CHAPTER 3

Introduction to Linear Programming

The development of linear programming has been ranked among the most important scientific advances of the mid-20th century, and we must agree with this assessment. Its impact since just 1950 has been extraordinary. Today it is a standard tool that has saved many thousands or millions of dollars for many companies or businesses of even moderate size in the various industrialized countries of the world, and its use in other sectors of society has been spreading rapidly. A major proportion of all scientific computation on computers is devoted to the use of linear programming. Dozens of textbooks have been written about linear programming, and *published* articles describing important applications now number in the hundreds.

What is the nature of this remarkable tool, and what kinds of problems does it address? You will gain insight into this topic as you work through subsequent examples. However, a verbal summary may help provide perspective. Briefly, the most common type of application involves the general problem of allocating *limited resources* among *competing activities* in a best possible (i.e., *optimal*) way. More precisely, this problem involves selecting the level of certain activities that compete for scarce resources that are necessary to perform those activities. The choice of activity levels then dictates how much of each resource will be consumed by each activity. The variety of situations to which this description applies is diverse, indeed, ranging from the allocation of production facilities to products to the allocation of national resources to domestic needs, from portfolio selection to the selection of shipping patterns, from agricultural planning to the design of radiation therapy, and so on. However, the one common ingredient in each of these situations is the necessity for allocating resources to activities by choosing the levels of those activities.

Linear programming uses a mathematical model to describe the problem of concern. The adjective *linear* means that all the mathematical functions in this model are required to be *linear functions.* The word *programming* does not refer here to computer programming; rather, it is essentially a synonym for *planning*. Thus, linear programming involves the *planning of activities* to obtain an optimal result, i.e., a result that reaches the specified goal best (according to the mathematical model) among all feasible alternatives.

Although allocating resources to activities is the most common type of application, linear programming has numerous other important applications as well. In fact, *any* problem whose mathematical model fits the very general format for the linear programming model is a linear programming problem. (For this reason, a linear programming problem and its model often are referred to interchangeably as simply a *linear program,* or even as just an *LP.*) Furthermore, a remarkably efficient solution procedure, called the **simplex method,** is available for solving linear programming problems of even enormous size. These are some of the reasons for the tremendous impact of linear programming in recent decades.

Because of its great importance, we devote this and the next seven chapters specifically to linear programming. After this chapter introduces the general features of linear programming, Chaps. 4 and 5 focus on the simplex method. Chapters 6 and 7 discuss the further analysis of linear programming problems *after* the simplex method has been initially applied. Chapter 8 presents several widely used extensions of the simplex method and introduces an *interior-point algorithm* that sometimes can be used to solve even larger linear programming problems than the simplex method can handle. Chapters 9 and 10 consider some special types of linear programming problems whose importance warrants individual study.

You also can look forward to seeing applications of linear programming to other areas of operations research (OR) in several later chapters.

We begin this chapter by developing a miniature prototype example of a linear programming problem. This example is small enough to be solved graphically in a straightforward way. Sections 3.2 and 3.3 present the general *linear programming model* and its basic assumptions. Section 3.4 gives some additional examples of linear programming applications. Section 3.5 describes how linear programming models of modest size can be conveniently displayed and solved on a spreadsheet. However, some linear programming problems encountered in practice require truly *massive* models. Section 3.6 illustrates how a massive model can arise and how it can still be formulated successfully with the help of a special modeling language such as MPL (its formulation is described in this section) or LINGO (its formulation of this model is presented in Supplement 2 to this chapter on the book's website).

3.1 PROTOTYPE EXAMPLE

company

window + doors

3 plants

The WYNDOR GLASS CO. produces high-quality glass products, including windows and glass doors. It has three plants. Aluminum frames and hardware are made in Plant 1, wood frames are made in Plant 2, and Plant 3 produces the glass and assembles the products.

Because of declining earnings, top management has decided to revamp the company's product line. Unprofitable products are being discontinued, releasing production capacity to launch two new products having large sales potential:

Product 1: An 8-foot glass door with aluminum framing
Product 2: A 4 × 6 foot double-hung wood-framed window

Product 1 requires some of the production capacity in Plants 1 and 3, but none in Plant 2. Product 2 needs only Plants 2 and 3. The marketing division has concluded that the company could sell as much of either product as could be produced by these plants. However, because both products would be competing for the same production capacity in Plant 3, it is not clear which *mix* of the two products would be *most profitable*. Therefore, an OR team has been formed to study this question.

The OR team began by having discussions with upper management to identify management's objectives for the study. These discussions led to developing the following definition of the problem:

> Determine what the *production rates* should be for the two products in order to *maximize their total profit,* subject to the restrictions imposed by the limited production capacities available in the three plants. (Each product will be produced in batches of 20, so the

An Application Vignette

Swift & Company is a diversified protein-producing business based in Greeley, Colorado. With annual sales of over $8 billion, beef and related products are by far the largest portion of the company's business.

To improve the company's sales and manufacturing performance, upper management concluded that it needed to achieve three major objectives. One was to enable the company's customer service representatives to talk to their more than 8,000 customers with accurate information about the availability of current and future inventory while considering requested delivery dates and maximum product age upon delivery. A second was to produce an efficient shift-level schedule for each plant over a 28-day horizon. A third was to accurately determine whether a plant can ship a requested order-line-item quantity on the requested date and time given the availability of cattle and constraints on the plant's capacity.

To meet these three challenges, an OR team developed an *integrated system of 45 linear programming models* based on three model formulations to dynamically schedule its beef-fabrication operations at five plants in real time as it receives orders. *The total audited benefits realized in the first year* of operation of this system *were* **$12.74 million,** including $12 million due to *optimizing the product mix.* Other benefits include a reduction in orders lost, a reduction in price discounting, and better on-time delivery.

Source: A. Bixby, B. Downs, and M. Self, "A Scheduling and Capable-to-Promise Application for Swift & Company," *Interfaces,* **36**(1): 39–50, Jan.–Feb. 2006. (A link to this article is provided on our website, www.mhhe.com/hillier.)

production rate is defined as the number of batches produced per week.) *Any* combination of production rates that satisfies these restrictions is permitted, including producing none of one product and as much as possible of the other.

The OR team also identified the data that needed to be gathered:

1. Number of hours of production time available per week in each plant for these new products. (Most of the time in these plants already is committed to current products, so the available capacity for the new products is quite limited.)
2. Number of hours of production time used in each plant for each batch produced of each new product.
3. Profit per batch produced of each new product. (*Profit per batch produced* was chosen as an appropriate measure after the team concluded that the incremental profit from each additional batch produced would be roughly *constant* regardless of the total number of batches produced. Because no substantial costs will be incurred to initiate the production and marketing of these new products, the total profit from each one is approximately this *profit per batch produced* times *the number of batches produced.*)

Obtaining reasonable estimates of these quantities required enlisting the help of key personnel in various units of the company. Staff in the manufacturing division provided the 1 data in the first category above. Developing estimates for the second category of data required some analysis by the manufacturing engineers involved in designing the production processes for the new products. By analyzing cost data from these same engineers and the marketing division, along with a pricing decision from the marketing division, the accounting department developed estimates for the third category.

Table 3.1 summarizes the data gathered.

The OR team immediately recognized that this was a linear programming problem of the classic **product mix** type, and the team next undertook the formulation of the corresponding mathematical model.

TABLE 3.1 Data for the Wyndor Glass Co. problem

	Production Time per Batch, Hours		
	Product		
Plant	**1**	**2**	**Production Time Available per Week, Hours**
1	1	0	4
2	0	2	12
3	3	2	18
Profit per batch	\$3,000	\$5,000	

Formulation as a Linear Programming Problem

The definition of the problem given above indicates that the decisions to be made are the number of batches of the respective products to be produced per week so as to maximize their total profit. Therefore, to formulate the mathematical (linear programming) model for this problem, let

x_1 = number of batches of product 1 produced per week

x_2 = number of batches of product 2 produced per week

Z = total profit per week (in thousands of dollars) from producing these two products

Thus, x_1 and x_2 are the *decision variables* for the model. Using the bottom row of Table 3.1, we obtain

$$Z = 3x_1 + 5x_2.$$

The objective is to choose the values of x_1 and x_2 so as to *maximize* $Z = 3x_1 + 5x_2$, subject to the restrictions imposed on their values by the limited production capacities available in the three plants. Table 3.1 indicates that each batch of product 1 produced per week uses 1 hour of production time per week in Plant 1, whereas only 4 hours per week are available. This restriction is expressed mathematically by the inequality $x_1 \leq 4$. Similarly, Plant 2 imposes the restriction that $2x_2 \leq 12$. The number of hours of production time used per week in Plant 3 by choosing x_1 and x_2 as the new products' production rates would be $3x_1 + 2x_2$. Therefore, the mathematical statement of the Plant 3 restriction is $3x_1 + 2x_2 \leq 18$. Finally, since production rates cannot be negative, it is necessary to restrict the decision variables to be nonnegative: $x_1 \geq 0$ and $x_2 \geq 0$.

To summarize, in the mathematical language of linear programming, the problem is to choose values of x_1 and x_2 so as to

$$\text{Maximize} \quad Z = 3x_1 + 5x_2,$$

subject to the restrictions

$$\begin{aligned} x_1 \quad\quad &\leq 4 \\ 2x_2 &\leq 12 \\ 3x_1 + 2x_2 &\leq 18 \end{aligned}$$

and

$$x_1 \geq 0, \quad x_2 \geq 0.$$

(Notice how the layout of the coefficients of x_1 and x_2 in this linear programming model essentially duplicates the information summarized in Table 3.1.)

This problem is a classic example of a **resource-allocation problem**, the most common type of linear programming problem. The key characteristic of resource-allocation problems is that most or all of their functional constraints are *resource constraints*. The right-hand side of a resource constraint represents the amount available of some resource and the left-hand side represents the amount used of that resource, so the left-hand side must be ≤ the right-hand side. Product-mix problems are one type of resource-allocation problem, but you will see examples of resource-allocations problems of other types in Sec. 3.4 along with examples of other categories of linear programming problems.

Graphical Solution

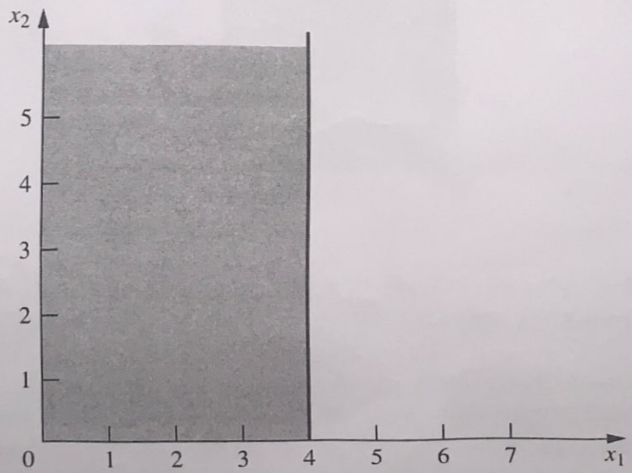
This very small problem has only two decision variables and therefore only two dimensions, so a graphical procedure can be used to solve it. This procedure involves constructing a two-dimensional graph with x_1 and x_2 as the axes. The first step is to identify the values of (x_1, x_2) that are permitted by the restrictions. This is done by drawing each line that borders the range of permissible values for one restriction. To begin, note that the nonnegativity restrictions $x_1 \geq 0$ and $x_2 \geq 0$ require (x_1, x_2) to lie on the *positive* side of the axes (including actually *on* either axis), i.e., in the first quadrant. Next, observe that the restriction $x_1 \leq 4$ means that (x_1, x_2) cannot lie to the right of the line $x_1 = 4$. These results are shown in Fig. 3.1, where the shaded area contains the only values of (x_1, x_2) that are still allowed.

In a similar fashion, the restriction $2x_2 \leq 12$ (or, equivalently, $x_2 \leq 6$) implies that the line $2x_2 = 12$ should be added to the boundary of the permissible region. The final restriction, $3x_1 + 2x_2 \leq 18$, requires plotting the points (x_1, x_2) such that $3x_1 + 2x_2 = 18$ (another line) to complete the boundary. (Note that the points such that $3x_1 + 2x_2 \leq 18$ are those that lie either underneath or on the line $3x_1 + 2x_2 = 18$, so this is the limiting line above which points do not satisfy the inequality.) The resulting region of permissible values of (x_1, x_2), called the **feasible region,** is shown in Fig. 3.2. (The demo called *Graphical Method* in your OR Tutor provides a more detailed example of constructing a feasible region.)

?

The final step is to pick out the point in this feasible region that maximizes the value of $Z = 3x_1 + 5x_2$. To discover how to perform this step efficiently, begin by trial and error. Try, for example, $Z = 10 = 3x_1 + 5x_2$ to see if there are in the permissible region any values of (x_1, x_2) that yield a value of Z as large as 10. By drawing the line $3x_1 + 5x_2 = 10$ (see Fig. 3.3), you can see that there are many points on this line that lie within the region. Having gained perspective by trying this arbitrarily chosen value of $Z = 10$, you should

FIGURE 3.1
Shaded area shows values of (x_1, x_2) allowed by $x_1 \geq 0$, $x_2 \geq 0$, $x_1 \leq 4$.



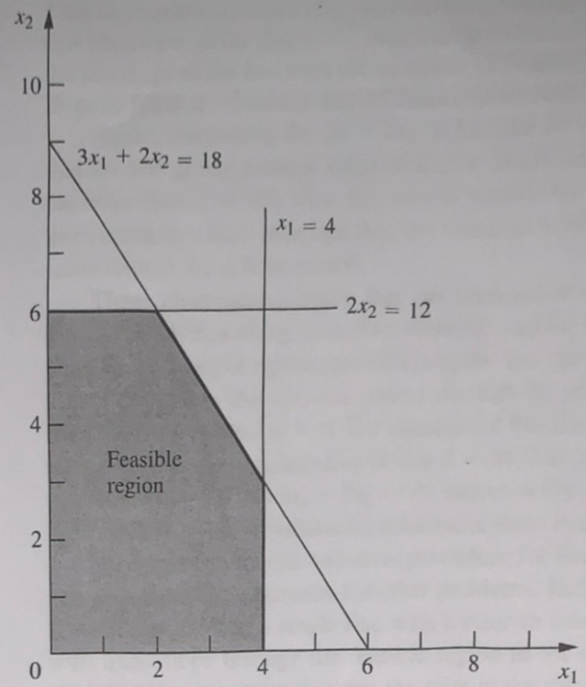


FIGURE 3.2
Shaded area shows the set of permissible values of (x_1, x_2), called the feasible region.

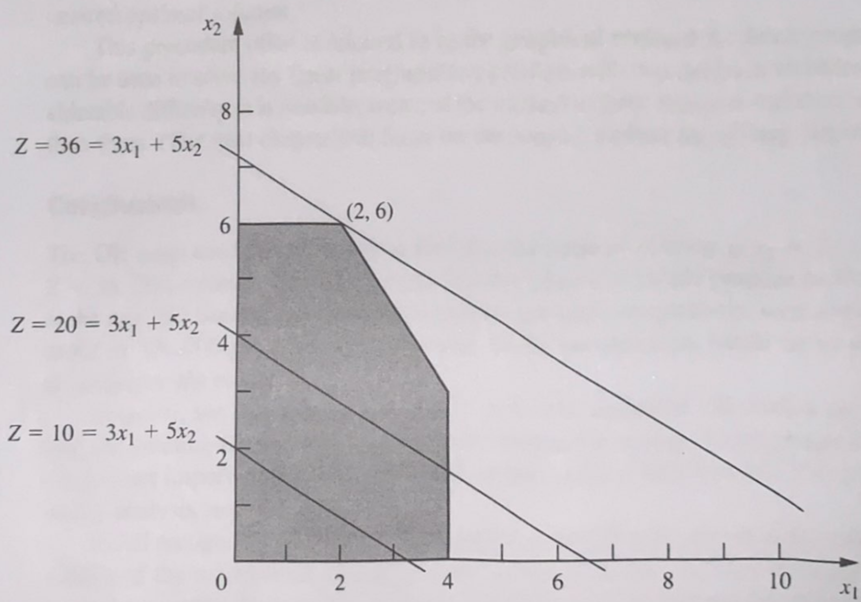


FIGURE 3.3
The value of (x_1, x_2) that maximizes $3x_1 + 5x_2$ is (2, 6).

next try a larger arbitrary value of Z, say, $Z = 20 = 3x_1 + 5x_2$. Again, Fig. 3.3 reveals that a segment of the line $3x_1 + 5x_2 = 20$ lies within the region, so that the maximum permissible value of Z must be at least 20.

Now notice in Fig. 3.3 that the two lines just constructed are parallel. This is no coincidence, since *any* line constructed in this way has the form $Z = 3x_1 + 5x_2$ for the chosen value of Z, which implies that $5x_2 = -3x_1 + Z$ or, equivalently,

$$x_2 = -\frac{3}{5}x_1 + \frac{1}{5}Z$$

This last equation, called the **slope-intercept form** of the objective function, demonstrates that the *slope* of the line is $-\frac{3}{5}$ (since each unit increase in x_1 changes x_2 by $-\frac{3}{5}$), whereas the *intercept* of the line with the x_2 axis is $\frac{1}{5}Z$ (since $x_2 = \frac{1}{5}Z$ when $x_1 = 0$). The fact that the slope is fixed at $-\frac{3}{5}$ means that *all* lines constructed in this way are parallel.

Again, comparing the $10 = 3x_1 + 5x_2$ and $20 = 3x_1 + 5x_2$ lines in Fig. 3.3, we note that the line giving a larger value of Z ($Z = 20$) is farther up and away from the origin than the other line ($Z = 10$). This fact also is implied by the slope-intercept form of the objective function, which indicates that the intercept with the x_1 axis ($\frac{1}{5}Z$) increases when the value chosen for Z is increased.

These observations imply that our trial-and-error procedure for constructing lines in Fig. 3.3 involves nothing more than drawing a family of parallel lines containing at least one point in the feasible region and selecting the line that corresponds to the largest value of Z. Figure 3.3 shows that this line passes through the point (2, 6), indicating that the **optimal solution** is $x_1 = 2$ and $x_2 = 6$. The equation of this line is $3x_1 + 5x_2 = 3(2) + 5(6) = 36 = Z$, indicating that the optimal value of Z is $Z = 36$. The point (2, 6) lies at the intersection of the two lines $2x_2 = 12$ and $3x_1 + 2x_2 = 18$, shown in Fig. 3.2, so that this point can be calculated algebraically as the simultaneous solution of these two equations.

Having seen the trial-and-error procedure for finding the optimal point (2, 6), you now can streamline this approach for other problems. Rather than draw several parallel lines, it is sufficient to form a single line with a ruler to establish the slope. Then move the ruler with fixed slope through the feasible region in the direction of improving Z. (When the objective is to *minimize* Z, move the ruler in the direction that *decreases* Z.) Stop moving the ruler at the last instant that it still passes through a point in this region. This point is the desired *optimal solution.*

This procedure often is referred to as the **graphical method** for linear programming. It can be used to solve any linear programming problem with two decision variables. With considerable difficulty, it is possible to extend the method to three decision variables but not more than three. (The next chapter will focus on the *simplex method* for solving larger problems.)

Conclusions

The OR team used this approach to find that the optimal solution is $x_1 = 2$, $x_2 = 6$, with $Z = 36$. This solution indicates that the Wyndor Glass Co. should produce products 1 and 2 at the rate of 2 batches per week and 6 batches per week, respectively, with a resulting total profit of \$36,000 per week. No other mix of the two products would be so profitable—*according to the model.*

However, we emphasized in Chap. 2 that well-conducted OR studies do not simply find *one* solution for the *initial* model formulated and then stop. All six phases described in Chap. 2 are important, including thorough testing of the model (see Sec. 2.4) and postoptimality analysis (see Sec. 2.3).

In full recognition of these practical realities, the OR team now is ready to evaluate the validity of the model more critically (to be continued in Sec. 3.3) and to perform sensitivity analysis on the effect of the estimates in Table 3.1 being different because of inaccurate estimation, changes of circumstances, etc. (to be continued in Sec. 7.2).

Continuing the Learning Process with Your OR Courseware

This is the first of many points in the book where you may find it helpful to use your *OR Courseware* on the book's website. A key part of this courseware is a program called **OR Tutor.** This program includes a complete demonstration example of the *graphical method* introduced in this section. To provide you with **another example** of a model formulation as well, this demonstration begins by introducing a problem and formulating a linear programming model for the problem before then applying the graphical method step by step to solve the model. Like the many other demonstration examples accompanying other sections of the book, this computer demonstration highlights concepts that are difficult to convey on the printed page. You may refer to Appendix 1 for documentation of the software.

If you would like to see still **more examples,** you can go to the **Solved Examples** section of the book's website. This section includes a few examples with complete solutions for almost every chapter as a supplement to the examples in the book and in OR Tutor. The examples for the current chapter begin with a relatively straightforward problem that involves formulating a small linear programming model and applying the graphical method. The subsequent examples become progressively more challenging.

Another key part of your OR Courseware is a program called **IOR Tutorial.** This program features many interactive procedures for interactively executing various solution methods presented in the book, which enables you to focus on learning and executing the logic of the method efficiently while the computer does the number crunching. Included is an interactive procedure for applying the graphical method for linear programming. Once you get the hang of it, a second procedure enables you to quickly apply the graphical method for performing sensitivity analysis on the effect of revising the data of the problem. You then can print out your work and results for your homework. Like the other procedures in IOR Tutorial, these procedures are designed specifically to provide you with an efficient, enjoyable, and enlightening learning experience while you do your homework.

When you formulate a linear programming model with more than two decision variables (so the graphical method cannot be used), the *simplex method* described in Chap. 4 enables you to still find an optimal solution immediately. Doing so also is helpful for *model validation,* since finding a *nonsensical* optimal solution signals that you have made a mistake in formulating the model.

We mentioned in Sec. 1.5 that your OR Courseware introduces you to four particularly popular commercial software packages—Excel with its Solver, a powerful Excel add-in called Analytical Solver Platform, LINGO/LINDO, and MPL/Solvers—for solving a variety of OR models. All four packages include the simplex method for solving linear programming models. Section 3.5 describes how to use Excel to formulate and solve linear programming models in a spreadsheet format with either Solver or Analytical Solver Platform for Education (ASPE), descriptions of the other packages are provided in Sec. 3.6 (MPL and LINGO), Supplements 1 and 2 to this chapter on the book's website (LINGO), Sec. 4.8 (LINDO and various solvers of MPL), and Appendix 4.1 (LINGO and LINDO). MPL, LINGO, and LINDO tutorials also are provided on the book's website. In addition, your OR Courseware includes an Excel file, a LINGO/LINDO file, and an MPL/Solvers file showing how the respective software packages can be used to solve each of the examples in this chapter.

3.2 THE LINEAR PROGRAMMING MODEL

The Wyndor Glass Co. problem is intended to illustrate a typical linear programming problem (miniature version). However, linear programming is too versatile to be completely characterized by a single example. In this section we discuss the general characteristics of linear programming problems, including the various legitimate forms of the mathematical model for linear programming.

Let us begin with some basic terminology and notation. The first column of Table 3.2 summarizes the components of the Wyndor Glass Co. problem. The second column then introduces more general terms for these same components that will fit many linear programming problems. The key terms are *resources* and *activities,* where m denotes the number of different kinds of resources that can be used and n denotes the number of activities being considered. Some typical resources are money and particular kinds of machines,