4 &\geq 0 \end{aligned}$$

Dual problem:

Minimize $z'' = 7w_1 + 5w_2 - 3w_3$
subject to

$$\begin{aligned} w_1 + w_2 &\geq -2 \\ 2w_1 + 4w_2 - w_3 &\geq 3 \\ w_1 - w_3 &\geq 0 \\ -w_2 - 5w_3 &\geq -1 \\ w_1 \geq 0, \quad w_3 \geq 0, \quad w_2 &\text{ unrestricted} \end{aligned}$$

9. Minimize $z' = 12w_1 + 48w_2 + 360w_3$
subject to

$$\begin{aligned} w_1 + 6w_2 + 36w_3 &\geq 40 \\ w_1 + 6w_2 + 24w_3 &\geq 30 \\ w_1 + 2w_2 + 18w_3 &\geq 20 \\ w_1 \geq 0, \quad w_2 \geq 0, \quad w_3 &\geq 0, \end{aligned}$$

where w_1, w_2, and w_3 denote fictitious prices representing the contributions to profit of one unit of land, capital, and labor, respectively.

11. Minimize $z' = \mathbf{b}^T\mathbf{w}$
subject to

$$\begin{aligned} \mathbf{A}^T\mathbf{w} &\geq \mathbf{c} \\ \mathbf{B}^T\mathbf{w} &= \mathbf{d} \\ \mathbf{w} &\geq \mathbf{0} \end{aligned}$$

Section 3.2, page 182

1. The dual problem has an optimal feasible solution with objective function value 117.81. Moreover, the slack variables for the first four constraints of the dual problem must be zero at the optimal solution.
3. The dual problem has either no feasible solutions or feasible solutions with unbounded objective function values.
5. $z = \mathbf{c}^T\mathbf{x} = 139$

7. $[1 \quad \frac{1}{3} \quad 0]^T$; $z = 11$
   Dual problem:
   Minimize $z' = 2w_1 + 3w_2 + 4w_3$
   subject to

$$w_1 + 2w_2 + 3w_3 \geq 8$$
$$w_1 + 3w_2 + 3w_3 \geq 9$$
$$2w_1 + 4w_2 + w_3 \geq 5$$
$$w_1 \geq 0, \quad w_2 \geq 0, \quad w_3 \geq 0$$

   Solution: $[0 \quad 1 \quad 2]^T$; $z' = 11$

11. $[0 \quad 1 \quad 1]$; $z' = 20$

Section 3.3, page 202

1.

		−1	2	−6	0	0	5	
c_B		x_1	x_2	x_3	x_4	x_5	x_6	x_B
2	x_2	−3	1	−1	−2	0	0	2
5	x_6	2	0	0	−1	0	1	3
0	x_5	6	0	7	6	1	0	1
		5	0	4	−9	0	0	19

3.

		2	3	5	1	0	0	0	0	0	
c_B		x_1	x_2	x_3	x_4	x_5	x_6	x_7	y_1	y_2	x_B
2	x_1	1	$\frac{1}{2}$	1	0	3	0	−1	0	0	$\frac{1}{2}$
1	x_4	0	1	$-\frac{1}{2}$	1	−2	0	−1	0	0	4
0	x_6	0	2	$-\frac{2}{3}$	0	$-\frac{1}{2}$	1	0	0	0	$\frac{3}{2}$
0	y_1	0	$\frac{3}{2}$	−2	0	−3	0	0	1	0	0
0	y_2	0	4	−1	0	2	0	2	0	1	0
		0	−1	$-\frac{7}{2}$	0	4	0	−3	0	0	5

$$\left[\frac{1}{2} \quad 0 \quad 0 \quad 4 \quad 0 \quad \frac{3}{2} \quad 0\right]^T; \qquad z = 5$$

7. Any point on the line segment joining $[5 \quad 0 \quad 3 \quad 0]^T$ and $[0 \quad 0 \quad 8 \quad 0]^T$; $z = 8$

9. No feasible solution

11. $[0 \quad 2 \quad 0 \quad 4]^T$; $z = 6$

13. *Exercise 6*

First: $\mathbf{B} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$, $\mathbf{B}^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

Second: $\mathbf{B} = \begin{bmatrix} 3 & 0 & 0 \\ 5 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}$, $\mathbf{B}^{-1} = \begin{bmatrix} \frac{1}{3} & 0 & 0 \\ -\frac{5}{3} & 1 & 0 \\ -\frac{1}{3} & 0 & 1 \end{bmatrix}$

Third: $\mathbf{B} = \begin{bmatrix} 3 & -1 & 0 \\ 5 & 3 & 0 \\ 1 & 0 & 1 \end{bmatrix}$, $\mathbf{B}^{-1} = \begin{bmatrix} \frac{3}{14} & \frac{1}{14} & 0 \\ -\frac{5}{14} & \frac{3}{14} & 0 \\ -\frac{3}{14} & -\frac{1}{14} & 1 \end{bmatrix}$

Final: $\mathbf{B} = \begin{bmatrix} 3 & -1 & 2 \\ 5 & 3 & 1 \\ 1 & 0 & 2 \end{bmatrix}$, $\mathbf{B}^{-1} = \begin{bmatrix} \frac{2}{7} & \frac{2}{21} & -\frac{1}{3} \\ -\frac{3}{7} & \frac{4}{21} & \frac{1}{3} \\ -\frac{1}{7} & -\frac{1}{21} & \frac{2}{3} \end{bmatrix}$

Exercise 9

First: $\mathbf{B} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$, $\mathbf{B}^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$

Second: $\mathbf{B} = \begin{bmatrix} 1 & 3 & 0 & 0 \\ 0 & 4 & 0 & 0 \\ 0 & 2 & 1 & 0 \\ 0 & 3 & 0 & 1 \end{bmatrix}$, $\mathbf{B}^{-1} = \begin{bmatrix} 1 & -\frac{3}{4} & 0 & 0 \\ 0 & \frac{1}{4} & 0 & 0 \\ 0 & -\frac{1}{2} & 1 & 0 \\ 0 & -\frac{3}{4} & 0 & 1 \end{bmatrix}$

Third: $\mathbf{B} = \begin{bmatrix} 1 & 3 & 1 & 0 \\ 0 & 4 & 0 & 0 \\ 0 & 2 & 3 & 0 \\ 0 & 3 & 1 & 1 \end{bmatrix}$, $\mathbf{B}^{-1} = \begin{bmatrix} 1 & -\frac{7}{12} & -\frac{1}{3} & 0 \\ 0 & \frac{1}{4} & 0 & 0 \\ 0 & -\frac{1}{6} & \frac{1}{3} & 0 \\ 0 & -\frac{7}{12} & -\frac{1}{3} & 1 \end{bmatrix}$

Final: $\mathbf{B} = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & -1 & 3 & 0 \\ 0 & 4 & 1 & 1 \end{bmatrix}$, $\mathbf{B}^{-1} = \begin{bmatrix} 1 & -\frac{2}{3} & -\frac{1}{3} & 0 \\ 0 & \frac{1}{2} & 0 & 0 \\ 0 & \frac{1}{6} & \frac{1}{3} & 0 \\ 0 & -\frac{13}{6} & -\frac{1}{3} & 1 \end{bmatrix}$

15. Invest \$70,000 in bond AAA and \$30,000 in stock BB. Return = \$7600

17. Use only the large backhoe for $6\frac{2}{3}$ h. Cost = \$266.67

Section 3.4, page 214

1. $[4 \quad \frac{10}{3} \quad 2 \quad 0 \quad 0]^T$; $z = 40$

3. $[0 \quad \frac{5}{2} \quad 2 \quad 2 \quad \frac{5}{2} \quad 0 \quad 0]$; $z = \frac{37}{2}$

5. No feasible solutions

Section 3.5, page 223

1. $\begin{bmatrix} 1 & 0 & 0 & -\frac{1}{3} \\ 0 & 1 & 0 & \frac{2}{3} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & \frac{1}{3} \end{bmatrix}$

3. $\begin{bmatrix} 1 & 0 & -2 \\ -\frac{5}{2} & 1 & 0 \\ -2 & 2 & -3 \end{bmatrix}$

5. $[\frac{10}{3} \quad \frac{5}{3} \quad \frac{1}{3}]^T$; $z = \frac{28}{3}$

7. Optimal solution is unbounded.

9. $[0 \quad 0 \quad 10 \quad 0 \quad 0 \quad 0 \quad 18 \quad 0]^T$; $z = 66$

Section 3.6, page 233

1. (a) $-\infty < \Delta c_1 \leq \frac{7}{9}$
$-\frac{7}{6} \leq \Delta c_2 < \infty$
$-1 \leq \Delta c_3 \leq 5$
$-\infty < \Delta c_4 \leq \frac{14}{9}$

(b) $-4 \leq \Delta b_1 \leq 17$
$-12 \leq \Delta b_2 \leq 12$
$-\frac{34}{3} \leq \Delta b_3 < \infty$

3. (a) $-2 \leq \Delta c_1 < \infty$
$-\infty < \Delta c_2 \leq 4$
$-1 \leq \Delta c_3 < \infty$
$-\infty < \Delta c_4 \leq \frac{1}{2}$
$-\infty < \Delta c_5 \leq 4$
$-1 \leq \Delta c_6 < \infty$

(b) $-4 \leq \Delta b_1 \leq 12$
$-9 \leq \Delta b_2 < \infty$
$-12 \leq \Delta b_3 \leq 12$

5. (a) Plant $\frac{15}{2}$ acres of corn, $\frac{3}{2}$ acres of soybeans, and 3 acres of oats.
Profit = \$405
(b) Plant seven acres of corn, three acres of oats, no soybeans.
Profit = \$340
(c) Plant six acres of corn, six acres of oats, no soybeans.
Profit = \$366
(d) At least \$10/acre

CHAPTER 4

Section 4.1, page 259

1. Let x_1 = number of type A and
x_2 = number of type B

Minimize $z = 22{,}000x_1 + 48{,}000x_2$
subject to

$$100x_1 + 200x_2 \geq 600$$
$$50x_1 + 140x_2 \leq 350$$
$$x_1 \geq 0, \quad x_2 \geq 0, \text{ integers.}$$

3. Let $x_i = \begin{cases} 1 & \text{if the } i\text{th CD is purchased} \\ 0 & \text{otherwise} \end{cases}$

$$a_{ij} = \begin{cases} 1 & \text{if the } j\text{th song is on the } i\text{th CD} \\ 0 & \text{otherwise} \end{cases}$$

Minimize $z = \sum_{i=1}^{10} c_i x_i$

subject to

$$\sum_{i=1}^{10} a_{ij} x_i \geq 1$$
$$j = 1, 2, \ldots, 6$$

5. Let w_i = person-weeks necessary for project i
c_i = cost of project i (in thousands of dollars)
v_i = value of completion of project i

$$x_i = \begin{cases} 1 & \text{if project } i \text{ is to be completed} \\ 0 & \text{otherwise} \end{cases}$$

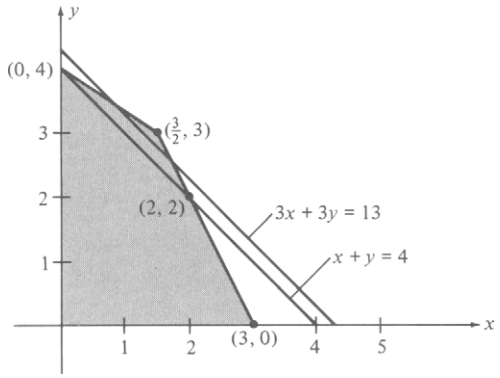
Maximize $z = \sum_{i=1}^{10} v_i x_i$

subject to

$$\sum_{i=1}^{10} w_i x_i \leq 1000$$
$$\sum_{i=1}^{10} c_i x_i \leq 1500.$$

Section 4.2, page 274

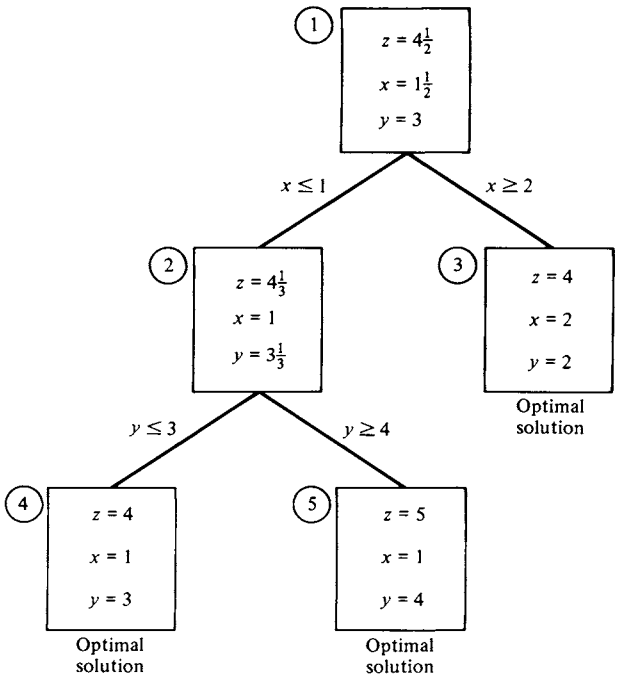
1. $-\frac{1}{2}x_3 + u_1 = -\frac{7}{8}$
3. $x = 2, \quad y = 2; \quad z = 4$



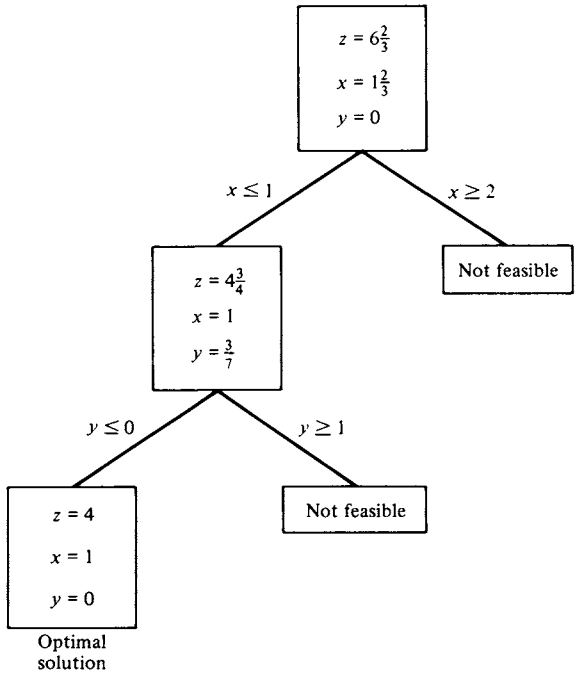
5. $x = 1, \quad y = 0; \quad z = 4$
7. $x = 8, \quad y = 2; \quad z = 44$
9. $x = 1, \quad y = \frac{10}{3}; \quad z = \frac{13}{3}$
11. $[0 \quad 4 \quad \frac{4}{3} \quad 0]^T$; $z = \frac{28}{3}$

Section 4.3, page 289

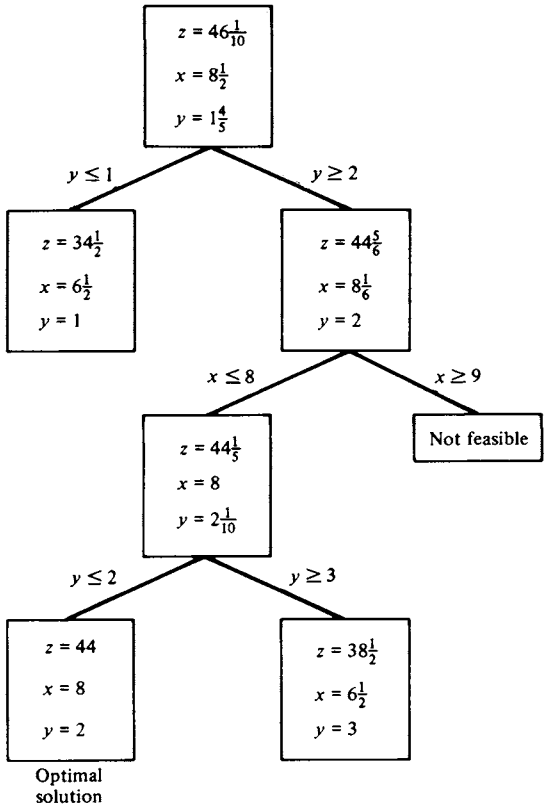
1.



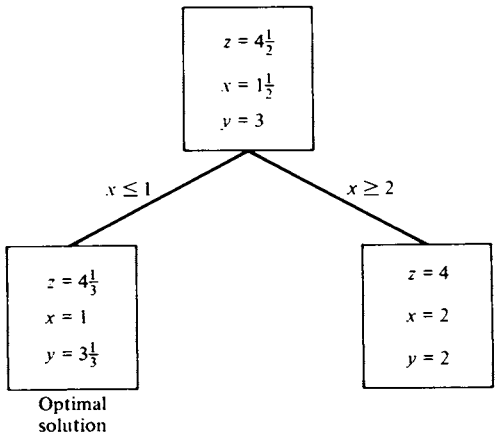
3.



5.


$z = 46\frac{1}{10}$
$x = 8\frac{1}{2}$
$y = 1\frac{4}{5}$
$y \leq 1$
$y \geq 2$
$z = 34\frac{1}{2}$
$x = 6\frac{1}{2}$
$y = 1$
$z = 44\frac{5}{6}$
$x = 8\frac{1}{6}$
$y = 2$
$x \leq 8$
$x \geq 9$
$z = 44\frac{1}{5}$
$x = 8$
$y = 2\frac{1}{10}$
Not feasible
$y \leq 2$
$y \geq 3$
$z = 44$
$x = 8$
$y = 2$
$z = 38\frac{1}{2}$
$x = 6\frac{1}{2}$
$y = 3$
Optimal solution

7.


$z = 4\frac{1}{2}$
$x = 1\frac{1}{2}$
$y = 3$
$x \leq 1$
$x \geq 2$
$z = 4\frac{1}{3}$
$x = 1$
$y = 3\frac{1}{3}$
$z = 4$
$x = 2$
$y = 2$
Optimal solution

9.

$$z = 9\tfrac{1}{3}$$
$$x_1 = x_4 = 0$$
$$x_2 = 4$$
$$x_3 = 1\tfrac{1}{3}$$

Optimal solution

11. Buy six machines of type A and no machines of type B.
Cost = \$132,000

13. Buy two bottles each of cola and ginger ale, five bottles of root beer, and three bottles of grape. Cost = \$7.28

CHAPTER 5

Section 5.1, page 324

3. (a)

				90
		70		
50			60	
50	40	30		30

(b)

				90
			40	30
100			10	
	40	100	10	

(c)

				90
			40	30
100			10	
	40	100	10	

5.

			30	70
20	60	80		
30			70	

$z = \$1730$

7.

				90
			40	30
100			10	
	40	100	10	

$z = \$950$

9.

10	70		
10		50	
50			
		20	80

or

60	20		
10		50	
	50		
		20	80

$z = \$940$

11.

	1		
1			
			1
		1	

$z = 12$

13.

20	20	20	40
	60		
		50	
40			

← Dummy supply

or

20		40	40
	60		
	20	30	
40			

← Dummy supply

$z = \$800$

Section 5.2, page 338

1. $\begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$; $z = 9$

3. $\begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}$; $z = 12$

5. $\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}$; $z = 14$

7. One of the possible answers is
   $702 \to 706$
   $705 \to 707$
   $708 \to 714$
   $709 \to 712$
   $713 \to 715$

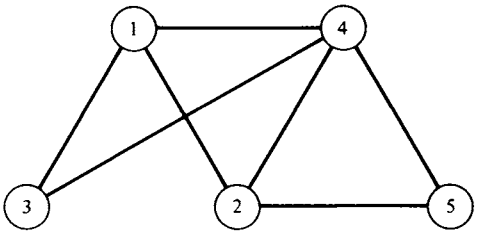
Section 5.3, page 344

1. (a) $\begin{bmatrix} 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 \end{bmatrix}$

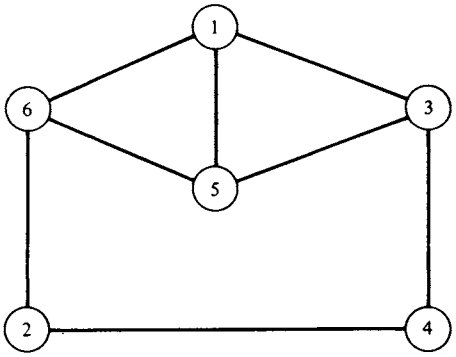
   (b) (i) $(1,4)$
   (ii) $(1,2)$, $(2,4)$
   (iii) $(1,2)$, $(2,6)$, $(6,5)$, $(5,3)$, $(3,2)$, $(2,4)$

   (c) (i) $(2,3)$, $(3,2)$
   (ii) $(2,1)$, $(1,4)$, $(4,2)$

3. (a)



(b)



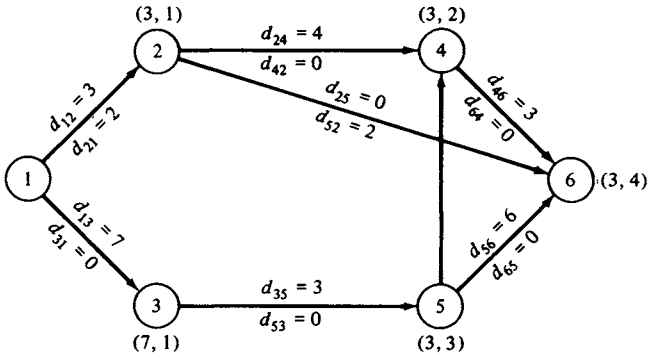
5. (a) $\begin{bmatrix} 0 & 10 & 0 & 20 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 9 & 0 & 0 & 0 \\ 0 & 12 & 0 & 0 & 5 \\ 0 & 0 & 17 & 0 & 0 \end{bmatrix}$

(b) 1

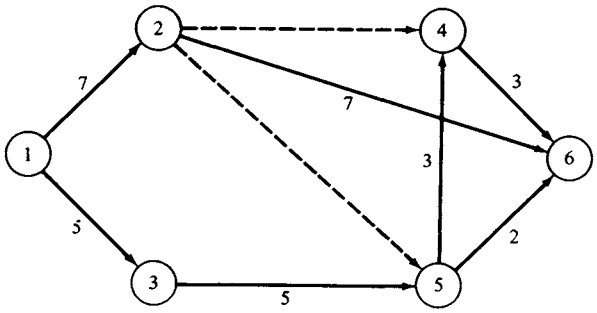
(c) 2

Section 5.4, page 363

1.

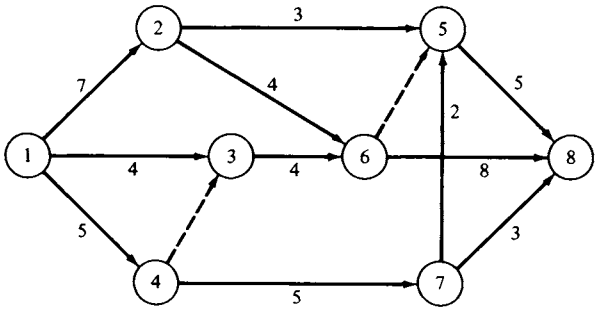


3.



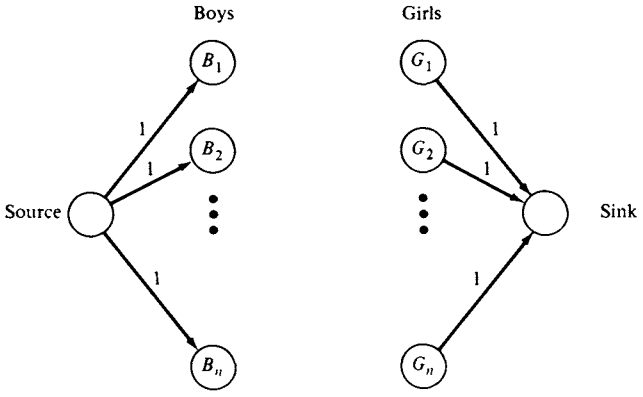
Maximum flow = 12

5.



Maximum flow = 16

7. (a)



Connect B_i to G_j with an arc of capacity 1 if boy i and girl j are friends.

(b)
$$\begin{bmatrix} 1 & \textcircled{1} & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & \textcircled{1} \\ \textcircled{1} & 0 & 1 & 0 & 0 \\ 0 & 0 & \textcircled{1} & 0 & 1 \\ 0 & 1 & 0 & \textcircled{1} & 0 \end{bmatrix}$$

(c)
$$\begin{bmatrix} 0 & 1 & 0 & \textcircled{1} & 0 \\ \textcircled{1} & 0 & 0 & 1 & 1 \\ 0 & 0 & \textcircled{1} & 0 & 1 \\ 0 & 0 & 0 & 1 & \textcircled{1} \\ 0 & \textcircled{1} & 0 & 0 & 0 \end{bmatrix}$$

(d) n lines are necessary to cover the 1s.

Section 5.5, page 375

1. Path: $1 \rightarrow 5 \rightarrow 7$; Length = 8

3. Path: $1 \rightarrow 2 \rightarrow 4 \rightarrow 9$; Length = 14

5. Replace the equipment at the beginning of the fourth year. Total equipment cost = \$380,000

Section 5.6, page 385

1. (a) and (b)

Node	*Early event times*	*Late event times*
1	0	0
2	3	8
3	6	6
4	8	8
5	5	10
6	9	14
7	16	20
8	9	9
9	7	12
10	18	18
11	17	22
12	25	25

(c) $1 \rightarrow 4 \rightarrow 8 \rightarrow 10 \rightarrow 12$ or $1 \rightarrow 3 \rightarrow 8 \rightarrow 10 \rightarrow 12$

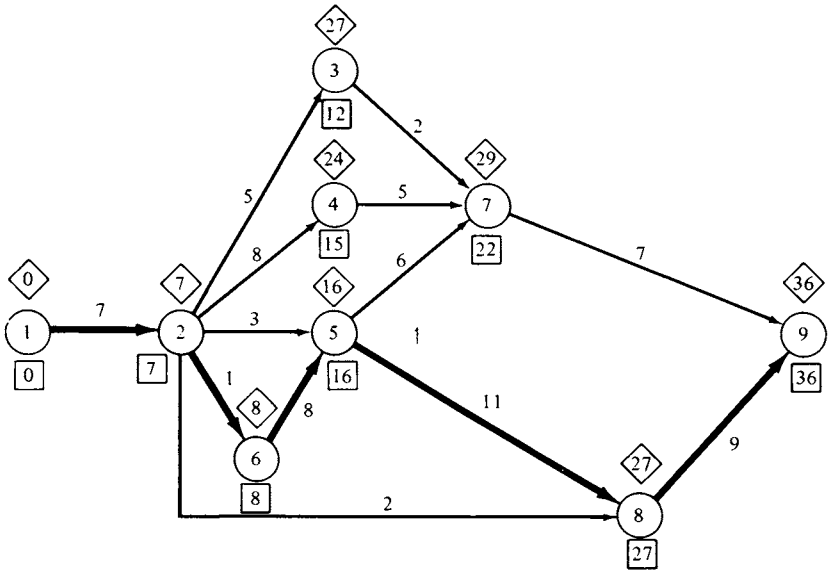
3. (a) and (b)

Node	*Early event times*	*Late event times*
1	0	0
2	3	3
3	8	8
4	7	14
5	7	21
6	22	22
7	9	23
8	14	14
9	22	22
10	15	29
11	15	29
12	31	31

(c) $1 \rightarrow 2 \rightarrow 3 \rightarrow 8 \rightarrow 6 \rightarrow 9 \rightarrow 12$

5. Insert dummy activity between node 2 and node 5.

7.



Critical path: $1 \rightarrow 2 \rightarrow 6 \rightarrow 5 \rightarrow 8 \rightarrow 9$

Section 5.7, page 393

3.

All remaining arcs should be labeled $[c, 1, 0]$, where c is the length of the arc.

Index

Selected chapters from draft of

An Introduction to Game Theory
by
Martin J. Osborne

Please send comments to
Martin J. Osborne
Department of Economics
150 St. George Street
University of Toronto
Toronto, Canada M5S 3G7
email: martin.osborne@utoronto.ca

This version: 2000/11/6

1 Introduction

1.1 What is game theory?

GAME THEORY aims to help us understand situations in which decision-makers interact. A game in the everyday sense—"a competitive activity ... in which players contend with each other according to a set of rules", in the words of my dictionary—is an example of such a situation, but the scope of game theory is vastly larger. Indeed, I devote very little space to games in the everyday sense; my main focus is the use of game theory to illuminate economic, political, and biological phenomena.

A list of some of the applications I discuss will give you an idea of the range of situations to which game theory can be applied: firms competing for business, political candidates competing for votes, jury members deciding on a verdict, animals fighting over prey, bidders competing in an auction, the evolution of siblings' behavior towards each other, competing experts' incentives to provide correct diagnoses, legislators' voting behavior under pressure from interest groups, and the role of threats and punishment in long-term relationships.

Like other sciences, game theory consists of a collection of models. A model is an abstraction we use to understand our observations and experiences. What "understanding" entails is not clear-cut. Partly, at least, it entails our perceiving relationships between situations, isolating principles that apply to a range of problems, so that we can fit into our thinking new situations that we encounter. For example, we may fit our observation of the path taken by a lobbed tennis ball into a model that assumes the ball moves forward at a constant velocity and is pulled towards the ground by the constant force of "gravity". This model enhances our understanding because it fits well no matter how hard or in which direction the ball is hit, and applies also to the paths taken by baseballs, cricket balls, and a wide variety of other missiles, launched in any direction.

A model is unlikely to help us understand a phenomenon if its assumptions are wildly at odds with our observations. At the same time, a model derives power from its simplicity; the assumptions upon which it rests should capture the essence

of the situation, not irrelevant details. For example, when considering the path taken by a lobbed tennis ball we should ignore the dependence of the force of gravity on the distance of the ball from the surface of the earth.

Models cannot be judged by an absolute criterion: they are neither "right" nor "wrong". Whether a model is useful or not depends, in part, on the purpose for which we use it. For example, when I determine the shortest route from Florence to Venice, I do not worry about the projection of the map I am using; I work under the assumption that the earth is flat. When I determine the shortest route from Beijing to Havana, however, I pay close attention to the projection—I assume that the earth is spherical. And were I to climb the Matterhorn I would assume that the earth is neither flat nor spherical!

One reason for improving our understanding of the world is to enhance our ability to mold it to our desires. The understanding that game theoretic models give is particularly relevant in the social, political, and economic arenas. Studying game theoretic models (or other models that apply to human interaction) may also suggest ways in which our behavior may be modified to improve our own welfare. By analyzing the incentives faced by negotiators locked in battle, for example, we may see the advantages and disadvantages of various strategies.

The models of game theory are precise expressions of ideas that can be presented verbally. However, verbal descriptions tend to be long and imprecise; in the interest of conciseness and precision, I frequently use mathematical symbols when describing models. Although I use the language of mathematics, I use few of its concepts; the ones I use are described in Chapter 17. My aim is to take advantage of the precision and conciseness of a mathematical formulation without losing sight of the underlying ideas.

Game-theoretic modeling starts with an idea related to some aspect of the interaction of decision-makers. We express this idea precisely in a model, incorporating features of the situation that appear to be relevant. This step is an art. We wish to put enough ingredients into the model to obtain nontrivial insights, but not so many that we are lead into irrelevant complications; we wish to lay bare the underlying structure of the situation as opposed to describe its every detail. The next step is to analyze the model—to discover its implications. At this stage we need to adhere to the rigors of logic; we must not introduce extraneous considerations absent from the model. Our analysis may yield results that confirm our idea, or that suggest it is wrong. If it is wrong, the analysis should help us to understand why it is wrong. We may see that an assumption is inappropriate, or that an important element is missing from the model; we may conclude that our idea is invalid, or that we need to investigate it further by studying a different model. Thus, the interaction between our ideas and models designed to shed light on them runs in two directions: the implications of models help us determine whether our ideas make sense, and these ideas, in the light of the implications of the models, may show us how the assumptions of our models are inappropriate. In either case, the process of formulating and analyzing a model should improve our understanding of the situation we are considering.

An outline of the history of game theory

Some game-theoretic ideas can be traced to the 18th century, but the major development of the theory began in the 1920s with the work of the mathematician Emile Borel (1871–1956) and the polymath John von Neumann (1903–57). A decisive event in the development of the theory was the publication in 1944 of the book *Theory of games and economic behavior* by von Neumann and Oskar Morgenstern. In the 1950s game-theoretic models began to be used in economic theory and political science, and psychologists began studying how human subjects behave in experimental games. In the 1970s game theory was first used as a tool in evolutionary biology. Subsequently, game theoretic methods have come to dominate microeconomic theory and are used also in many other fields of economics and a wide range of other social and behavioral sciences. The 1994 Nobel prize in economics was awarded to the game theorists John C. Harsanyi (1920–2000), John F. Nash (1928–), and Reinhard Selten (1930–).

John von Neumann

John von Neumann, the most important figure in the early development of game theory, was born in Budapest, Hungary, in 1903. He displayed exceptional mathematical ability as a child (he had mastered calculus by the age of 8), but his father, concerned about his son's financial prospects, did not want him to become a mathematician. As a compromise he enrolled in mathematics at the University of Budapest in 1921, but immediately left to study chemistry, first at the University of Berlin and subsequently at the Swiss Federal Institute of Technology in Zurich, from which he earned a degree in chemical engineering in 1925. During his time in Germany and Switzerland he returned to Budapest to write examinations, and in 1926 obtained a PhD in mathematics from the University of Budapest. He taught in Berlin and Hamburg, and, from 1930 to 1933, at Princeton University. In 1933 he became the youngest of the first six professors of the School of Mathematics at the Institute for Advanced Study in Princeton (Einstein was another).

Von Neumann's first published scientific paper appeared in 1922, when he was 19 years old. In 1928 he published a paper that establishes a key result on strictly competitive games (a result that had eluded Borel). He made many major contributions in pure and applied mathematics and in physics—enough, according to Halmos (1973), "for about three ordinary careers, in pure mathematics alone". While at the Institute for Advanced Study he collaborated with the Princeton economist Oskar Morgenstern in writing *Theory of games and economic behavior*, the book that established game theory as a field. In the 1940s he became increasingly involved in applied work. In 1943 he became a consultant to the Manhattan project, which was developing an atomic bomb. In 1944 he became involved with the development of the first electronic computer, to which he made major contributions. He

stayed at Princeton until 1954, when he became a member of the US Atomic Energy Commission. He died in 1957.

1.2 The theory of rational choice

The theory of rational choice is a component of many models in game theory. Briefly, this theory is that a decision-maker chooses the best action according to her preferences, among all the actions available to her. No qualitative restriction is placed on the decision-maker's preferences; her "rationality" lies in the consistency of her decisions when faced with different sets of available actions, not in the nature of her likes and dislikes.

1.2.1 Actions

The theory is based on a model with two components: a set A consisting of all the actions that, under some circumstances, are available to the decision-maker, and a specification of the decision-maker's preferences. In any given situation the decision-maker is faced with a subset[1] of A, from which she must choose a single element. The decision-maker knows this subset of available choices, and takes it as given; in particular, the subset is not influenced by the decision-maker's preferences. The set A could, for example, be the set of bundles of goods that the decision-maker can possibly consume; given her income at any time, she is restricted to choose from the subset of A containing the bundles she can afford.

1.2.2 Preferences and payoff functions

As to preferences, we assume that the decision-maker, when presented with any pair of actions, knows which of the pair she prefers, or knows that she regards both actions as equally desirable (is "indifferent between the actions"). We assume further that these preferences are consistent in the sense that if the decision-maker prefers the action a to the action b, and the action b to the action c, then she prefers the action a to the action c. No other restriction is imposed on preferences. In particular, we do not rule out the possibility that a person's preferences are altruistic in the sense that how much she likes an outcome depends on some other person's welfare. Theories that use the model of rational choice aim to derive implications that do not depend on any qualitative characteristic of preferences.

How can we describe a decision-maker's preferences? One way is to specify, for each possible pair of actions, the action the decision-maker prefers, or to note that the decision-maker is indifferent between the actions. Alternatively we can "represent" the preferences by a *payoff function*, which associates a number with each action in such a way that actions with higher numbers are preferred. More

[1] See Chapter 17 for a description of mathematical terminology.

precisely, the payoff function u represents a decision-maker's preferences if, for any actions a in A and b in A,

$$u(a) > u(b) \text{ if and only if the decision-maker prefers } a \text{ to } b. \tag{5.1}$$

(A better name than payoff function might be "preference indicator function"; in economic theory a payoff function that represents a consumer's preferences is often referred to as a "utility function".)

◆ EXAMPLE 5.2 (Payoff function representing preferences) A person is faced with the choice of three vacation packages, to Havana, Paris, and Venice. She prefers the package to Havana to the other two, which she regards as equivalent. Her preferences between the three packages are represented by any payoff function that assigns the same number to both Paris and Venice and a higher number to Havana. For example, we can set $u(\text{Havana}) = 1$ and $u(\text{Paris}) = u(\text{Venice}) = 0$, or $u(\text{Havana}) = 10$ and $u(\text{Paris}) = u(\text{Venice}) = 1$, or $u(\text{Havana}) = 0$ and $u(\text{Paris}) = u(\text{Venice}) = -2$.

? EXERCISE 5.3 (Altruistic preferences) Person 1 cares both about her income and about person 2's income. Precisely, the value she attaches to each unit of her own income is the same as the value she attaches to any two units of person 2's income. How do her preferences order the outcomes $(1, 4)$, $(2, 1)$, and $(3, 0)$, where the first component in each case is person 1's income and the second component is person 2's income? Give a payoff function consistent with these preferences.

A decision-maker's preferences, in the sense used here, convey only *ordinal* information. They may tell us that the decision-maker prefers the action a to the action b to the action c, for example, but they do not tell us "how much" she prefers a to b, or whether she prefers a to b "more" than she prefers b to c. Consequently a payoff function that represents a decision-maker's preferences also conveys only ordinal information. It may be tempting to think that the payoff numbers attached to actions by a payoff function convey intensity of preference—that if, for example, a decision-maker's preferences are represented by a payoff function u for which $u(a) = 0$, $u(b) = 1$, and $u(c) = 100$, then the decision-maker likes c a lot more than b but finds little difference between a and b. *But a payoff function contains no such information!* The *only* conclusion we can draw from the fact that $u(a) = 0$, $u(b) = 1$, and $u(c) = 100$ is that the decision-maker prefers c to b to a; her preferences are represented equally well by the payoff function v for which $v(a) = 0$, $v(b) = 100$, and $v(c) = 101$, for example, or any other function w for which $w(a) < w(b) < w(c)$.

From this discussion we see that a decision-maker's preferences are represented by many different payoff functions. Looking at the condition (5.1) under which the payoff function u represents a decision-maker's preferences, we see that if u represents a decision-maker's preferences and the payoff function v assigns a higher number to the action a than to the action b if and only if the payoff function u does

so, then v also represents these preferences. Stated more compactly, if u represents a decision-maker's preferences and v is another payoff function for which

$$v(a) > v(b) \text{ if and only if } u(a) > u(b)$$

then v also represents the decision-maker's preferences. Or, more succinctly, if u represents a decision-maker's preferences then any increasing function of u also represents these preferences.

? EXERCISE 6.1 (Alternative representations of preferences) A decision-maker's preferences over the set $A = \{a, b, c\}$ are represented by the payoff function u for which $u(a) = 0$, $u(b) = 1$, and $u(c) = 4$. Are they also represented by the function v for which $v(a) = -1$, $v(b) = 0$, and $v(c) = 2$? How about the function w for which $w(a) = w(b) = 0$ and $w(c) = 8$?

Sometimes it is natural to formulate a model in terms of preferences and then find payoff functions that represent these preferences. In other cases it is natural to start with payoff functions, even if the analysis depends only on the underlying preferences, not on the specific representation we choose.

1.2.3 The theory of rational choice

The theory of rational choice is that in any given situation the decision-maker chooses the member of the available subset of A that is best according to her preferences. Allowing for the possibility that there are several equally attractive best actions, **the theory of rational choice** is:

> *the action chosen by a decision-maker is at least as good, according to her preferences, as every other available action.*

For any action, we can design preferences with the property that no other action is preferred. Thus if we have no information about a decision-maker's preferences, and make no assumptions about their character, any *single* action is consistent with the theory. However, if we assume that a decision-maker who is indifferent between two actions sometimes chooses one action and sometimes the other, not every *collection* of choices for different sets of available actions is consistent with the theory. Suppose, for example, we observe that a decision-maker chooses a whenever she faces the set $\{a, b\}$, but sometimes chooses b when facing the set $\{a, b, c\}$. The fact that she always chooses a when faced with $\{a, b\}$ means that she prefers a to b (if she were indifferent then she would sometimes choose b). But then when she faces the set $\{a, b, c\}$ she must choose either a or c, never b. Thus her choices are inconsistent with the theory. (More concretely, if you choose the same dish from the menu of your favorite lunch spot whenever there are no specials then, regardless of your preferences, it is inconsistent for you to choose some other item *from the menu* on a day when there is an off-menu special.)

If you have studied the standard economic theories of the consumer and the firm, you have encountered the theory of rational choice before. In the economic

theory of the consumer, for example, the set of available actions is the set of all bundles of goods that the consumer can afford. In the theory of the firm, the set of available actions is the set of all input-output vectors, and the action a is preferred to the action b if and only if a yields a higher profit than does b.

1.2.4 Discussion

The theory of rational choice is enormously successful; it is a component of countless models that enhance our understanding of social phenomena. It pervades economic theory to such an extent that arguments are classified as "economic" as much because they apply the theory of rational choice as because they involve particularly "economic" variables.

Nevertheless, under some circumstances its implications are at variance with observations of human decision-making. To take a small example, adding an undesirable action to a set of actions sometimes significantly changes the action chosen (see Rabin 1998, 38). The significance of such discordance with the theory depends upon the phenomenon being studied. If we are considering how the markup of price over cost in an industry depends on the number of firms, for example, this sort of weakness in the theory may be unimportant. But if we are studying how advertising, designed specifically to influence peoples' preferences, affects consumers' choices, then the inadequacies of the model of rational choice may be crucial.

No general theory currently challenges the supremacy of rational choice theory. But you should bear in mind as you read this book that the model of choice that underlies most of the theories has its limits; some of the phenomena that you may think of explaining using a game theoretic model may lie beyond these limits. As always, the proof of the pudding is in the eating: if a model enhances our understanding of the world, then it serves its purpose.

1.3 Coming attractions

Part I presents the main models in game theory: a strategic game, an extensive game, and a coalitional game. These models differ in two dimensions. A strategic game and an extensive game focus on the actions of individuals, whereas a coalitional game focuses on the outcomes that can be achieved by groups of individuals; a strategic game and a coalitional game consider situations in which actions are chosen once and for all, whereas an extensive game allows for the possibility that plans may be revised as they are carried out.

The model, consisting of actions and preferences, to which rational choice theory is applied is tailor-made for the theory; if we want to develop another theory, we need to add elements to the model in addition to actions and preferences. The same is not true of most models in game theory: strategic interaction is sufficiently complex that even a relatively simple model can admit more than one theory of the outcome. We refer to a theory that specifies a set of outcomes for a model as a

2 Nash Equilibrium: Theory

Prerequisite: Chapter 1.

2.1 Strategic games

A STRATEGIC GAME is a model of interacting decision-makers. In recognition of the interaction, we refer to the decision-makers as *players*. Each player has a set of possible *actions*. The model captures interaction between the players by allowing each player to be affected by the actions of *all* players, not only her own action. Specifically, each player has *preferences* about the action *profile*—the list of all the players' actions. (See Section 17.5, in the mathematical appendix, for a discussion of profiles.)

More precisely, a strategic game is defined as follows. (The qualification "with ordinal preferences" distinguishes this notion of a strategic game from a more general notion studied in Chapter 4.)

▶ DEFINITION 11.1 (*Strategic game with ordinal preferences*) A **strategic game** (with ordinal preferences) consists of

- a set of **players**
- for each player, a set of **actions**
- for each player, **preferences** over the set of action profiles.

A very wide range of situations may be modeled as strategic games. For example, the players may be firms, the actions prices, and the preferences a reflection of the firms' profits. Or the players may be candidates for political office, the actions

campaign expenditures, and the preferences a reflection of the candidates' probabilities of winning. Or the players may be animals fighting over some prey, the actions concession times, and the preferences a reflection of whether an animal wins or loses. In this chapter I describe some simple games designed to capture fundamental conflicts present in a variety of situations. The next chapter is devoted to more detailed applications to specific phenomena.

As in the model of rational choice by a single decision-maker (Section 1.2), it is frequently convenient to specify the players' preferences by giving *payoff functions* that represent them. Bear in mind that these payoffs have only *ordinal* significance. If a player's payoffs to the action profiles a, b, and c are 1, 2, and 10, for example, the only conclusion we can draw is that the player prefers c to b and b to a; the numbers do *not* imply that the player's preference between c and b is stronger than her preference between a and b.

Time is absent from the model. The idea is that each player chooses her action once and for all, and the players choose their actions "simultaneously" in the sense that no player is informed, when she chooses her action, of the action chosen by any other player. (For this reason, a strategic game is sometimes referred to as a "simultaneous move game".) Nevertheless, an action may involve activities that extend over time, and may take into account an unlimited number of contingencies. An action might specify, for example, "if company X's stock falls below \$10, buy 100 shares; otherwise, do not buy any shares". (For this reason, an action is sometimes called a "strategy".) However, the fact that time is absent from the model means that when analyzing a situation as a strategic game, we abstract from the complications that may arise if a player is allowed to change her plan as events unfold: we assume that actions are chosen once and for all.

2.2 Example: the Prisoner's Dilemma

One of the most well-known strategic games is the *Prisoner's Dilemma*. Its name comes from a story involving suspects in a crime; its importance comes from the huge variety of situations in which the participants face incentives similar to those faced by the suspects in the story.

◆ EXAMPLE 12.1 (Prisoner's Dilemma) Two suspects in a major crime are held in separate cells. There is enough evidence to convict each of them of a minor offense, but not enough evidence to convict either of them of the major crime unless one of them acts as an informer against the other (finks). If they both stay quiet, each will be convicted of the minor offense and spend one year in prison. If one and only one of them finks, she will be freed and used as a witness against the other, who will spend four years in prison. If they both fink, each will spend three years in prison.

This situation may be modeled as a strategic game:

Players The two suspects.

Actions Each player's set of actions is $\{Quiet, Fink\}$.

Preferences Suspect 1's ordering of the action profiles, from best to worst, is $(Fink, Quiet)$ (she finks and suspect 2 remains quiet, so she is freed), $(Quiet, Quiet)$ (she gets one year in prison), $(Fink, Fink)$ (she gets three years in prison), $(Quiet, Fink)$ (she gets four years in prison). Suspect 2's ordering is $(Quiet, Fink)$, $(Quiet, Quiet)$, $(Fink, Fink)$, $(Fink, Quiet)$.

We can represent the game compactly in a table. First choose payoff functions that represent the suspects' preference orderings. For suspect 1 we need a function u_1 for which

$$u_1(Fink, Quiet) > u_1(Quiet, Quiet) > u_1(Fink, Fink) > u_1(Quiet, Fink).$$

A simple specification is $u_1(Fink, Quiet) = 3$, $u_1(Quiet, Quiet) = 2$, $u_1(Fink, Fink) = 1$, and $u_1(Quiet, Fink) = 0$. For suspect 2 we can similarly choose the function u_2 for which $u_2(Quiet, Fink) = 3$, $u_2(Quiet, Quiet) = 2$, $u_2(Fink, Fink) = 1$, and $u_2(Fink, Quiet) = 0$. Using these representations, the game is illustrated in Figure 13.1. In this figure the two rows correspond to the two possible actions of player 1, the two columns correspond to the two possible actions of player 2, and the numbers in each box are the players' payoffs to the action profile to which the box corresponds, with player 1's payoff listed first.

		Suspect 2	
		Quiet	*Fink*
Suspect 1	*Quiet*	2, 2	0, 3
	Fink	3, 0	1, 1

Figure 13.1 The *Prisoner's Dilemma* (Example 12.1).

The *Prisoner's Dilemma* models a situation in which there are gains from cooperation (each player prefers that both players choose *Quiet* than they both choose *Fink*) but each player has an incentive to "free ride" (choose *Fink*) whatever the other player does. The game is important not because we are interested in understanding the incentives for prisoners to confess, but because many other situations have similar structures. Whenever each of two players has two actions, say C (corresponding to *Quiet*) and D (corresponding to *Fink*), player 1 prefers (D, C) to (C, C) to (D, D) to (C, D), and player 2 prefers (C, D) to (C, C) to (D, D) to (D, C), the *Prisoner's Dilemma* models the situation that the players face. Some examples follow.

2.2.1 Working on a joint project

You are working with a friend on a joint project. Each of you can either work hard or goof off. If your friend works hard then you prefer to goof off (the outcome of the project would be better if you worked hard too, but the increment in its value to you is not worth the extra effort). You prefer the outcome of your both working

hard to the outcome of your both goofing off (in which case nothing gets accomplished), and the worst outcome for you is that you work hard and your friend goofs off (you hate to be "exploited"). If your friend has the same preferences then the game that models the situation you face is given in Figure 14.1, which, as you can see, differs from the *Prisoner's Dilemma* only in the names of the actions.

	Work hard	*Goof off*
Work hard	$2, 2$	$0, 3$
Goof off	$3, 0$	$1, 1$

Figure 14.1 Working on a joint project.

I am *not* claiming that a situation in which two people pursue a joint project *necessarily* has the structure of the *Prisoner's Dilemma*, only that the players' preferences in such a situation *may* be the same as in the *Prisoner's Dilemma*! If, for example, each person prefers to work hard than to goof off when the other person works hard, then the *Prisoner's Dilemma* does *not* model the situation: the players' preferences are different from those given in Figure 14.1.

? EXERCISE 14.1 (Working on a joint project) Formulate a strategic game that models a situation in which two people work on a joint project in the case that their preferences are the same as those in the game in Figure 14.1 except that each person prefers to work hard than to goof off when the other person works hard. Present your game in a table like the one in Figure 14.1.

2.2.2 Duopoly

In a simple model of a duopoly, two firms produce the same good, for which each firm charges either a low price or a high price. Each firm wants to achieve the highest possible profit. If both firms choose *High* then each earns a profit of \$1000. If one firm chooses *High* and the other chooses *Low* then the firm choosing *High* obtains no customers and makes a loss of \$200, whereas the firm choosing *Low* earns a profit of \$1200 (its unit profit is low, but its volume is high). If both firms choose *Low* then each earns a profit of \$600. Each firm cares only about its profit, so we can represent its preferences by the profit it obtains, yielding the game in Figure 14.2.

	High	*Low*
High	$1000, 1000$	$-200, 1200$
Low	$1200, -200$	$600, 600$

Figure 14.2 A simple model of a price-setting duopoly.

Bearing in mind that what matters are the players' preferences, not the particular payoff functions that we use to represent them, we see that this game, like the previous one, differs from the *Prisoner's Dilemma* only in the names of the actions.

The action *High* plays the role of *Quiet*, and the action *Low* plays the role of *Fink*; firm 1 prefers $(Low, High)$ to $(High, High)$ to (Low, Low) to $(High, Low)$, and firm 2 prefers $(High, Low)$ to $(High, High)$ to (Low, Low) to $(Low, High)$.

As in the previous example, I do not claim that the incentives in a duopoly are *necessarily* those in the *Prisoner's Dilemma*; different assumptions about the relative sizes of the profits in the four cases generate a different game. Further, in this case one of the abstractions incorporated into the model—that each firm has only two prices to choose between—may not be harmless; if the firms may choose among many prices then the structure of the interaction may change. (A richer model is studied in Section 3.2.)

2.2.3 The arms race

Under some assumptions about the countries' preferences, an arms race can be modeled as the *Prisoner's Dilemma*. (Because the *Prisoner's Dilemma* was first studied in the early 1950s, when the USA and USSR were involved in a nuclear arms race, you might suspect that US nuclear strategy was influenced by game theory; the evidence suggests that it was not.) Assume that each country can build an arsenal of nuclear bombs, or can refrain from doing so. Assume also that each country's favorite outcome is that it has bombs and the other country does not; the next best outcome is that neither country has any bombs; the next best outcome is that both countries have bombs (what matters is relative strength, and bombs are costly to build); and the worst outcome is that only the other country has bombs. In this case the situation is modeled by the *Prisoner's Dilemma*, in which the action *Don't build bombs* corresponds to *Quiet* in Figure 13.1 and the action *Build bombs* corresponds to *Fink*. However, once again the assumptions about preferences necessary for the *Prisoner's Dilemma* to model the situation may not be satisfied: a country may prefer *not* to build bombs if the other country does not, for example (bomb-building may be very costly), in which case the situation is modeled by a different game.

2.2.4 Common property

Two farmers are deciding how much to allow their sheep to graze on the village common. Each farmer prefers that her sheep graze a lot than a little, regardless of the other farmer's action, but prefers that both farmers' sheep graze a little than both farmers' sheep graze a lot (in which case the common is ruined for future use). Under these assumptions the game is the *Prisoner's Dilemma*. (A richer model is studied in Section 3.1.5.)

2.2.5 Other situations modeled as the Prisoner's Dilemma

A huge number of other situations have been modeled as the *Prisoner's Dilemma*, from mating hermaphroditic fish to tariff wars between countries.

? EXERCISE 16.1 (Hermaphroditic fish) Members of some species of hermaphroditic fish choose, in each mating encounter, whether to play the role of a male or a female. Each fish has a preferred role, which uses up fewer resources and hence allows more future mating. A fish obtains a payoff of H if it mates in its preferred role and L if it mates in the other role, where $H > L$. (Payoffs are measured in terms of number of offspring, which fish are evolved to maximize.) Consider an encounter between two fish whose preferred roles are the same. Each fish has two possible actions: mate in either role, and insist on its preferred role. If both fish offer to mate in either role, the roles are assigned randomly, and each fish's payoff is $\frac{1}{2}(H + L)$ (the average of H and L). If each fish insists on its preferred role, the fish do not mate; each goes off in search of another partner, and obtains the payoff S. The higher the chance of meeting another partner, the larger is S. Formulate this situation as a strategic game and determine the range of values of S, for any given values of H and L, for which the game differs from the *Prisoner's Dilemma* only in the names of the actions.

2.3 Example: Bach or Stravinsky?

In the *Prisoner's Dilemma* the main issue is whether or not the players will cooperate (choose *Quiet*). In the following game the players agree that it is better to cooperate than not to cooperate, but disagree about the best outcome.

◆ EXAMPLE 16.2 (Bach or Stravinsky?) Two people wish to go out together. Two concerts are available: one of music by Bach, and one of music by Stravinsky. One person prefers Bach and the other prefers Stravinsky. If they go to different concerts, each of them is equally unhappy listening to the music of either composer.

We can model this situation as the two-player strategic game in Figure 16.1, in which the person who prefers Bach chooses a row and the person who prefers Stravinsky chooses a column.

	Bach	*Stravinsky*
Bach	2, 1	0, 0
Stravinsky	0, 0	1, 2

Figure 16.1 *Bach or Stravinsky?* (*BoS*) (Example 16.2).

This game is also referred to as the "Battle of the Sexes" (though the conflict it models surely occurs no more frequently between people of the opposite sex than it does between people of the same sex). I refer to the games as *BoS*, an acronym that fits both names. (I assume that each player is indifferent between listening to Bach and listening to Stravinsky when she is alone only for consistency with the standard specification of the game. As we shall see, the analysis of the game remains the same in the absence of this assumption.)

Like the *Prisoner's Dilemma*, *BoS* models a wide variety of situations. Consider, for example, two officials of a political party deciding the stand to take on an issue.

Suppose that they disagree about the best stand, but are both better off if they take the same stand than if they take different stands; both cases in which they take different stands, in which case voters do not know what to think, are equally bad. Then *BoS* captures the situation they face. Or consider two merging firms that currently use different computer technologies. As two divisions of a single firm they will both be better off if they both use the same technology; each firm prefers that the common technology be the one it used in the past. *BoS* models the choices the firms face.

2.4 Example: Matching Pennies

Aspects of both conflict and cooperation are present in both the *Prisoner's Dilemma* and *BoS*. The next game is purely conflictual.

◆ EXAMPLE 17.1 (Matching Pennies) Two people choose, simultaneously, whether to show the Head or the Tail of a coin. If they show the same side, person 2 pays person 1 a dollar; if they show different sides, person 1 pays person 2 a dollar. Each person cares only about the amount of money she receives, and (naturally!) prefers to receive more than less. A strategic game that models this situation is shown in Figure 17.1. (In this representation of the players' preferences, the payoffs are equal to the amounts of money involved. We could equally well work with another representation—for example, 2 could replace each 1, and 1 could replace each -1.)

	Head	*Tail*
Head	$1, -1$	$-1, \ \ 1$
Tail	$-1, \ \ 1$	$1, -1$

Figure 17.1 *Matching Pennies* (Example 17.1).

In this game the players' interests are diametrically opposed (such a game is called "strictly competitive"): player 1 wants to take the same action as the other player, whereas player 2 wants to take the opposite action.

This game may, for example, model the choices of appearances for new products by an established producer and a new firm in a market of fixed size. Suppose that each firm can choose one of two different appearances for the product. The established producer prefers the newcomer's product to look different from its own (so that its customers will not be tempted to buy the newcomer's product), whereas the newcomer prefers that the products look alike. Or the game could model a relationship between two people in which one person wants to be like the other, whereas the other wants to be different.

? EXERCISE 17.2 (Games without conflict) Give some examples of two-player strategic games in which each player has two actions and the players have the same pref-

erences, so that there is no conflict between their interests. (Present your games as tables like the one in Figure 17.1.)

2.5 Example: the Stag Hunt

A sentence in *Discourse on the origin and foundations of inequality among men* (1755) by the philosopher Jean-Jacques Rousseau discusses a group of hunters who wish to catch a stag. They will succeed if they all remain sufficiently attentive, but each is tempted to desert her post and catch a hare. One interpretation of the sentence is that the interaction between the hunters may be modeled as the following strategic game.

◆ EXAMPLE 18.1 (Stag Hunt) Each of a group of hunters has two options: she may remain attentive to the pursuit of a stag, or catch a hare. If all hunters pursue the stag, they catch it and share it equally; if any hunter devotes her energy to catching a hare, the stag escapes, and the hare belongs to the defecting hunter alone. Each hunter prefers a share of the stag to a hare.

The strategic game that corresponds to this specification is:

Players The hunters.

Actions Each player's set of actions is $\{Stag, Hare\}$.

Preferences For each player, the action profile in which all players choose *Stag* (resulting in her obtaining a share of the stag) is ranked highest, followed by any profile in which she chooses *Hare* (resulting in her obtaining a hare), followed by any profile in which she chooses *Stag* and one or more of the other players chooses *Hare* (resulting in her leaving empty-handed).

Like other games with many players, this game cannot easily be presented in a table like that in Figure 17.1. For the case in which there are two hunters, the game is shown in Figure 18.1.

	Stag	*Hare*
Stag	$2, 2$	$0, 1$
Hare	$1, 0$	$1, 1$

Figure 18.1 The *Stag Hunt* (Example 18.1) for the case of two hunters.

The variant of the two-player *Stag Hunt* shown in Figure 19.1 has been suggested as an alternative to the *Prisoner's Dilemma* as a model of an arms race, or, more generally, of the "security dilemma" faced by a pair of countries. The game differs from the *Prisoner's Dilemma* in that a country prefers the outcome in which both countries refrain from arming themselves to the one in which it alone arms itself: the cost of arming outweighs the benefit if the other country does not arm itself.

	Refrain	*Arm*
Refrain	$3, 3$	$0, 2$
Arm	$2, 0$	$1, 1$

Figure 19.1 A variant of the two-player *Stag Hunt* that models the "security dilemma".

2.6 Nash equilibrium

What actions will be chosen by the players in a strategic game? We wish to assume, as in the theory of a rational decision-maker (Section 1.2), that each player chooses the best available action. In a game, the best action for any given player depends, in general, on the other players' actions. So when choosing an action a player must have in mind the actions the other players will choose. That is, she must form a *belief* about the other players' actions.

On what basis can such a belief be formed? The assumption underlying the analysis in this chapter and the next two chapters is that each player's belief is derived from her past experience playing the game, and that this experience is sufficiently extensive that she *knows* how her opponents will behave. No one tells her the actions her opponents will choose, but her previous involvement in the game leads her to be sure of these actions. (The question of *how* a player's experience can lead her to the correct beliefs about the other players' actions is addressed briefly in Section 4.9.)

Although we assume that each player has experience playing the game, we assume that she views each play of the game in isolation. She does not become familiar with the behavior of specific opponents and consequently does not condition her action on the opponent she faces; nor does she expect her current action to affect the other players' future behavior.

It is helpful to think of the following idealized circumstances. For each player in the game there is a population of many decision-makers who may, on any occasion, take that player's role. In each play of the game, players are selected randomly, one from each population. Thus each player engages in the game repeatedly, against ever-varying opponents. Her experience leads her to beliefs about the actions of "typical" opponents, not any specific set of opponents.

As an example, think of the interaction between buyers and sellers. Buyers and sellers repeatedly interact, but to a first approximation many of the pairings may be modeled as random. In many cases a buyer transacts only once with any given seller, or interacts repeatedly but anonymously (when the seller is a large store, for example).

In summary, the solution theory we study has two components. First, each player chooses her action according to the model of rational choice, given her belief about the other players' actions. Second, every player's belief about the other players' actions is correct. These two components are embodied in the following definition.

JOHN F. NASH, JR.

A few of the ideas of John F. Nash Jr., developed while he was a graduate student at Princeton from 1948 to 1950, transformed game theory. Nash was born in 1928 in Bluefield, West Virginia, USA, where he grew up. He was an undergraduate mathematics major at Carnegie Institute of Technology from 1945 to 1948. In 1948 he obtained both a B.S. and an M.S., and began graduate work in the Department of Mathematics at Princeton University. (One of his letters of recommendation, from a professor at Carnegie Institute of Technology, was a single sentence: "This man is a genius" (Kuhn et al. 1995, 282).) A paper containing the main result of his thesis was submitted to the *Proceedings of the National Academy of Sciences* in November 1949, fourteen months after he started his graduate work. ("A fine goal to set ... graduate students", to quote Kuhn! (See Kuhn et al. 1995, 282.)) He completed his PhD the following year, graduating on his 22nd birthday. His thesis, 28 pages in length, introduces the equilibrium notion now known as "Nash equilibrium" and delineates a class of strategic games that have Nash equilibria (Proposition 116.1 in this book). The notion of Nash equilibrium vastly expanded the scope of game theory, which had previously focussed on two-player "strictly competitive" games (in which the players' interests are directly opposed). While a graduate student at Princeton, Nash also wrote the seminal paper in bargaining theory, Nash (1950b) (the ideas of which originated in an elective class in international economics he took as an undergraduate). He went on to take an academic position in the Department of Mathematics at MIT, where he produced "a remarkable series of papers" (Milnor 1995, 15); he has been described as "one of the most original mathematical minds of [the twentieth] century" (Kuhn 1996). He shared the 1994 Nobel prize in economics with the game theorists John C. Harsanyi and Reinhard Selten.

> A *Nash equilibrium* is an action profile a^* with the property that no player i can do better by choosing an action different from a_i^*, given that every other player j adheres to a_j^*.

In the idealized setting in which the players in any given play of the game are drawn randomly from a collection of populations, a Nash equilibrium corresponds to a *steady state*. If, whenever the game is played, the action profile is the same Nash equilibrium a^*, then no player has a reason to choose any action different from her component of a^*; there is no pressure on the action profile to change. Expressed differently, a Nash equilibrium embodies a stable "social norm": if everyone else adheres to it, no individual wishes to deviate from it.

The second component of the theory of Nash equilibrium—that the players' beliefs about each other's actions are correct—implies, in particular, that two players' beliefs about a third player's action are the same. For this reason, the condition is sometimes said to be that the players' "expectations are coordinated".

The situations to which we wish to apply the theory of Nash equilibrium do

not in general correspond exactly to the idealized setting described above. For example, in some cases the players do not have much experience with the game; in others they do not view each play of the game in isolation. Whether or not the notion of Nash equilibrium is appropriate in any given situation is a matter of judgment. In some cases, a poor fit with the idealized setting may be mitigated by other considerations. For example, inexperienced players may be able to draw conclusions about their opponents' likely actions from their experience in other situations, or from other sources. (One aspect of such reasoning is discussed in the box on page 30). Ultimately, the test of the appropriateness of the notion of Nash equilibrium is whether it gives us insights into the problem at hand.

With the aid of an additional piece of notation, we can state the definition of a Nash equilibrium precisely. Let a be an action profile, in which the action of each player i is a_i. Let a_i' be any action of player i (either equal to a_i, or different from it). Then (a_i', a_{-i}) denotes the action profile in which every player j *except* i chooses her action a_j as specified by a, whereas player i chooses a_i'. (The $-i$ subscript on a stands for "except i".) That is, (a_i', a_{-i}) is the action profile in which all the players other than i adhere to a while i "deviates" to a_i'. (If $a_i' = a_i$ then of course $(a_i', a_{-i}) = (a_i, a_{-i}) = a$.) If there are three players, for example, then (a_2', a_{-2}) is the action profile in which players 1 and 3 adhere to a (player 1 chooses a_1, player 3 chooses a_3) and player 2 deviates to a_2'.

Using this notation, we can restate the condition for an action profile a^* to be a Nash equilibrium: no player i has any action a_i for which she prefers (a_i, a^*_{-i}) to a^*. Equivalently, for every player i and every action a_i of player i, the action profile a^* is at least as good for player i as the action profile (a_i, a^*_{-i}).

▶ DEFINITION 21.1 (*Nash equilibrium of strategic game with ordinal preferences*) The action profile a^* in a strategic game with ordinal preferences is a **Nash equilibrium** if, for every player i and every action a_i of player i, a^* is at least as good according to player i's preferences as the action profile (a_i, a^*_{-i}) in which player i chooses a_i while every other player j chooses a^*_j. Equivalently, for every player i,

$$u_i(a^*) \geq u_i(a_i, a^*_{-i}) \text{ for every action } a_i \text{ of player } i, \tag{21.2}$$

where u_i is a payoff function that represents player i's preferences.

This definition implies neither that a strategic game necessarily has a Nash equilibrium, nor that it has at most one. Examples in the next section show that some games have a single Nash equilibrium, some possess no Nash equilibrium, and others have many Nash equilibria.

The definition of a Nash equilibrium is designed to model a steady state among experienced players. An alternative approach to understanding players' actions in strategic games assumes that the players know each others' preferences, and considers what each player can deduce about the other players' actions from their rationality and their knowledge of each other's rationality. This approach is studied in Chapter 12. For many games, it leads to a conclusion different from that of

Nash equilibrium. For games in which the conclusion is the same the approach offers us an alternative interpretation of a Nash equilibrium, as the outcome of rational calculations by players who do not necessarily have any experience playing the game.

STUDYING NASH EQUILIBRIUM EXPERIMENTALLY

The theory of strategic games lends itself to experimental study: arranging for subjects to play games and observing their choices is relatively straightforward. A few years after game theory was launched by von Neumann and Morgenstern's (1944) book, reports of laboratory experiments began to appear. Subsequently a huge number of experiments have been conducted, illuminating many issues relevant to the theory. I discuss selected experimental evidence throughout the book.

The theory of Nash equilibrium, as we have seen, has two components: the players act in accordance with the theory of rational choice, given their beliefs about the other players' actions, and these beliefs are correct. If every subject understands the game she is playing and faces incentives that correspond to the preferences of the player whose role she is taking, then a divergence between the observed outcome and a Nash equilibrium can be blamed on a failure of one or both of these two components. Experimental evidence has the potential of indicating the types of games for which the theory works well and, for those in which the theory does not work well, of pointing to the faulty component and giving us hints about the characteristics of a better theory. In designing an experiment that cleanly tests the theory, however, we need to confront several issues.

The model of rational choice takes preferences as given. Thus to test the theory of Nash equilibrium experimentally, we need to ensure that each subject's preferences are those of the player whose role she is taking in the game we are examining. The standard way of inducing the appropriate preferences is to pay each subject an amount of money directly related to the payoff given by a payoff function that represents the preferences of the player whose role the subject is taking. Such remuneration works if each subject likes money and cares only about the amount of money she receives, ignoring the amounts received by her opponents. The assumption that people like receiving money is reasonable in many cultures, but the assumption that people care only about their own monetary rewards—are "selfish"—may, in some contexts at least, not be reasonable. Unless we check whether our subjects are selfish in the context of our experiment, we will jointly test two hypotheses: that humans are selfish—a hypothesis not part of game theory—and that the notion of Nash equilibrium models their behavior. In some cases we may indeed wish to test these hypotheses jointly. But in order to test the theory of Nash equilibrium alone we need to ensure that we induce the preferences we wish to study.

Assuming that better decisions require more effort, we need also to ensure that

each subject finds it worthwhile to put in the extra effort required to obtain a higher payoff. If we rely on monetary payments to provide incentives, the amount of money a subject can obtain must be sufficiently sensitive to the quality of her decisions to compensate her for the effort she expends (paying a flat fee, for example, is inappropriate). In some cases, monetary payments may not be necessary: under some circumstances, subjects drawn from a highly competitive culture like that of the USA may be sufficiently motivated by the possibility of obtaining a high score, even if that score does not translate into a monetary payoff.

The notion of Nash equilibrium models action profiles compatible with steady states. Thus to study the theory experimentally we need to collect observations of subjects' behavior when they have experience playing the game. But they should not have obtained that experience while knowingly facing the same opponents repeatedly, for the theory assumes that the players consider each play of the game in isolation, not as part of an ongoing relationship. One option is to have each subject play the game against many different opponents, gaining experience about how the other subjects on average play the game, but not about the choices of any other given player. Another option is to describe the game in terms that relate to a situation in which the subjects already have experience. A difficulty with this second approach is that the description we give may connote more than simply the payoff numbers of our game. If we describe the *Prisoner's Dilemma* in terms of cooperation on a joint project, for example, a subject may be biased toward choosing the action she has found appropriate when involved in joint projects, even if the structures of those interactions were significantly different from that of the *Prisoner's Dilemma*. As she plays the experimental game repeatedly she may come to appreciate how it differs from the games in which she has been involved previously, but her biases may disappear only slowly.

Whatever route we take to collect data on the choices of subjects experienced in playing the game, we confront a difficult issue: how do we know when the outcome has converged? Nash's theory concerns only equilibria; it has nothing to say about the path players' choices will take on the way to an equilibrium, and so gives us no guide as to whether 10, 100, or 1,000 plays of the game are enough to give a chance for the subjects' expectations to become coordinated.

Finally, we can expect the theory of Nash equilibrium to correspond to reality only approximately: like all useful theories, it definitely is not *exactly* correct. How do we tell whether the data are close enough to the theory to support it? One possibility is to compare the theory of Nash equilibrium with some other theory. But for many games there is no obvious alternative theory—and certainly not one with the generality of Nash equilibrium. Statistical tests can sometimes aid in deciding whether the data is consistent with the theory, though ultimately we remain the judge of whether or not our observations persuade us that the theory enhances our understanding of human behavior in the game.

2.7 Examples of Nash equilibrium

2.7.1 Prisoner's Dilemma

By examining the four possible pairs of actions in the *Prisoner's Dilemma* (reproduced in Figure 24.1), we see that $(Fink, Fink)$ is the unique Nash equilibrium.

	Quiet	*Fink*
Quiet	$2, 2$	$0, 3$
Fink	$3, 0$	$1, 1$

Figure 24.1 The *Prisoner's Dilemma*.

The action pair $(Fink, Fink)$ is a Nash equilibrium because (*i*) given that player 2 chooses *Fink*, player 1 is better off choosing *Fink* than *Quiet* (looking at the right column of the table we see that *Fink* yields player 1 a payoff of 1 whereas *Quiet* yields her a payoff of 0), and (*ii*) given that player 1 chooses *Fink*, player 2 is better off choosing *Fink* than *Quiet* (looking at the bottom row of the table we see that *Fink* yields player 2 a payoff of 1 whereas *Quiet* yields her a payoff of 0).

No other action profile is a Nash equilibrium:

- $(Quiet, Quiet)$ does not satisfy (21.2) because when player 2 chooses *Quiet*, player 1's payoff to *Fink* exceeds her payoff to *Quiet* (look at the first components of the entries in the left column of the table). (Further, when player 1 chooses *Quiet*, player 2's payoff to *Fink* exceeds her payoff to *Quiet*: player 2, as well as player 1, wants to deviate. To show that a pair of actions is not a Nash equilibrium, however, it is not necessary to study player 2's decision once we have established that player 1 wants to deviate: it is enough to show that *one* player wishes to deviate to show that a pair of actions is not a Nash equilibrium.)
- $(Fink, Quiet)$ does not satisfy (21.2) because when player 1 chooses *Fink*, player 2's payoff to *Fink* exceeds her payoff to *Quiet* (look at the second components of the entries in the bottom row of the table).
- $(Quiet, Fink)$ does not satisfy (21.2) because when player 2 chooses *Fink*, player 1's payoff to *Fink* exceeds her payoff to *Quiet* (look at the first components of the entries in the right column of the table).

In summary, in the only Nash equilibrium of the *Prisoner's Dilemma* both players choose *Fink*. In particular, the incentive to free ride eliminates the possibility that the mutually desirable outcome $(Quiet, Quiet)$ occurs. In the other situations discussed in Section 2.2 that may be modeled as the *Prisoner's Dilemma*, the outcomes predicted by the notion of Nash equilibrium are thus as follows: both people goof off when working on a joint project; both duopolists charge a low price; both countries build bombs; both farmers graze their sheep a lot. (The overgrazing

of a common thus predicted is sometimes called the "tragedy of the commons". The intuition that some of these dismal outcomes may be avoided if the same pair of people play the game repeatedly is explored in Chapter 14.)

In the *Prisoner's Dilemma*, the Nash equilibrium action of each player (*Fink*) is the best action for each player not only if the other player chooses her equilibrium action (*Fink*), but also if she chooses her other action (*Quiet*). The action pair $(Fink, Fink)$ is a Nash equilibrium because if a player believes that her opponent will choose *Fink* then it is optimal for her to choose *Fink*. But in fact it is optimal for a player to choose *Fink regardless* of the action she expects her opponent to choose. In most of the games we study, a player's Nash equilibrium action does not satisfy this condition: the action is optimal if the other players choose their Nash equilibrium actions, but some other action is optimal if the other players choose non-equilibrium actions.

? EXERCISE 25.1 (Altruistic players in the *Prisoner's Dilemma*) Each of two players has two possible actions, *Quiet* and *Fink*; each action pair results in the players' receiving amounts of *money* equal to the numbers corresponding to that action pair in Figure 24.1. (For example, if player 1 chooses *Quiet* and player 2 chooses *Fink*, then player 1 receives nothing, whereas player 2 receives \$3.) The players are not "selfish"; rather, the preferences of each player i are represented by the payoff function $m_i(a) + \alpha m_j(a)$, where $m_i(a)$ is the amount of money received by player i when the action profile is a, j is the other player, and α is a given nonnegative number. Player 1's payoff to the action pair $(Quiet, Quiet)$, for example, is $2 + 2\alpha$.

a. Formulate a strategic game that models this situation in the case $\alpha = 1$. Is this game the *Prisoner's Dilemma*?

b. Find the range of values of α for which the resulting game is the *Prisoner's Dilemma*. For values of α for which the game is not the *Prisoner's Dilemma*, find its Nash equilibria.

? EXERCISE 25.2 (Selfish and altruistic social behavior) Two people enter a bus. Two adjacent cramped seats are free. Each person must decide whether to sit or stand. Sitting alone is more comfortable than sitting next to the other person, which is more comfortable than standing.

a. Suppose that each person cares only about her own comfort. Model the situation as a strategic game. Is this game the *Prisoner's Dilemma*? Find its Nash equilibrium (equilibria?).

b. Suppose that each person is altruistic, ranking the outcomes according to the *other* person's comfort, and, out of politeness, prefers to stand than to sit if the other person stands. Model the situation as a strategic game. Is this game the *Prisoner's Dilemma*? Find its Nash equilibrium (equilibria?).

c. Compare the people's comfort in the equilibria of the two games.

EXPERIMENTAL EVIDENCE ON THE *Prisoner's Dilemma*

The *Prisoner's Dilemma* has attracted a great deal of attention by economists, psychologists, sociologists, and biologists. A huge number of experiments have been conducted with the aim of discovering how people behave when playing the game. Almost all these experiments involve each subject's playing the game repeatedly against an unchanging opponent, a situation that calls for an analysis significantly different from the one in this chapter (see Chapter 14).

The evidence on the outcome of isolated plays of the game is inconclusive. No experiment of which I am aware carefully induces the appropriate preferences and is specifically designed to elicit a steady state action profile (see the box on page 22). Thus in each case the choice of *Quiet* by a player could indicate that she is not "selfish" or that she is not experienced in playing the game, rather than providing evidence against the notion of Nash equilibrium.

In two experiments with very low payoffs, each subject played the game a small number of times against different opponents; between 50% and 94% of subjects chose *Fink*, depending on the relative sizes of the payoffs and some details of the design (Rapoport, Guyer, and Gordon 1976, 135–137, 211–213, and 223-226). A more recent experiment finds that in the last 10 of 20 rounds of play against different opponents, 78% of subjects choose *Fink* (Cooper, DeJong, Forsythe, and Ross 1996). In face-to-face games in which communication is allowed, the incidence of the choice of *Fink* tends to be lower: from 29% to 70% depending on the nature of the communication allowed (Deutsch 1958, and Frank, Gilovich, and Regan 1993, 163–167). (In all these experiments, the subjects were college students in the USA or Canada.)

One source of the variation in the results seems to be that some designs induce preferences that differ from those of the *Prisoner's Dilemma*; no clear answer emerges to the question of whether the notion of Nash equilibrium is relevant to the *Prisoner's Dilemma*. If, nevertheless, one interprets the evidence as showing that some subjects in the *Prisoner's Dilemma* systematically choose *Quiet* rather than *Fink*, one must fault the rational choice component of Nash equilibrium, not the coordinated expectations component. Why? Because, as noted in the text, *Fink* is optimal *no matter* what a player thinks her opponent will choose, so that any model in which the players act according to the model of rational choice, whether or not their expectations are coordinated, predicts that each player chooses *Fink*.

2.7.2 BoS

To find the Nash equilibria of *BoS* (Figure 16.1), we can examine each pair of actions in turn:

- (*Bach*, *Bach*): If player 1 switches to *Stravinsky* then her payoff decreases from 2 to 0; if player 2 switches to *Stravinsky* then her payoff decreases from 1 to 0.

Thus a deviation by either player decreases her payoff. Thus (*Bach*, *Bach*) is a Nash equilibrium.

- (*Bach*, *Stravinsky*): If player 1 switches to *Stravinsky* then her payoff increases from 0 to 1. Thus (*Bach*, *Stravinsky*) is not a Nash equilibrium. (Player 2 can increase her payoff by deviating, too, but to show the pair is not a Nash equilibrium it suffices to show that one player can increase her payoff by deviating.)

- (*Stravinsky*, *Bach*): If player 1 switches to *Bach* then her payoff increases from 0 to 2. Thus (*Stravinsky*, *Bach*) is not a Nash equilibrium.

- (*Stravinsky*, *Stravinsky*): If player 1 switches to *Bach* then her payoff decreases from 1 to 0; if player 2 switches to *Bach* then her payoff decreases from 2 to 0. Thus a deviation by either player decreases her payoff. Thus (*Stravinsky*, *Stravinsky*) is a Nash equilibrium.

We conclude that the game has two Nash equilibria: (*Bach*, *Bach*) and (*Stravinsky*, *Stravinsky*). That is, both of these outcomes are compatible with a steady state; both outcomes are stable social norms. If, in every encounter, both players choose *Bach*, then no player has an incentive to deviate; if, in every encounter, both players choose *Stravinsky*, then no player has an incentive to deviate. If we use the game to model the choices of men when matched with women, for example, then the notion of Nash equilibrium shows that two social norms are stable: both players choose the action associated with the outcome preferred by women, and both players choose the action associated with the outcome preferred by men.

2.7.3 Matching Pennies

By checking each of the four pairs of actions in *Matching Pennies* (Figure 17.1) we see that the game has no Nash equilibrium. For the pairs of actions (*Head*, *Head*) and (*Tail*, *Tail*), player 2 is better off deviating; for the pairs of actions (*Head*, *Tail*) and (*Tail*, *Head*), player 1 is better off deviating. Thus for this game the notion of Nash equilibrium isolates no steady state. In Chapter 4 we return to this game; an extension of the notion of a Nash equilibrium gives us an understanding of the likely outcome.

2.7.4 The Stag Hunt

Inspection of Figure 18.1 shows that the two-player *Stag Hunt* has two Nash equilibria: (*Stag*, *Stag*) and (*Hare*, *Hare*). If one player remains attentive to the pursuit of the stag, then the other player prefers to remain attentive; if one player chases a hare, the other one prefers to chase a hare (she cannot catch a stag alone). (The equilibria of the variant of the game in Figure 19.1 are analogous: (*Refrain*, *Refrain*) and (*Arm*, *Arm*).)

Unlike the Nash equilibria of *BoS*, one of these equilibria is better for both players than the other: each player prefers $(Stag, Stag)$ to $(Hare, Hare)$. This fact has no bearing on the equilibrium status of $(Hare, Hare)$, since the condition for an equilibrium is that a *single* player cannot gain by deviating, *given* the other player's behavior. Put differently, an equilibrium is immune to any *unilateral* deviation; coordinated deviations by groups of players are not contemplated. However, the existence of two equilibria raises the possibility that one equilibrium might more likely be the outcome of the game than the other. I return to this issue in Section 2.7.6.

I argue that the many-player *Stag Hunt* (Example 18.1) also has two Nash equilibria: the action profile $(Stag, \ldots, Stag)$ in which every players joins in the pursuit of the stag, and the profile $(Hare, \ldots, Hare)$ in which every player catches a hare.

- $(Stag, \ldots, Stag)$ is a Nash equilibrium because each player prefers this profile to that in which she alone chooses *Hare*. (A player is better off remaining attentive to the pursuit of the stag than running after a hare if all the other players remain attentive.)
- $(Hare, \ldots, Hare)$ is a Nash equilibrium because each player prefers this profile to that in which she alone pursues the stag. (A player is better off catching a hare than pursuing the stag if no one else pursues the stag.)
- No other profile is a Nash equilibrium, because in any other profile at least one player chooses *Stag* and at least one player chooses *Hare*, so that any player choosing *Stag* is better off switching to *Hare*. (A player is better off catching a hare than pursing the stag if at least one other person chases a hare, since the stag can be caught only if everyone pursues it.)

? EXERCISE 28.1 (Variants of the *Stag Hunt*) Consider two variants of the n-hunter *Stag Hunt* in which only m hunters, with $2 \leq m < n$, need to pursue the stag in order to catch it. (Continue to assume that there is a single stag.) Assume that a captured stag is shared only by the hunters that catch it.

a. Assume, as before, that each hunter prefers the fraction $1/n$ of the stag to a hare. Find the Nash equilibria of the strategic game that models this situation.

b. Assume that each hunter prefers the fraction $1/k$ of the stag to a hare, but prefers the hare to any smaller fraction of the stag, where k is an integer with $m \leq k \leq n$. Find the Nash equilibria of the strategic game that models this situation.

The following more difficult exercise enriches the hunters' choices in the *Stag Hunt*. This extended game has been proposed as a model that captures Keynes' basic insight about the possibility of multiple economic equilibria, some undesirable (Bryant 1983, 1994).

?? EXERCISE 28.2 (Extension of the *Stag hunt*) Extend the n-hunter *Stag Hunt* by giving each hunter K (a positive integer) units of effort, which she can allocate between pursuing the stag and catching hares. Denote the effort hunter i devotes

to pursuing the stag by e_i, a nonnegative integer equal to at most K. The chance that the stag is caught depends on the smallest of all the hunters' efforts, denoted $\min_j e_j$. ("A chain is as strong as its weakest link.") Hunter i's payoff to the action profile $(e_1, \ldots, e_n)$ is $2 \min_j e_j - e_i$. (She is better off the more likely the stag is caught, and worse off the more effort she devotes to pursuing the stag, which means she catches fewer hares.) Is the action profile $(e, \ldots, e)$, in which every hunter devotes the same effort to pursuing the stag, a Nash equilibrium for any value of e? (What is a player's payoff to this profile? What is her payoff if she deviates to a lower or higher effort level?) Is any action profile in which not all the players' effort levels are the same a Nash equilibrium? (Consider a player whose effort exceeds the minimum effort level of all players. What happens to her payoff if she reduces her effort level to the minimum?)

2.7.5 Hawk–Dove

The game in the next exercise captures a basic feature of animal conflict.

? EXERCISE 29.1 (Hawk–Dove) Two animals are fighting over some prey. Each can be passive or aggressive. Each prefers to be aggressive if its opponent is passive, and passive if its opponent is aggressive; given its own stance, it prefers the outcome when its opponent is passive to that in which its opponent is aggressive. Formulate this situation as a strategic game and find its Nash equilibria.

2.7.6 A coordination game

Consider two people who wish to go out together, but who, unlike the dissidents in *BoS*, agree on the more desirable concert—say they both prefer *Bach*. A strategic game that models this situation is shown in Figure 29.1; it is an example of a *coordination game*. By examining the four action pairs, we see that the game has two Nash equilibria: (*Bach*, *Bach*) and (*Stravinsky*, *Stravinsky*). In particular, the action pair (*Stravinsky*, *Stravinsky*) in which both people choose their less-preferred concert is a Nash equilibrium.

	Bach	*Stravinsky*
Bach	2, 2	0, 0
Stravinsky	0, 0	1, 1

Figure 29.1 *A coordination game.*

Is the equilibrium in which both people choose *Stravinsky* plausible? People who argue that the technology of Apple computers originally dominated that of IBM computers, and that the Beta format for video recording is better than VHS, would say "yes". In both cases users had a strong interest in adopting the same standard, and one standard was better than the other; in the steady state that emerged in each case, the inferior technology was adopted by a large majority of users.

FOCAL POINTS

In games with many Nash equilibria, the theory isolates more than one pattern of behavior compatible with a steady state. In some games, some of these equilibria seem more likely to attract the players' attentions than others. To use the terminology of Schelling (1960), some equilibria are *focal*. In the coordination game in Figure 29.1, where the players agree on the more desirable Nash equilibrium and obtain the same payoff to every nonequilibrium action pair, the preferable equilibrium seems more likely to be focal (though two examples are given in the text of steady states involving the inferior equilibrium). In the variant of this game in which the two equilibria are equally good (i.e. $(2, 2)$ is replaced by $(1, 1)$), nothing in the structure of the game gives any clue as to which steady state might occur. In such a game, the names or nature of the actions, or other information, may predispose the players to one equilibrium rather than the other.

Consider, for example, voters in an election. Pre-election polls may give them information about each other's intended actions, pointing them to one of many Nash equilibria. Or consider a situation in which two players independently divide \$100 into two piles, each receiving \$10 if they choose the same divisions and nothing otherwise. The strategic game that models this situation has many Nash equilibria, in each of which both players choose the same division. But the equilibrium in which both players choose the $(\$50, \$50)$ division seems likely to command the players' attentions, possibly for esthetic reasons (it is an appealing division), and possibly because it is a steady state in an unrelated game in which the chosen division determines the players' payoffs.

The theory of Nash equilibrium is neutral about the equilibrium that will occur in a game with many equilibria. If features of the situation not modeled by the notion of a strategic game make some equilibria focal then those equilibria may be more likely to emerge as steady states, and the rate at which a steady state is reached may be higher than it otherwise would have been.

If two people played this game in a laboratory it seems likely that the outcome would be $(Bach, Bach)$. Nevertheless, $(Stravinsky, Stravinsky)$ also corresponds to a steady state: if either action pair is reached, there is no reason for either player to deviate from it.

2.7.7 Provision of a public good

The model in the next exercise captures an aspect of the provision of a "public good", like a park or a swimming pool, whose use by one person does not diminish its value to another person (at least, not until it is overcrowded). (Other aspects of public good provision are studied in Section 2.8.4.)

? EXERCISE 31.1 (Contributing to a public good) Each of n people chooses whether or not to contribute a fixed amount toward the provision of a public good. The good is provided if and only if at least k people contribute, where $2 \leq k \leq n$; if it is not provided, contributions are not refunded. Each person ranks outcomes from best to worst as follows: (*i*) any outcome in which the good is provided and she does not contribute, (*ii*) any outcome in which the good is provided and she contributes, (*iii*) any outcome in which the good is not provided and she does not contribute, (*iv*) any outcome in which the good is not provided and she contributes. Formulate this situation as a strategic game and find its Nash equilibria. (Is there a Nash equilibrium in which more than k people contribute? One in which k people contribute? One in which fewer than k people contribute? (Be careful!))

2.7.8 Strict and nonstrict equilibria

In all the Nash equilibria of the games we have studied so far a deviation by a player leads to an outcome *worse* for that player than the equilibrium outcome. The definition of Nash equilibrium (21.1), however, requires only that the outcome of a deviation be *no better* for the deviant than the equilibrium outcome. And, indeed, some games have equilibria in which a player is indifferent between her equilibrium action and some other action, given the other players' actions.

Consider the game in Figure 31.1. This game has a unique Nash equilibrium, namely (T, L). (For every other pair of actions, one of the players is better off changing her action.) When player 2 chooses L, as she does in this equilibrium, player 1 is equally happy choosing T or B; if she deviates to B then she is no worse off than she is in the equilibrium. We say that the Nash equilibrium (T, L) is not a *strict equilibrium*.

	L	M	R
T	$1, 1$	$1, 0$	$0, 1$
B	$1, 0$	$0, 1$	$1, 0$

Figure 31.1 A game with a unique Nash equilibrium, which is not a strict equilibrium.

For a general game, an equilibrium is strict if each player's equilibrium action is *better* than all her other actions, given the other players' actions. Precisely, an action profile a^* is a **strict Nash equilibrium** if for every player i we have $u_i(a^*) > u_i(a_i, a^*_{-i})$ for every action $a_i \neq a^*_i$ of player i. (Contrast the strict inequality in this definition with the weak inequality in (21.2).)

2.7.9 Additional examples

The following exercises are more difficult than most of the previous ones. In the first two, the number of actions of each player is arbitrary, so you cannot mechanically examine each action profile individually, as we did for games in which each player has two actions. Instead, you can consider groups of action profiles that

have features in common, and show that all action profiles in any given group are or are not equilibria. Deciding how best to group the profiles into types calls for some intuition about the character of a likely equilibrium; the exercises contain suggestions on how to proceed.

⁇ EXERCISE 32.1 (Guessing two-thirds of the average) Each of three people announces an integer from 1 to K. If the three integers are different, the person whose integer is closest to $\frac{2}{3}$ of the average of the three integers wins \$1. If two or more integers are the same, \$1 is split equally between the people whose integer is closest to $\frac{2}{3}$ of the average integer. Is there any integer k such that the action profile (k, k, k), in which every person announces the same integer k, is a Nash equilibrium? (If $k \geq 2$, what happens if a person announces a smaller number?) Is any other action profile a Nash equilibrium? (What is the payoff of a person whose number is the highest of the three? Can she increase this payoff by announcing a different number?)

Game theory is used widely in political science, especially in the study of elections. The game in the following exercise explores citizens' costly decisions to vote.

⁇ EXERCISE 32.2 (Voter participation) Two candidates, A and B, compete in an election. Of the n citizens, k support candidate A and m ($= n - k$) support candidate B. Each citizen decides whether to vote, at a cost, for the candidate she supports, or to abstain. A citizen who abstains receives the payoff of 2 if the candidate she supports wins, 1 if this candidate ties for first place, and 0 if this candidate loses. A citizen who votes receives the payoffs $2 - c$, $1 - c$, and $-c$ in these three cases, where $0 < c < 1$.

a. For $k = m = 1$, is the game the same (except for the names of the actions) as any considered so far in this chapter?

b. For $k = m$, find the set of Nash equilibria. (Is the action profile in which everyone votes a Nash equilibrium? Is there any Nash equilibrium in which the candidates tie and not everyone votes? Is there any Nash equilibrium in which one of the candidates wins by one vote? Is there any Nash equilibrium in which one of the candidates wins by two or more votes?)

c. What is the set of Nash equilibria for $k < m$?

If, when sitting in a traffic jam, you have ever thought about the time you might save if another road were built, the next exercise may lead you to think again.

⁇ EXERCISE 32.3 (Choosing a route) Four people must drive from A to B at the same time. Two routes are available, one via X and one via Y. (Refer to the left panel of Figure 33.1.) The roads from A to X, and from Y to B are both short and narrow; in each case, one car takes 6 minutes, and each additional car increases the travel time *per car* by 3 minutes. (If two cars drive from A to X, for example, *each car* takes 9 minutes.) The roads from A to Y, and from X to B are long and wide; on A to Y one car takes 20 minutes, and each additional car increases the travel time *per car*

by 1 minute; on X to B one car takes 20 minutes, and each additional car increases the travel time *per car* by 0.9 minutes. Formulate this situation as a strategic game and find the Nash equilibria. (If all four people take one of the routes, can any of them do better by taking the other route? What if three take one route and one takes the other route, or if two take each route?)

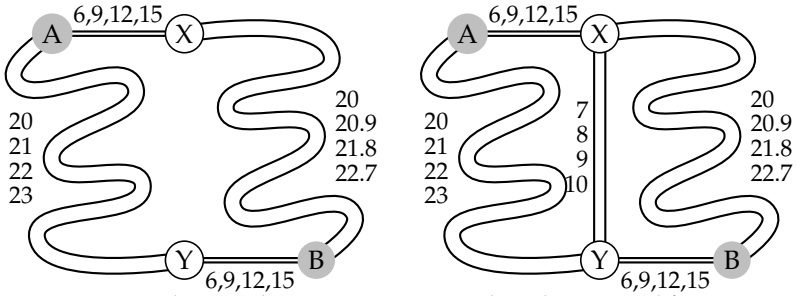


Figure 33.1 Getting from A to B: the road networks in Exercise 32.3. The numbers beside each road are the travel times *per car* when 1, 2, 3, or 4 cars take that road.

Now suppose that a relatively short, wide road is built from X to Y, giving each person four options for travel from A to B: A–X–B, A–Y–B, A–X–Y–B, and A–Y–X–B. Assume that a person who takes A–X–Y–B travels the A–X portion at the same time as someone who takes A–X–B, and the Y–B portion at the same time as someone who takes A–Y–B. (Think of there being constant flows of traffic.) On the road between X and Y, one car takes 7 minutes and each additional car increases the travel time *per car* by 1 minute. Find the Nash equilibria in this new situation. Compare each person's travel time with her travel time in the equilibrium before the road from X to Y was built.

2.8 Best response functions

2.8.1 Definition

We can find the Nash equilibria of a game in which each player has only a few actions by examining each action profile in turn to see if it satisfies the conditions for equilibrium. In more complicated games, it is often better to work with the players' "best response functions".

Consider a player, say player i. For any given actions of the players other than i, player i's actions yield her various payoffs. We are interested in the best actions—those that yield her the highest payoff. In *BoS*, for example, *Bach* is the best action for player 1 if player 2 chooses *Bach*; *Stravinsky* is the best action for player 1 if player 2 chooses *Stravinsky*. In particular, in *BoS*, player 1 has a single best action for each action of player 2. By contrast, in the game in Figure 31.1, both T and B are best actions for player 1 if player 2 chooses L: they both yield the payoff of 1, and player 1 has no action that yields a higher payoff (in fact, she has no other action).