12
Integer Programming

In Chap. 3 you saw several examples of the numerous and diverse applications of linear programming. However, one key limitation that prevents many more applications is the assumption of divisibility (see Sec. 3.3), which requires that noninteger values be permissible for decision variables. In many practical problems, the decision variables actually make sense only if they have integer values. For example, it is often necessary to assign people, machines, and vehicles to activities in integer quantities. If requiring integer values is the only way in which a problem deviates from a linear programming formulation, then it is an *integer programming* (**IP**) problem. (The more complete name is *integer linear programming,* but the adjective *linear* normally is dropped except when this problem is contrasted with the more esoteric integer nonlinear programming problem, which is beyond the scope of this book.)

The mathematical model for integer programming is the linear programming model (see Sec. 3.2) with the one additional restriction that the variables must have integer values. If only *some* of the variables are required to have integer values (so the divisibility assumption holds for the rest), this model is referred to as **mixed integer programming (MIP).** When distinguishing the all-integer problem from this mixed case, we call the former *pure* integer programming.

For example, the Wyndor Glass Co. problem presented in Sec. 3.1 actually would have been an IP problem if the two decision variables x_1 and x_2 had represented the total number of units to be produced of products 1 and 2, respectively, instead of the production rates. Because both products (glass doors and wood-framed windows) necessarily come in whole units, x_1 and x_2 would have to be restricted to integer values.

Another example of an IP problem is provided by the prize-winning OR study done for the **San Francisco Police Department** that we introduced (and referenced) in Sec. 2.1. As indicated there, this study resulted in the development of a computerized system for optimally scheduling and deploying police patrol officers. The new system provided annual savings of $11 million, an annual $3 million increase in traffic citation revenues, and a 20 percent improvement in response times. The main decision variables in the mathematical model were the number of officers to schedule to go on duty at each of the shift start times. Since this number had to be an integer, these decision variables were restricted to having integer values.

There have been numerous such applications of integer programming that involve a direct extension of linear programming where the divisibility assumption must be dropped. However, another area of application may be of even greater importance, namely, problems involving a number of interrelated "yes-or-no decisions." In such decisions, the only two possible choices are *yes* and *no*. For example, should we undertake a particular fixed project? Should we make a particular fixed investment? Should we locate a facility in a particular site?

With just two choices, we can represent such decisions by decision variables that are restricted to just two values, say 0 and 1. Thus, the jth yes-or-no decision would be represented by, say, x_j such that

$$x_j = \begin{cases} 1 & \text{if decision } j \text{ is yes} \\ 0 & \text{if decision } j \text{ is no.} \end{cases}$$

Such variables are called **binary variables** (or 0–1 variables). Consequently, IP problems that contain only binary variables sometimes are called **binary integer programming (BIP)** problems (or 0–1 integer programming problems).

Section 12.1 presents a miniature version of a typical BIP problem and Sec. 12.2 surveys a variety of other BIP applications. Additional formulation possibilities with binary variables are discussed in Sec. 12.3, and Sec. 12.4 presents a series of formulation examples. The remaining sections then deal with ways to solve IP problems, including both BIP and MIP problems.

12.1 PROTOTYPE EXAMPLE

The CALIFORNIA MANUFACTURING COMPANY is considering expansion by building a new factory in either Los Angeles or San Francisco, or perhaps even in both cities. It also is considering building at most one new warehouse, but the choice of location is restricted to a city where a new factory is being built. The *net present value* (total profitability considering the time value of money) of each of these alternatives is shown in the fourth column of Table 12.1. The rightmost column gives the capital required (already included in the net present value) for the respective investments, where the total capital available is \$10 million. The objective is to find the feasible combination of alternatives that maximizes the total net present value.

TABLE 12.1 Data for the California Manufacturing Co. example

Decision Number	Yes-or-No Question	Decision Variable	Net Present Value	Capital Required
1	Build factory in Los Angeles?	x_1	\$9 million	\$6 million
2	Build factory in San Francisco?	x_2	\$5 million	\$3 million
3	Build warehouse in Los Angeles?	x_3	\$6 million	\$5 million
4	Build warehouse in San Francisco?	x_4	\$4 million	\$2 million
			Capital available:	\$10 million

The BIP Model

Although this problem is small enough that it can be solved very quickly by inspection (build factories in both cities but no warehouse), let us formulate the IP model for illustrative purposes. All the decision variables have the *binary* form

$$x_j = \begin{cases} 1 & \text{if decision } j \text{ is yes,} \\ 0 & \text{if decision } j \text{ is no,} \end{cases} \qquad (j = 1, 2, 3, 4).$$

Let

Z = total net present value of these decisions.

If the investment is made to build a particular facility (so that the corresponding decision variable has a value of 1), the estimated net present value from that investment is given in the fourth column of Table 12.1. If the investment is not made (so the decision variable equals 0), the net present value is 0. Therefore, using units of millions of dollars,

$$Z = 9x_1 + 5x_2 + 6x_3 + 4x_4.$$

The rightmost column of Table 12.1 indicates that the amount of capital expended on the four facilities cannot exceed \$10 million. Consequently, continuing to use units of millions of dollars, one constraint in the model is

$$6x_1 + 3x_2 + 5x_3 + 2x_4 \leq 10.$$

Because the last two decisions represent *mutually exclusive alternatives* (the company wants *at most* one new warehouse), we also need the constraint

$$x_3 + x_4 \leq 1.$$

Furthermore, decisions 3 and 4 are *contingent decisions,* because they are contingent on decisions 1 and 2, respectively (the company would consider building a warehouse in a city only if a new factory also were going there). Thus, in the case of decision 3, we require that $x_3 = 0$ if $x_1 = 0$. This restriction on x_3 (when $x_1 = 0$) is imposed by adding the constraint

$$x_3 \leq x_1.$$

Similarly, the requirement that $x_4 = 0$ if $x_2 = 0$ is imposed by adding the constraint

$$x_4 \leq x_2.$$

Therefore, after we rewrite these two constraints to bring all variables to the left-hand side, the complete BIP model is

$$\text{Maximize} \qquad Z = 9x_1 + 5x_2 + 6x_3 + 4x_4,$$

subject to

$$\begin{aligned} 6x_1 + 3x_2 + 5x_3 + 2x_4 &\leq 10 \\ x_3 + x_4 &\leq 1 \\ -x_1 + x_3 &\leq 0 \\ -x_2 + x_4 &\leq 0 \\ x_j &\leq 1 \\ x_j &\geq 0 \end{aligned}$$

and

x_j is integer, for $j = 1, 2, 3, 4$.

Equivalently, the last three lines of this model can be replaced by the single restriction

x_j is binary, for $j = 1, 2, 3, 4$.

Except for its small size, this example is typical of many real applications of integer programming where the basic decisions to be made are of the yes-or-no type. Like the second pair of decisions for this example, groups of yes-or-no decisions often constitute groups of **mutually exclusive alternatives** such that *only one* decision in the group can be yes. Each group requires a constraint that the sum of the corresponding binary variables must be equal to 1 (if *exactly one* decision in the group must be yes) or less than or equal to 1 (if *at most one* decision in the group can be yes). Occasionally, decisions of the yes-or-no type are **contingent decisions**, i.e., decisions that depend upon previous decisions. For example, one decision is said to be *contingent* on another decision if it is allowed to be yes *only if* the other is yes. This situation occurs when the contingent decision involves a follow-up action that would become irrelevant, or even impossible, if the other decision were no. The form that the resulting constraint takes always is that illustrated by the third and fourth constraints in the example.

LINGO

Software Options for Solving Such Models

All the software packages featured in your OR Courseware (Excel, LINGO/LINDO, and MPL/CPLEX) include an algorithm for solving (pure or mixed) BIP models, as well as an algorithm for solving general (pure or mixed) IP models where variables need to be integer but not binary.

When using the Excel Solver, the procedure is basically the same as for linear programming. The one difference arises when you click on the "Add" button on the Solver dialogue box to add the constraints. In addition to the constraints that fit linear programming, you also need to add the integer constraints. In the case of integer variables that are not binary, this is accomplished in the Add Constraint dialogue box by choosing the range of integer-restricted variables on the left-hand side and then choosing "int" from the pop-up menu. In the case of binary variables, choose "bin" from the pop-up menu instead. (In earlier versions of Excel that do not include the "bin" option, choose "int" and then add ≥ 0 and ≤ 1 constraints on these binary variables.)

A LINGO model uses the function @BIN() to specify that the variable named inside the parentheses is a binary variable. For a *general* integer variable (one restricted to integer values but not just binary values), the function @GIN() is used in the same way. In either case, the function can be embedded inside an @FOR statement to impose this binary or integer constraint on an entire set of variables.

In a LINDO model, the binary or integer constraints are inserted after the END statement. A variable X is specified to be a general integer variable by entering GIN X. Alternatively, for any positive integer value of *n,* the statement GIN *n* specifies that the first *n* variables are general integer variables. Binary variables are handled in the same way except for substituting the word INTEGER for GIN.

For an MPL model, the keyword INTEGER is used to designate general integer variables, whereas BINARY is used for binary variables. In the variables section of an MPL

model, all you need to do is add the appropriate adjective (INTEGER or BINARY) in front of the label VARIABLES to specify that the set of variables listed below the label is of that type. Alternatively, you can ignore this specification in the variables section and instead place the integer or binary constraints in the model section anywhere after the other constraints. In this case, the label over the set of variables becomes just INTEGER or BINARY.

MPL

The prime MPL solver CPLEX includes state-of-the-art algorithms for solving pure or mixed IP or BIP models. By selecting *MIP Strategy* from the *CPLEX Parameters* submenu in the *Options* menu, an experienced practitioner can even choose from a wide variety of options for exactly how to execute the algorithm to best fit the particular problem.

These instructions for how to use the various software packages become clearer when you see them applied to examples. The Excel, LINGO/LINDO, and MPL/CPLEX files for this chapter in your OR Courseware show how each of these software options would be applied to the prototype example introduced in this section, as well as to the subsequent IP examples.

The latter part of the chapter will focus on IP algorithms that are similar to those used in these software packages. Section 12.6 will use the prototype example to illustrate the application of the pure BIP algorithm presented there.

12.2 SOME BIP APPLICATIONS

Just as in the California Manufacturing Co. example, managers frequently must face *yes-or-no decisions.* Therefore, *binary integer programming* (BIP) is widely used to aid in these decisions.

We now will introduce various types of yes-or-no decisions. We also will mention some examples of actual applications where BIP was used to address these decisions.

Each of these applications is fully described in an article in the journal called *Interfaces.* In each case, we will mention the specific issue in which the article appears in case you want to read further.

Capital Budgeting with Fixed Investment Proposals

Linear programming sometimes is used to make capital budgeting decisions about how much to invest in various projects. However, as the California Manufacturing Co. example demonstrates, some capital budgeting decisions do not involve *how much* to invest, but rather, *whether* to invest a fixed amount. Specifically, the four decisions in the example were whether to invest the fixed amount of capital required to build a certain kind of facility (factory or warehouse) in a certain location (Los Angeles or San Francisco).

Management often must face decisions about whether to make fixed investments (those where the amount of capital required has been fixed in advance). Should we acquire a certain subsidiary being spun off by another company? Should we purchase a certain source of raw materials? Should we add a new production line to produce a certain input item ourselves rather than continuing to obtain it from a supplier?

In general, capital budgeting decisions about fixed investments are yes-or-no decisions of the following type.

Each yes-or-no decision:
Should we make a certain fixed investment?

$$\text{Its decision variable} = \begin{cases} 1 & \text{if yes} \\ 0 & \text{if no.} \end{cases}$$

The July–August 1990 issue of *Interfaces* describes how the *Turkish Petroleum Refineries Corporation* used BIP to analyze capital investments worth tens of millions of dollars to expand refinery capacity and conserve energy.

A rather different example that still falls somewhat into this category is described in the January–February 1997 issue of *Interfaces.* A major OR study was conducted for the *South African National Defense Force* to upgrade its capabilities with a smaller budget. The "investments" under consideration in this case were acquisition costs and ongoing expenses that would be required to provide specific types of military capabilities. A mixed BIP model was formulated to choose those specific capabilities that would maximize the overall effectiveness of the Defense Force while satisfying a budget constraint. The model had over 16,000 variables (including 256 binary variables) and over 5,000 functional constraints. The resulting optimization of the size and shape of the defense force provided savings of over $1.1 billion per year as well as vital nonmonetary benefits. The impact of this study won it the prestigious *first prize* among the 1996 Franz Edelman Awards for Management Science Achievement.

Site Selection

In this global economy, many corporations are opening up new plants in various parts of the world to take advantage of lower labor costs, etc. Before selecting a site for a new plant, many potential sites may need to be analyzed and compared. (The California Manufacturing Co. example had just two potential sites for each of two kinds of facilities.) Each of the potential sites involves a yes-or-no decision of the following type.

Each yes-or-no decision:
Should a certain site be selected for the location of a certain new facility?

$$\text{Its decision variable} = \begin{cases} 1 & \text{if yes} \\ 0 & \text{if no.} \end{cases}$$

In many cases, the objective is to select the sites so as to minimize the total cost of the new facilities that will provide the required output.

As described in the January–February 1990 issue of *Interfaces,* AT&T used a BIP model to help dozens of their customers select the sites for their telemarketing centers. The model minimizes labor, communications, and real estate costs while providing the desired level of coverage by the centers. In one year alone (1988), this approach enabled 46 AT&T customers to make their yes-or-no decisions on site locations swiftly and confidently, while committing to $375 million in annual network services and $31 million in equipment sales from AT&T.

We next describe an important type of problem for many corporations where site selection plays a key role.

Designing a Production and Distribution Network

Manufacturers today face great competitive pressure to get their products to market more quickly as well as to reduce their production and distribution costs. Therefore, any corporation that distributes its products over a wide geographical area (or even worldwide) must pay continuing attention to the design of its production and distribution network.

This design involves addressing the following kinds of yes-or-no decisions.

Should a certain plant remain open?
Should a certain site be selected for a new plant?
Should a certain distribution center remain open?
Should a certain site be selected for a new distribution center?

If each market area is to be served by a single distribution center, then we also have another kind of yes-or-no decision for each combination of a market area and a distribution center.

Should a certain distribution center be assigned to serve a certain market area?

For each of the yes-or-no decisions of any of these kinds,

$$\text{Its decision variable} = \begin{cases} 1 & \text{if yes} \\ 0 & \text{if no.} \end{cases}$$

Ault Foods Limited (July–August 1994 issue of *Interfaces*) used this approach to design its production and distribution center. Management considered 10 sites for plants, 13 sites for distribution centers, and 48 market areas. This application of BIP was credited with saving the company $200,000 per year.

Digital Equipment Corporation (January–February 1995 issue of *Interfaces*) provides another example of an application of this kind. At the time, this large multinational corporation was serving one-quarter million customer sites, with more than half of its $14 billion annual revenues coming from 81 countries outside the United States. Therefore, this application involved restructuring the corporation's entire *global supply chain,* consisting of its suppliers, plants, distribution centers, potential sites, and market areas all around the world. The restructuring generated annual cost reductions of $500 million in manufacturing and $300 million in logistics, as well as a reduction of over $400 million in required capital assets.

Dispatching Shipments

Once a production and distribution network has been designed and put into operation, daily operating decisions need to be made about how to send the shipments. Some of these decisions again are yes-or-no decisions.

For example, suppose that trucks are being used to transport the shipments and each truck typically makes deliveries to several customers during each trip. It then becomes necessary to select a route (sequence of customers) for each truck, so each candidate for a route leads to the following yes-or-no decision.

Should a certain route be selected for one of the trucks?

$$\text{Its decision variable} = \begin{cases} 1 & \text{if yes} \\ 0 & \text{if no.} \end{cases}$$

The objective would be to select the routes that would minimize the total cost of making all the deliveries.

Various complications also can be considered. For example, if different truck sizes are available, each candidate for selection would include both a certain route and a cer-

tain truck size. Similarly, if timing is an issue, a time period for the departure also can be specified as part of the yes-or-no decision. With both factors, each yes-or-no decision would have the form shown below.

Should all the following be selected simultaneously for a delivery run:

1. A certain route,
2. A certain size of truck, and
3. A certain time period for the departure?

$$\text{Its decision variable} = \begin{cases} 1 & \text{if yes} \\ 0 & \text{if no.} \end{cases}$$

Here are a few of the companies which use BIP to help make these kinds of decisions. A Michigan-based retail chain called *Quality Stores* (March–April 1987 issue of *Interfaces*) makes the routing decisions for its delivery trucks this way, thereby saving about $450,000 per year. *Air Products and Chemicals, Inc.* (December 1983 issue of *Interfaces*) saves approximately $2 million annually (about 8 percent of its prior distribution costs) by using this approach to produce its daily delivery schedules. The *Reynolds Metals Co.* (January–February 1991 issue of *Interfaces*) achieves savings of over $7 million annually with an automated dispatching system based partially on BIP for its freight shipments from over 200 plants, warehouses, and suppliers.

Scheduling Interrelated Activities

We all schedule interrelated activities in our everyday lives, even if it is just scheduling when to begin our various homework assignments. So too, managers must schedule various kinds of interrelated activities. When should we begin production for various new orders? When should we begin marketing various new products? When should we make various capital investments to expand our production capacity?

For any such activity, the decision about when to begin can be expressed in terms of a series of yes-or-no decisions, with one of these decisions for each of the possible time periods in which to begin, as shown below.

Should a certain activity begin in a certain time period?

$$\text{Its decision variable} = \begin{cases} 1 & \text{if yes} \\ 0 & \text{if no.} \end{cases}$$

Since a particular activity can begin in only one time period, the choice of the various time periods provides a group of *mutually exclusive alternatives,* so the decision variable for only one time period can have a value of 1.

For example, this approach was used to schedule the building of a series of office buildings on property adjacent to *Texas Stadium* (home of the Dallas Cowboys) over a 7-year planning horizon. In this case, the model had 49 binary decision variables, 7 for each office building corresponding to each of the 7 years in which its construction could begin. This application of BIP was credited with increasing the profit by $6.3 million. (See the October 1983 issue of *Interfaces.*)

A somewhat similar application on a vastly larger scale occurred in *China* recently (January–February 1995 issue of *Interfaces*). China was facing at least $240 billion in

new investments over a 15-year horizon to meet the energy needs of its rapidly growing economy. Shortages of coal and electricity required developing new infrastructure for transporting coal and transmitting electricity, as well as building new dams and plants for generating thermal, hydro, and nuclear power. Therefore, the Chinese State Planning Commission and the World Bank collaborated in developing a huge mixed BIP model to guide the decisions on which projects to approve and when to undertake them over the 15-year planning period to minimize the total discounted cost. It is estimated that this OR application is saving China about $6.4 billion over the 15 years.

Scheduling Asset Divestitures

This next application actually is another example of the preceding one (scheduling interrelated activities). However, rather than dealing with such activities as constructing office buildings or investing in hydroelectric plants, the activities now are *selling* (divesting) *assets* to generate income. The assets can be either *financial* assets, such as stocks and bonds, or *physical* assets, such as real estate. Given a group of assets, the problem is to determine when to sell each one to maximize the net present value of total profit from these assets while generating the desired income stream.

In this case, each yes-or-no decision has the following form.

Should a certain asset be sold in a certain time period?

$$\text{Its decision variable} = \begin{cases} 1 & \text{if yes} \\ 0 & \text{if no.} \end{cases}$$

One company that deals with these kinds of yes-or-no decisions is *Homart Development Company* (January–February 1987 issue of *Interfaces*), which ranks among the largest commercial land developers in the United States. One of its most important strategic issues is scheduling divestiture of shopping malls and office buildings. At any particular time, well over 100 assets will be under consideration for divestiture over the next 10 years. Applying BIP to guide these decisions is credited with adding $40 million of profit from the divestiture plan.

Airline Applications

The airline industry is an especially heavy user of OR throughout its operations. For example, one large consulting firm called SABRE (spun off by American Airlines) employs several hundred OR professionals solely to focus on the problem of companies involved with transportation, including especially airlines. We will mention here just two of the applications which specifically use BIP.

One is the *fleet assignment problem.* Given several different types of airplanes available, the problem is to assign a specific type to each flight leg in the schedule so as to maximize the total profit from meeting the schedule. The basic trade-off is that if the airline uses an airplane that is too small on a particular flight leg, it will leave potential customers behind, while if it uses an airplane that is too large, it will suffer the greater expense of the larger airplane to fly empty seats.

For each combination of an airplane type and a flight leg, we have the following yes-or-no decision.

Should a certain type of airplane be assigned to a certain flight leg?

$$\text{Its decision variable} = \begin{cases} 1 & \text{if yes} \\ 0 & \text{if no.} \end{cases}$$

Delta Air Lines (January–February 1994 issue of *Interfaces*) flies over 2,500 domestic flight legs every day, using about 450 airplanes of 10 different types. They use a huge integer programming model (about 40,000 functional constraints, 20,000 binary variables, and 40,000 general integer variables) to solve their fleet assignment problem each time a change is needed. This application saves Delta approximately $100 million per year.

A fairly similar application is the *crew scheduling problem.* Here, rather than assigning airplane types to flight legs, we are instead assigning sequences of flight legs to crews of pilots and flight attendants. Thus, for each feasible sequence of flight legs that leaves from a crew base and returns to the same base, the following yes-or-no decision must be made.

Should a certain sequence of flight legs be assigned to a crew?

$$\text{Its decision variable} = \begin{cases} 1 & \text{if yes} \\ 0 & \text{if no.} \end{cases}$$

The objective is to minimize the total cost of providing crews that cover each flight leg in the schedule.

American Airlines (July–August 1989 and January–February 1991 issues of *Interfaces*) achieves annual savings of over $20 million by using BIP to solve its crew scheduling problem on a monthly basis.

A full-fledged formulation example of this type will be presented at the end of Sec. 12.4.

12.3 INNOVATIVE USES OF BINARY VARIABLES IN MODEL FORMULATION

You have just seen a number of examples where the *basic decisions* of the problem are of the *yes-or-no type,* so that *binary variables* are introduced to represent these decisions. We now will look at some other ways in which binary variables can be very useful. In particular, we will see that these variables sometimes enable us to take a problem whose natural formulation is intractable and *reformulate* it as a pure or mixed IP problem.

This kind of situation arises when the original formulation of the problem fits either an IP or a linear programming format *except* for minor disparities involving combinatorial relationships in the model. By expressing these combinatorial relationships in terms of questions that must be answered yes or no, *auxiliary* binary variables can be introduced to the model to represent these yes-or-no decisions. Introducing these variables reduces the problem to an MIP problem (or a *pure* IP problem if all the original variables also are required to have integer values).

Some cases that can be handled by this approach are discussed next, where the x_j denote the *original* variables of the problem (they may be either continuous or integer variables) and the y_i denote the *auxiliary* binary variables that are introduced for the reformulation.

Either-Or Constraints

Consider the important case where a choice can be made between two constraints, so that *only one* (either one) must hold (whereas the other one can hold but is not required to do so). For example, there may be a choice as to which of two resources to use for a certain purpose, so that it is necessary for only one of the two resource availability constraints to hold mathematically. To illustrate the approach to such situations, suppose that one of the requirements in the overall problem is that

$$\begin{aligned} &\text{Either} \quad 3x_1 + 2x_2 \leq 18 \\ &\text{or} \quad\quad x_1 + 4x_2 \leq 16, \end{aligned}$$

i.e., at least one of these two inequalities must hold but not necessarily both. This requirement must be reformulated to fit it into the linear programming format where *all* specified constraints must hold. Let M be a very large positive number. Then this requirement can be rewritten as

$$\begin{aligned} &\text{Either} \quad 3x_1 + 2x_2 \leq 18 \\ &\qquad\quad\;\; x_1 + 4x_2 \leq 16 + M \\ &\text{or} \quad\quad 3x_1 + 2x_2 \leq 18 + M \\ &\qquad\quad\;\; x_1 + 4x_2 \leq 16. \end{aligned}$$

The key is that adding M to the right-hand side of such constraints has the effect of eliminating them, because they would be satisfied automatically by any solutions that satisfy the other constraints of the problem. (This formulation assumes that the set of feasible solutions for the overall problem is a bounded set and that M is large enough that it will not eliminate any feasible solutions.) This formulation is equivalent to the set of constraints

$$\begin{aligned} 3x_1 + 2x_2 &\leq 18 + My \\ x_1 + 4x_2 &\leq 16 + M(1 - y). \end{aligned}$$

Because the *auxiliary variable* y must be either 0 or 1, this formulation guarantees that one of the original constraints must hold while the other is, in effect, eliminated. This new set of constraints would then be appended to the other constraints in the overall model to give a pure or mixed IP problem (depending upon whether the x_j are integer or continuous variables).

This approach is related directly to our earlier discussion about expressing combinatorial relationships in terms of questions that must be answered yes or no. The combinatorial relationship involved concerns the combination of the *other* constraints of the model with the *first* of the two *alternative* constraints and then with the *second.* Which of these two combinations of constraints is *better* (in terms of the value of the objective function that then can be achieved)? To rephrase this question in yes-or-no terms, we ask two complementary questions:

1. Should $x_1 + 4x_2 \leq 16$ be selected as the constraint that must hold?
2. Should $3x_1 + 2x_2 \leq 18$ be selected as the constraint that must hold?

Because exactly one of these questions is to be answered affirmatively, we let the binary terms y and $1 - y$, respectively, represent these yes-or-no decisions. Thus, $y = 1$ if the an-

swer is yes to the first question (and no to the second), whereas $1 - y = 1$ (that is, $y = 0$) if the answer is yes to the second question (and no to the first). Since $y + 1 - y = 1$ (one yes) automatically, there is no need to add another constraint to force these two decisions to be mutually exclusive. (If separate binary variables y_1 and y_2 had been used instead to represent these yes-or-no decisions, then an additional constraint $y_1 + y_2 = 1$ would have been needed to make them mutually exclusive.)

A formal presentation of this approach is given next for a more general case.

K out of *N* Constraints Must Hold

Consider the case where the overall model includes a set of N possible constraints such that only some K of these constraints *must* hold. (Assume that $K < N$.) Part of the optimization process is to choose the *combination* of K constraints that permits the objective function to reach its best possible value. The $N - K$ constraints *not* chosen are, in effect, eliminated from the problem, although feasible solutions might coincidentally still satisfy some of them.

This case is a direct generalization of the preceding case, which had $K = 1$ and $N = 2$. Denote the N possible constraints by

$$\begin{aligned} f_1(x_1, x_2, \ldots, x_n) &\le d_1 \\ f_2(x_1, x_2, \ldots, x_n) &\le d_2 \\ &\vdots \\ f_N(x_1, x_2, \ldots, x_n) &\le d_N. \end{aligned}$$

Then, applying the same logic as for the preceding case, we find that an equivalent formulation of the requirement that some K of these constraints *must* hold is

$$\begin{aligned} f_1(x_1, x_2, \ldots, x_n) &\le d_1 + My_1 \\ f_2(x_1, x_2, \ldots, x_n) &\le d_2 + My_2 \\ &\vdots \\ f_N(x_1, x_2, \ldots, x_n) &\le d_N + My_N \\ \sum_{i=1}^{N} y_i &= N - K, \end{aligned}$$

and

$$y_i \text{ is binary}, \qquad \text{for } i = 1, 2, \ldots, N,$$

where M is an extremely large positive number. For each binary variable y_i ($i = 1, 2, \ldots, N$), note that $y_i = 0$ makes $My_i = 0$, which reduces the new constraint i to the original constraint i. On the other hand, $y_i = 1$ makes $(d_i + My_i)$ so large that (again assuming a bounded feasible region) the new constraint i is automatically satisfied by any solution that satisfies the other new constraints, which has the effect of eliminating the original constraint i. Therefore, because the constraints on the y_i guarantee that K of these variables will equal 0 and those remaining will equal 1, K of the original constraints will be unchanged and the other $(N - K)$ original constraints will, in effect, be eliminated. The choice of *which* K constraints should be retained is made by applying the appropriate algorithm to the overall problem so it finds an optimal solution for *all* the variables simultaneously.

Functions with *N* Possible Values

Consider the situation where a given function is required to take on any one of N given values. Denote this requirement by

$$f(x_1, x_2, \ldots, x_n) = d_1 \quad \text{or} \quad d_2, \ldots, \quad \text{or} \quad d_N.$$

One special case is where this function is

$$f(x_1, x_2, \ldots, x_n) = \sum_{j=1}^{n} a_j x_j,$$

as on the left-hand side of a linear programming constraint. Another special case is where $f(x_1, x_2, \ldots, x_n) = x_j$ for a given value of j, so the requirement becomes that x_j must take on any one of N given values.

The equivalent IP formulation of this requirement is the following:

$$f(x_1, x_2, \ldots, x_n) = \sum_{i=1}^{N} d_i y_i$$

$$\sum_{i=1}^{N} y_i = 1$$

and

$$y_i \text{ is binary,} \qquad \text{for } i = 1, 2, \ldots, N.$$

so this new set of constraints would replace this requirement in the statement of the overall problem. This set of constraints provides an *equivalent* formulation because exactly one y_i must equal 1 and the others must equal 0, so exactly one d_i is being chosen as the value of the function. In this case, there are N yes-or-no questions being asked, namely, should d_i be the value chosen ($i = 1, 2, \ldots, N$)? Because the y_i respectively represent these *yes-or-no decisions,* the second constraint makes them *mutually exclusive alternatives.*

To illustrate how this case can arise, reconsider the Wyndor Glass Co. problem presented in Sec. 3.1. Eighteen hours of production time per week in Plant 3 currently is unused and available for the two new products *or* for certain future products that will be ready for production soon. In order to leave any remaining capacity in usable blocks for these future products, management now wants to impose the restriction that the production time used by the two current new products be 6 *or* 12 *or* 18 hours per week. Thus, the third constraint of the original model ($3x_1 + 2x_2 \leq 18$) now becomes

$$3x_1 + 2x_2 = 6 \quad \text{or} \quad 12 \quad \text{or} \quad 18.$$

In the preceding notation, $N = 3$ with $d_1 = 6$, $d_2 = 12$, and $d_3 = 18$. Consequently, management's new requirement should be formulated as follows:

$$3x_1 + 2x_2 = 6y_1 + 12y_2 + 18y_3$$
$$y_1 + y_2 + y_3 = 1$$

and

$$y_1, y_2, y_3 \text{ are binary.}$$

The overall model for this new version of the problem then consists of the original model (see Sec. 3.1) plus this new set of constraints that replaces the original third constraint. This replacement yields a very tractable MIP formulation.

The Fixed-Charge Problem

It is quite common to incur a fixed charge or setup cost when undertaking an activity. For example, such a charge occurs when a production run to produce a batch of a particular product is undertaken and the required production facilities must be set up to initiate the run. In such cases, the total cost of the activity is the sum of a variable cost related to the level of the activity and the setup cost required to initiate the activity. Frequently the variable cost will be at least roughly proportional to the level of the activity. If this is the case, the *total cost* of the activity (say, activity j) can be represented by a function of the form

$$f_j(x_j) = \begin{cases} k_j + c_j x_j & \text{if } x_j > 0 \\ 0 & \text{if } x_j = 0, \end{cases}$$

where x_j denotes the level of activity j ($x_j \geq 0$), k_j denotes the setup cost, and c_j denotes the cost for each incremental unit. Were it not for the setup cost k_j, this cost structure would suggest the possibility of a *linear programming* formulation to determine the optimal levels of the competing activities. Fortunately, even with the k_j, MIP can still be used.

To formulate the overall model, suppose that there are n activities, each with the preceding cost structure (with $k_j \geq 0$ in every case and $k_j > 0$ for some $j = 1, 2, \ldots, n$), and that the problem is to

$$\text{Minimize} \qquad Z = f_1(x_1) + f_2(x_2) + \cdots + f_n(x_n),$$

subject to

given linear programming constraints.

To convert this problem to an MIP format, we begin by posing n questions that must be answered yes or no; namely, for each $j = 1, 2, \ldots, n$, should activity j be undertaken ($x_j > 0$)? Each of these *yes-or-no decisions* is then represented by an auxiliary *binary variable* y_j, so that

$$Z = \sum_{j=1}^{n} (c_j x_j + k_j y_j),$$

where

$$y_j = \begin{cases} 1 & \text{if } x_j > 0 \\ 0 & \text{if } x_j = 0. \end{cases}$$

Therefore, the y_j can be viewed as *contingent decisions* similar to (but not identical to) the type considered in Sec. 12.1. Let M be an extremely large positive number that exceeds the maximum feasible value of any x_j ($j = 1, 2, \ldots, n$). Then the constraints

$$x_j \leq M y_j \qquad \text{for } j = 1, 2, \ldots, n$$

will ensure that $y_j = 1$ rather than 0 whenever $x_j > 0$. The one difficulty remaining is that these constraints leave y_j free to be either 0 or 1 when $x_j = 0$. Fortunately, this difficulty

is automatically resolved because of the nature of the objective function. The case where $k_j = 0$ can be ignored because y_j can then be deleted from the formulation. So we consider the only other case, namely, where $k_j > 0$. When $x_j = 0$, so that the constraints permit a choice between $y_j = 0$ and $y_j = 1$, $y_j = 0$ must yield a smaller value of Z than $y_j = 1$. Therefore, because the objective is to minimize Z, an algorithm yielding an optimal solution would always choose $y_j = 0$ when $x_j = 0$.

To summarize, the MIP formulation of the fixed-charge problem is

$$\text{Minimize} \qquad Z = \sum_{j=1}^{n} (c_j x_j + k_j y_j),$$

subject to

the original constraints, plus
$x_j - My_j \leq 0$

and

y_j is binary, for $j = 1, 2, \ldots, n$.

If the x_j also had been restricted to be integer, then this would be a *pure* IP problem.

To illustrate this approach, look again at the Nori & Leets Co. air pollution problem described in Sec. 3.4. The first of the abatement methods considered—increasing the height of the smokestacks—actually would involve a substantial *fixed charge* to get ready for *any* increase in addition to a variable cost that would be roughly proportional to the amount of increase. After conversion to the equivalent annual costs used in the formulation, this fixed charge would be \$2 million each for the blast furnaces and the open-hearth furnaces, whereas the variable costs are those identified in Table 3.14. Thus, in the preceding notation, $k_1 = 2$, $k_2 = 2$, $c_1 = 8$, and $c_2 = 10$, where the objective function is expressed in units of *millions* of dollars. Because the other abatement methods do not involve any fixed charges, $k_j = 0$ for $j = 3, 4, 5, 6$. Consequently, the new MIP formulation of this problem is

$$\text{Minimize} \qquad Z = 8x_1 + 10x_2 + 7x_3 + 6x_4 + 11x_5 + 9x_6 + 2y_1 + 2y_2,$$

subject to

the constraints given in Sec. 3.4, plus
$x_1 - My_1 \leq 0$,
$x_2 - My_2 \leq 0$,

and

y_1, y_2 are binary.

Binary Representation of General Integer Variables

Suppose that you have a pure IP problem where most of the variables are *binary* variables, but the presence of a few *general* integer variables prevents you from solving the problem by one of the very efficient BIP algorithms now available. A nice way to circumvent this difficulty is to use the *binary representation* for each of these general integer variables. Specifically, if the bounds on an integer variable x are

$$0 \leq x \leq u$$

and if N is defined as the integer such that

$$2^N \leq u < 2^{N+1},$$

then the **binary representation** of x is

$$x = \sum_{i=0}^{N} 2^i y_i,$$

where the y_i variables are (auxiliary) binary variables. Substituting this binary representation for each of the general integer variables (with a different set of auxiliary binary variables for each) thereby reduces the entire problem to a BIP model.

For example, suppose that an IP problem has just two general integer variables x_1 and x_2 along with many binary variables. Also suppose that the problem has nonnegativity constraints for both x_1 and x_2 and that the functional constraints include

$$\begin{aligned} x_1 &\leq 5 \\ 2x_1 + 3x_2 &\leq 30. \end{aligned}$$

These constraints imply that $u = 5$ for x_1 and $u = 10$ for x_2, so the above definition of N gives $N = 2$ for x_1 (since $2^2 \leq 5 < 2^3$) and $N = 3$ for x_2 (since $2^3 \leq 10 < 2^4$). Therefore, the binary representations of these variables are

$$\begin{aligned} x_1 &= y_0 + 2y_1 + 4y_2 \\ x_2 &= y_3 + 2y_4 + 4y_5 + 8y_6. \end{aligned}$$

After we substitute these expressions for the respective variables throughout all the functional constraints and the objective function, the two functional constraints noted above become

$$\begin{aligned} y_0 + 2y_1 + 4y_2 &\leq 5 \\ 2y_0 + 4y_1 + 8y_2 + 3y_3 + 6y_4 + 12y_5 + 24y_6 &\leq 30. \end{aligned}$$

Observe that each feasible value of x_1 corresponds to one of the feasible values of the vector (y_0, y_1, y_2), and similarly for x_2 and (y_3, y_4, y_5, y_6). For example, $x_1 = 3$ corresponds to $(y_0, y_1, y_2) = (1, 1, 0)$, and $x_2 = 5$ corresponds to $(y_3, y_4, y_5, y_6) = (1, 0, 1, 0)$.

For an IP problem where *all* the variables are (bounded) general integer variables, it is possible to use this same technique to reduce the problem to a BIP model. However, this is not advisable for most cases because of the explosion in the number of variables involved. Applying a good IP algorithm to the original IP model generally should be more efficient than applying a good BIP algorithm to the much larger BIP model.

In general terms, for *all* the formulation possibilities with auxiliary binary variables discussed in this section, we need to strike the same note of caution. This approach sometimes requires adding a relatively large number of such variables, which can make the model *computationally infeasible.* (Section 12.5 will provide some perspective on the sizes of IP problems that can be solved.)

12.4 SOME FORMULATION EXAMPLES

We now present a series of examples that illustrate a variety of formulation techniques with binary variables, including those discussed in the preceding sections. For the sake of clarity, these examples have been kept very small. In actual applications, these formulations typically would be just a small part of a vastly larger model.

EXAMPLE 1 Making Choices When the Decision Variables Are Continuous.

The Research and Development Division of the GOOD PRODUCTS COMPANY has developed three possible new products. However, to avoid undue diversification of the company's product line, management has imposed the following restriction.

> **Restriction 1:** From the three possible new products, *at most two* should be chosen to be produced.

Each of these products can be produced in either of two plants. For administrative reasons, management has imposed a second restriction in this regard.

> **Restriction 2:** Just one of the two plants should be chosen to be the sole producer of the new products.

The production cost per unit of each product would be essentially the same in the two plants. However, because of differences in their production facilities, the number of hours of production time needed per unit of each product might differ between the two plants. These data are given in Table 12.2, along with other relevant information, including marketing estimates of the number of units of each product that could be sold per week if it is produced. The objective is to choose the products, the plant, and the production rates of the chosen products so as to maximize total profit.

In some ways, this problem resembles a standard *product mix problem* such as the Wyndor Glass Co. example described in Sec. 3.1. In fact, if we changed the problem by dropping the two restrictions *and* by requiring each unit of a product to use the production hours given in Table 12.2 in *both plants* (so the two plants now perform different operations needed by the products), it would become just such a problem. In particular, if we let x_1, x_2, x_3 be the production rates of the respective products, the model then becomes

$$\text{Maximize} \qquad Z = 5x_1 + 7x_2 + 3x_3,$$

subject to

$$\begin{aligned} 3x_1 + 4x_2 + 2x_3 &\leq 30 \\ 4x_1 + 6x_2 + 2x_3 &\leq 40 \\ x_1 &\leq 7 \\ x_2 &\leq 5 \\ x_3 &\leq 9 \end{aligned}$$

TABLE 12.2 Data for Example 1 (the Good Products Co. problem)

	Production Time Used for Each Unit Produced			Production Time Available per Week
	Product 1	Product 2	Product 3	
Plant 1	3 hours	4 hours	2 hours	30 hours
Plant 2	4 hours	6 hours	2 hours	40 hours
Unit profit	5	7	3	(thousands of dollars)
Sales potential	7	5	9	(units per week)

and

$$x_1 \geq 0, \qquad x_2 \geq 0, \qquad x_3 \geq 0.$$

For the real problem, however, restriction 1 necessitates adding to the model the constraint

The number of strictly positive decision variables (x_1, x_2, x_3) must be ≤ 2.

This constraint does not fit into a linear or an integer programming format, so the key question is how to convert it to such a format so that a corresponding algorithm can be used to solve the overall model. If the decision variables were binary variables, then the constraint would be expressed in this format as $x_1 + x_2 + x_3 \leq 2$. However, with *continuous* decision variables, a more complicated approach involving the introduction of auxiliary binary variables is needed.

Requirement 2 necessitates replacing the first two functional constraints ($3x_1 + 4x_2 + 2x_3 \leq 30$ and $4x_1 + 6x_2 + 2x_3 \leq 40$) by the restriction

$$\begin{array}{ll} \text{Either} & 3x_1 + 4x_2 + 2x_3 \leq 30 \\ \text{Or} & 4x_1 + 6x_2 + 2x_3 \leq 40 \end{array}$$

must hold, where the choice of which constraint must hold corresponds to the choice of which plant will be used to produce the new products. We discussed in the preceding section how such an either-or constraint can be converted to a linear or an integer programming format, again with the help of an auxiliary binary variable.

Formulation with Auxiliary Binary Variables. To deal with requirement 1, we introduce three auxiliary binary variables (y_1, y_2, y_3) with the interpretation

$$y_j = \begin{cases} 1 & \text{if } x_j > 0 \text{ can hold (can produce product } j) \\ 0 & \text{if } x_j = 0 \text{ must hold (cannot produce product } j), \end{cases}$$

for $j = 1, 2, 3$. To enforce this interpretation in the model with the help of M (an extremely large positive number), we add the constraints

$$\begin{aligned} & x_1 \leq My_1 \\ & x_2 \leq My_2 \\ & x_3 \leq My_3 \\ & y_1 + y_2 + y_3 \leq 2 \\ & y_j \text{ is binary,} \qquad \text{for } j = 1, 2, 3. \end{aligned}$$

The either-or constraint and nonnegativity constraints give a *bounded* feasible region for the decision variables (so each $x_j \leq M$ throughout this region). Therefore, in each $x_j \leq My_j$ constraint, $y_j = 1$ allows any value of x_j in the feasible region, whereas $y_j = 0$ forces $x_j = 0$. (Conversely, $x_j > 0$ forces $y_j = 1$, whereas $x_j = 0$ allows either value of y_j.) Consequently, when the fourth constraint forces choosing at most two of the y_j to equal 1, this amounts to choosing at most two of the new products as the ones that can be produced.

To deal with requirement 2, we introduce another auxiliary binary variable y_4 with the interpretation

$$y_4 = \begin{cases} 1 & \text{if } 4x_1 + 6x_2 + 2x_3 \leq 40 \text{ must hold (choose Plant 2)} \\ 0 & \text{if } 3x_1 + 4x_2 + 2x_3 \leq 30 \text{ must hold (choose Plant 1).} \end{cases}$$

As discussed in Sec. 12.3, this interpretation is enforced by adding the constraints,

$$3x_1 + 4x_2 + 2x_3 \leq 30 + My_4$$
$$4x_1 + 6x_2 + 2x_3 \leq 40 + M(1 - y_4)$$
y_4 is binary.

Consequently, after we move all variables to the left-hand side of the constraints, the complete model is

$$\text{Maximize} \qquad Z = 5x_1 + 7x_2 + 3x_3,$$

subject to

$$\begin{aligned} x_1 &\leq 7 \\ x_2 &\leq 5 \\ x_3 &\leq 9 \\ x_1 - My_1 &\leq 0 \\ x_2 - My_2 &\leq 0 \\ x_3 - My_3 &\leq 0 \\ y_1 + y_2 + y_3 &\leq 2 \\ 3x_1 + 4x_2 + 2x_3 - My_4 &\leq 30 \\ 4x_1 + 6x_2 + 2x_3 + My_4 &\leq 40 + M \end{aligned}$$

and

$$x_1 \geq 0, \qquad x_2 \geq 0, \qquad x_3 \geq 0$$
y_j is binary, for $j = 1, 2, 3, 4$.

This now is an MIP model, with three variables (the x_j) not required to be integer and four binary variables, so an MIP algorithm can be used to solve the model. When this is done (after substituting a large numerical value for M),[1] the optimal solution is $y_1 = 1$, $y_2 = 0$, $y_3 = 1$, $y_4 = 1$, $x_1 = 5\frac{1}{2}$, $x_2 = 0$, and $x_3 = 9$; that is, choose products 1 and 3 to produce, choose Plant 2 for the production, and choose the production rates of $5\frac{1}{2}$ units per week for product 1 and 9 units per week for product 3. The resulting total profit is \$54,500 per week.

EXAMPLE 2 Violating Proportionality.

The SUPERSUDS CORPORATION is developing its marketing plans for next year's new products. For three of these products, the decision has been made to purchase a total of five TV spots for commercials on national television networks. The problem we will focus on is how to allocate the five spots to these three products, with a maximum of three spots (and a minimum of zero) for each product.

[1]In practice, some care is taken to choose a value for M that definitely is large enough to avoid eliminating any feasible solutions, but as small as possible otherwise in order to avoid unduly enlarging the feasible region for the LP-relaxation (and to avoid numerical instability). For this example, a careful examination of the constraints reveals that the minimum feasible value of M is $M = 9$.

Table 12.3 shows the estimated impact of allocating zero, one, two, or three spots to each product. This impact is measured in terms of the *profit* (in units of millions of dollars) from the *additional sales* that would result from the spots, considering also the cost of producing the commercial and purchasing the spots. The objective is to allocate five spots to the products so as to maximize the total profit.

This small problem can be solved easily by dynamic programming (Chap. 10) or even by inspection. (The optimal solution is to allocate two spots to product 1, no spots to product 2, and three spots to product 3.) However, we will show two different BIP formulations for illustrative purposes. Such a formulation would become necessary if this small problem needed to be incorporated into a larger IP model involving the allocation of resources to marketing activities for all the corporation's new products.

One Formulation with Auxiliary Binary Variables. A natural formulation would be to let x_1, x_2, x_3 be the number of TV spots allocated to the respective products. The contribution of each x_j to the objective function then would be given by the corresponding column in Table 12.3. However, each of these columns violates the assumption of proportionality described in Sec. 3.3. Therefore, we cannot write a *linear* objective function in terms of these integer decision variables.

Now see what happens when we introduce an *auxiliary binary variable* y_{ij} for each positive integer value of $x_i = j$ ($j = 1, 2, 3$), where y_{ij} has the interpretation

$$y_{ij} = \begin{cases} 1 & \text{if } x_i = j \\ 0 & \text{otherwise.} \end{cases}$$

(For example, $y_{21} = 0$, $y_{22} = 0$, and $y_{23} = 1$ mean that $x_2 = 3$.) The resulting *linear* BIP model is

$$\text{Maximize} \qquad Z = y_{11} + 3y_{12} + 3y_{13} + 2y_{22} + 3y_{23} - y_{31} + 2y_{32} + 4y_{33},$$

subject to

$$\begin{aligned} y_{11} + y_{12} + y_{13} &\le 1 \\ y_{21} + y_{22} + y_{23} &\le 1 \\ y_{31} + y_{32} + y_{33} &\le 1 \\ y_{11} + 2y_{12} + 3y_{13} + y_{21} + 2y_{22} + 3y_{23} + y_{31} + 2y_{32} + 3y_{33} &= 5 \end{aligned}$$

TABLE 12.3 Data for Example 2 (the Supersuds Corp. problem)

Number of TV Spots	Profit		
	Product		
	1	2	3
0	0	0	0
1	1	0	−1
2	3	2	2
3	3	3	4

and

each y_{ij} is binary.

Note that the first three functional constraints ensure that each x_i will be assigned just one of its possible values. (Here $y_{i1} + y_{i2} + y_{i3} = 0$ corresponds to $x_i = 0$, which contributes nothing to the objective function.) The last functional constraint ensures that $x_1 + x_2 + x_3 = 5$. The *linear* objective function then gives the total profit according to Table 12.3.

Solving this BIP model gives an optimal solution of

$$\begin{array}{llllll} y_{11} = 0, & y_{12} = 1, & y_{13} = 0, & \text{so} & x_1 = 2 \\ y_{21} = 0, & y_{22} = 0, & y_{23} = 0, & \text{so} & x_2 = 0 \\ y_{31} = 0, & y_{32} = 0, & y_{33} = 1, & \text{so} & x_3 = 3. \end{array}$$

Another Formulation with Auxiliary Binary Variables. We now redefine the above auxiliary binary variables y_{ij} as follows:

$$y_{ij} = \begin{cases} 1 & \text{if } x_i \geq j \\ 0 & \text{otherwise.} \end{cases}$$

Thus, the difference is that $y_{ij} = 1$ now if $x_i \geq j$ instead of $x_i = j$. Therefore,

$$\begin{array}{llllll} x_i = 0 & \Rightarrow & y_{i1} = 0, & y_{i2} = 0, & y_{i3} = 0, \\ x_i = 1 & \Rightarrow & y_{i1} = 1, & y_{i2} = 0, & y_{i3} = 0, \\ x_i = 2 & \Rightarrow & y_{i1} = 1, & y_{i2} = 1, & y_{i3} = 0, \\ x_i = 3 & \Rightarrow & y_{i1} = 1, & y_{i2} = 1, & y_{i3} = 1, \\ \text{so } x_i = y_{i1} + y_{i2} + y_{i3} & & & & \end{array}$$

for $i = 1, 2, 3$. Because allowing $y_{i2} = 1$ is contingent upon $y_{i1} = 1$ and allowing $y_{i3} = 1$ is contingent upon $y_{i2} = 1$, these definitions are enforced by adding the constraints

$$y_{i2} \leq y_{i1} \quad \text{and} \quad y_{i3} \leq y_{i2}, \quad \text{for } i = 1, 2, 3.$$

The new definition of the y_{ij} also changes the objective function, as illustrated in Fig. 12.1 for the product 1 portion of the objective function. Since y_{11}, y_{12}, y_{13} provide the successive increments (if any) in the value of x_1 (starting from a value of 0), the coefficients of y_{11}, y_{12}, y_{13} are given by the respective *increments* in the product 1 column of Table 12.3 ($1 - 0 = 1$, $3 - 1 = 2$, $3 - 3 = 0$). These *increments* are the *slopes* in Fig. 12.1, yielding $1y_{11} + 2y_{12} + 0y_{13}$ for the product 1 portion of the objective function. Note that applying this approach to all three products still must lead to a *linear* objective function.

After we bring all variables to the left-hand side of the constraints, the resulting complete BIP model is

$$\text{Maximize} \quad Z = y_{11} + 2y_{12} + 2y_{22} + y_{23} - y_{31} + 3y_{32} + 2y_{33},$$

subject to

$$\begin{aligned} y_{12} - y_{11} &\leq 0 \\ y_{13} - y_{12} &\leq 0 \\ y_{22} - y_{21} &\leq 0 \\ y_{23} - y_{22} &\leq 0 \end{aligned}$$

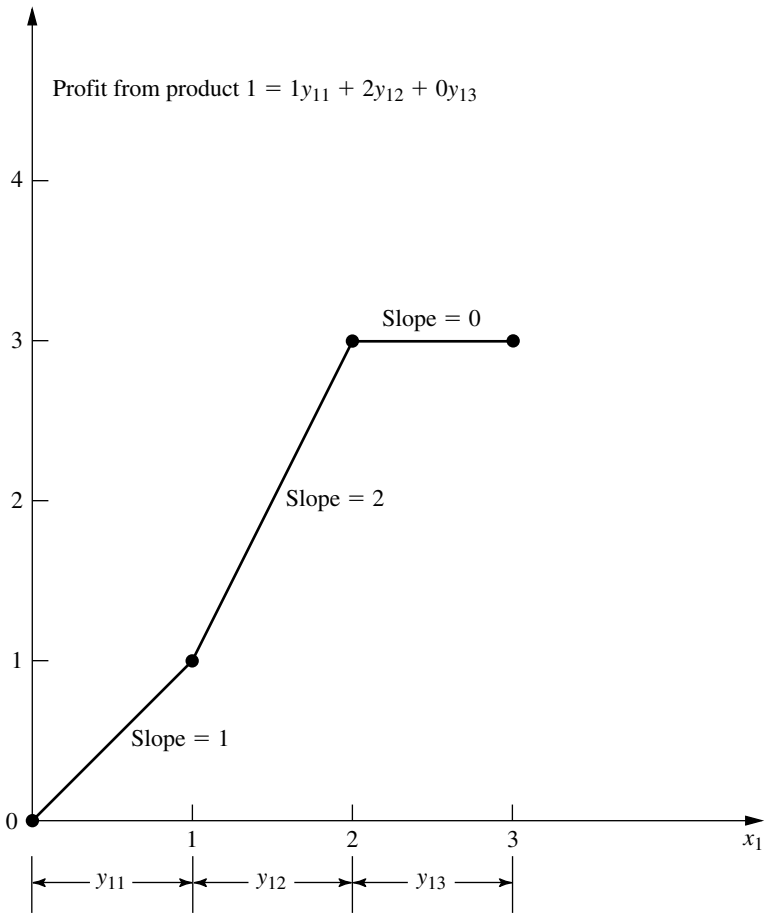


FIGURE 12.1
The profit from the additional sales of product 1 that would result from x_1 TV spots, where the slopes give the corresponding coefficients in the objective function for the second BIP formulation for Example 2 (the Supersuds Corp. problem).

$$y_{32} - y_{31} \leq 0$$
$$y_{33} - y_{32} \leq 0$$
$$y_{11} + y_{12} + y_{13} + y_{21} + y_{22} + y_{23} + y_{31} + y_{32} + y_{33} = 5$$

and

each y_{ij} is binary.

Solving this BIP model gives an optimal solution of

$$y_{11} = 1, \quad y_{12} = 1, \quad y_{13} = 0, \quad \text{so} \quad x_1 = 2$$
$$y_{21} = 0, \quad y_{22} = 0, \quad y_{23} = 0, \quad \text{so} \quad x_2 = 0$$
$$y_{31} = 1, \quad y_{32} = 1, \quad y_{33} = 1, \quad \text{so} \quad x_3 = 3.$$

There is little to choose between this BIP model and the preceding one other than personal taste. They have the same number of binary variables (the prime consideration in determining computational effort for BIP problems). They also both have some *special*

structure (constraints for *mutually exclusive alternatives* in the first model and constraints for *contingent decisions* in the second) that can lead to speedup. The second model does have more functional constraints than the first.

EXAMPLE 3 Covering All Characteristics.

SOUTHWESTERN AIRWAYS needs to assign its crews to cover all its upcoming flights. We will focus on the problem of assigning three crews based in San Francisco to the flights listed in the first column of Table 12.4. The other 12 columns show the 12 feasible sequences of flights for a crew. (The numbers in each column indicate the order of the flights.) Exactly three of the sequences need to be chosen (one per crew) in such a way that every flight is covered. (It is permissible to have more than one crew on a flight, where the extra crews would fly as passengers, but union contracts require that the extra crews would still need to be paid for their time as if they were working.) The cost of assigning a crew to a particular sequence of flights is given (in thousands of dollars) in the bottom row of the table. The objective is to minimize the total cost of the three crew assignments that cover all the flights.

Formulation with Binary Variables. With 12 feasible sequences of flights, we have 12 yes-or-no decisions:

Should sequence j be assigned to a crew? $(j = 1, 2, \ldots, 12)$

Therefore, we use 12 binary variables to represent these respective decisions:

$$x_j = \begin{cases} 1 & \text{if sequence } j \text{ is assigned to a crew} \\ 0 & \text{otherwise.} \end{cases}$$

The most interesting part of this formulation is the nature of each constraint that ensures that a corresponding flight is covered. For example, consider the last flight in Table

TABLE 12.4 Data for Example 3 (the Southwestern Airways problem)

	Feasible Sequence of Flights											
Flight	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
1. San Francisco to Los Angeles	1			1			1			1		
2. San Francisco to Denver		1			1			1			1	
3. San Francisco to Seattle			1			1			1			1
4. Los Angeles to Chicago				2			2		3	2		3
5. Los Angeles to San Francisco	2					3				5	5	
6. Chicago to Denver				3	3				4			
7. Chicago to Seattle							3	3		3	3	4
8. Denver to San Francisco		2		4	4				5			
9. Denver to Chicago					2			2			2	
10. Seattle to San Francisco			2				4	4				5
11. Seattle to Los Angeles						2			2	4	4	2
Cost, $1,000's	2	3	4	6	7	5	7	8	9	9	8	9

12.4 [Seattle to Los Angeles (LA)]. Five sequences (namely, sequences 6, 9, 10, 11, and 12) include this flight. Therefore, at least one of these five sequences must be chosen. The resulting constraint is

$$x_6 + x_9 + x_{10} + x_{11} + x_{12} \geq 1.$$

Using similar constraints for the other 10 flights, the complete BIP model is

$$\text{Minimize} \quad Z = 2x_1 + 3x_2 + 4x_3 + 6x_4 + 7x_5 + 5x_6 + 7x_7 + 8x_8 + 9x_9 + 9x_{10} + 8x_{11} + 9x_{12},$$

subject to

$$\begin{aligned}
x_1 + x_4 + x_7 + x_{10} &\geq 1 && \text{(SF to LA)} \\
x_2 + x_5 + x_8 + x_{11} &\geq 1 && \text{(SF to Denver)} \\
x_3 + x_6 + x_9 + x_{12} &\geq 1 && \text{(SF to Seattle)} \\
x_4 + x_7 + x_9 + x_{10} + x_{12} &\geq 1 && \text{(LA to Chicago)} \\
x_1 + x_6 + x_{10} + x_{11} &\geq 1 && \text{(LA to SF)} \\
x_4 + x_5 + x_9 &\geq 1 && \text{(Chicago to Denver)} \\
x_7 + x_8 + x_{10} + x_{11} + x_{12} &\geq 1 && \text{(Chicago to Seattle)} \\
x_2 + x_4 + x_5 + x_9 &\geq 1 && \text{(Denver to SF)} \\
x_5 + x_8 + x_{11} &\geq 1 && \text{(Denver to Chicago)} \\
x_3 + x_7 + x_8 + x_{12} &\geq 1 && \text{(Seattle to SF)} \\
x_6 + x_9 + x_{10} + x_{11} + x_{12} &\geq 1 && \text{(Seattle to LA)} \\
\sum_{j=1}^{12} x_j &= 3 && \text{(assign three crews)}
\end{aligned}$$

and

$$x_j \text{ is binary,} \qquad \text{for } j = 1, 2, \ldots, 12.$$

One optimal solution for this BIP model is

$$\begin{aligned}
x_3 &= 1 && \text{(assign sequence 3 to a crew)} \\
x_4 &= 1 && \text{(assign sequence 4 to a crew)} \\
x_{11} &= 1 && \text{(assign sequence 11 to a crew)}
\end{aligned}$$

and all other $x_j = 0$, for a total cost of \$18,000. (Another optimal solution is $x_1 = 1$, $x_5 = 1$, $x_{12} = 1$, and all other $x_j = 0$.)

This example illustrates a broader class of problems called **set covering problems**.[1] Any set covering problem can be described in general terms as involving a number of potential *activities* (such as flight sequences) and *characteristics* (such as flights). Each activity possesses some but not all of the characteristics. The objective is to determine the least costly combination of activities that collectively possess (cover) each characteristic

[1]Strictly speaking, a set covering problem does not include any *other* functional constraints such as the last functional constraint in the above crew scheduling example. It also is sometimes assumed that every coefficient in the objective function being minimized equals *one,* and then the name *weighted set covering problem* is used when this assumption does not hold.

at least once. Thus, let S_i be the set of all activities that possess characteristic i. At least one member of the set S_i must be included among the chosen activities, so a constraint,

$$\sum_{j \in S_i} x_j \geq 1,$$

is included for each characteristic i.

A related class of problems, called **set partitioning problems,** changes each such constraint to

$$\sum_{j \in S_i} x_j = 1,$$

so now *exactly* one member of each set S_i must be included among the chosen activities. For the crew scheduling example, this means that each flight must be included *exactly* once among the chosen flight sequences, which rules out having extra crews (as passengers) on any flight.

12.5 SOME PERSPECTIVES ON SOLVING INTEGER PROGRAMMING PROBLEMS

It may seem that IP problems should be relatively easy to solve. After all, *linear programming* problems can be solved extremely efficiently, and the only difference is that IP problems have far fewer solutions to be considered. In fact, *pure* IP problems with a bounded feasible region are guaranteed to have just a *finite* number of feasible solutions.

Unfortunately, there are two fallacies in this line of reasoning. One is that having a finite number of feasible solutions ensures that the problem is readily solvable. Finite numbers can be astronomically large. For example, consider the simple case of BIP problems. With n variables, there are 2^n solutions to be considered (where some of these solutions can subsequently be discarded because they violate the functional constraints). Thus, each time n is increased by 1, the number of solutions is *doubled.* This pattern is referred to as the **exponential growth** of the difficulty of the problem. With $n = 10$, there are more than 1,000 solutions (1,024); with $n = 20$, there are more than 1,000,000; with $n = 30$, there are more than 1 billion; and so forth. Therefore, even the fastest computers are incapable of performing exhaustive enumeration (checking each solution for feasibility and, if it is feasible, calculating the value of the objective value) for BIP problems with more than a few dozen variables, let alone for *general* IP problems with the same number of integer variables. Sophisticated algorithms, such as those described in subsequent sections, can do somewhat better. In fact, Sec. 12.8 discusses how some algorithms have successfully solved certain *vastly* larger BIP problems. The best algorithms today are capable of solving *many* pure BIP problems with a few hundred variables and *some* considerably larger ones (including certain problems with several tens of thousands of variables). Nevertheless, because of *exponential growth,* even the best algorithms cannot be guaranteed to solve every relatively small problem (less than a hundred binary or integer variables). Depending on their characteristics, certain relatively small problems can be much more difficult to solve than some much larger ones.

The second fallacy is that removing some feasible solutions (the noninteger ones) from a linear programming problem will make it easier to solve. To the contrary, it is only because all these feasible solutions are there that the guarantee can be given (see Sec. 5.1)

that there will be a corner-point feasible (CPF) solution [and so a corresponding basic feasible (BF) solution] that is optimal for the overall problem. *This* guarantee is the key to the remarkable efficiency of the simplex method. As a result, linear programming problems generally are *much* easier to solve than IP problems.

Consequently, most successful algorithms for integer programming incorporate the simplex method (or dual simplex method) as much as they can by relating portions of the IP problem under consideration to the corresponding linear programming problem (i.e., the same problem except that the integer restriction is deleted). For any given IP problem, this corresponding linear programming problem commonly is referred to as its **LP relaxation.** The algorithms presented in the next two sections illustrate how a sequence of LP relaxations for portions of an IP problem can be used to solve the overall IP problem effectively.

There is one special situation where solving an IP problem is no more difficult than solving its LP relaxation once by the simplex method, namely, when the optimal solution to the latter problem turns out to satisfy the integer restriction of the IP problem. When this situation occurs, this solution *must* be optimal for the IP problem as well, because it is the best solution among all the feasible solutions for the LP relaxation, which includes all the feasible solutions for the IP problem. Therefore, it is common for an IP algorithm to begin by applying the simplex method to the LP relaxation to check whether this fortuitous outcome has occurred.

Although it generally is quite fortuitous indeed for the optimal solution to the LP relaxation to be integer as well, there actually exist several *special types* of IP problems for which this outcome is *guaranteed.* You already have seen the most prominent of these special types in Chaps. 8 and 9, namely, the *minimum cost flow problem* (with integer parameters) and its special cases (including the *transportation problem,* the *assignment problem,* the *shortest-path problem,* and the *maximum flow problem*). This guarantee can be given for these types of problems because they possess a certain *special structure* (e.g., see Table 8.6) that ensures that every BF solution is integer, as stated in the integer solutions property given in Secs. 8.1 and 9.6. Consequently, these special types of IP problems can be treated as linear programming problems, because they can be solved completely by a streamlined version of the simplex method.

Although this much simplification is somewhat unusual, in practice IP problems frequently have *some* special structure that can be exploited to simplify the problem. (Examples 2 and 3 in the preceding section fit into this category, because of their *mutually exclusive alternatives* constraints or *contingent decisions* constraints or *set-covering* constraints.) Sometimes, very large versions of these problems can be solved successfully. Special-purpose algorithms designed specifically to exploit certain kinds of special structures are becoming increasingly important in integer programming.

Thus, the two primary determinants of *computational difficulty* for an IP problem are (1) the *number of integer variables* and (2) any *special structure* in the problem. This situation is in contrast to linear programming, where the number of (functional) constraints is much more important than the number of variables. In integer programming, the number of constraints is of *some* importance (especially if LP relaxations are being solved), but it is strictly secondary to the other two factors. In fact, there occasionally are cases where *increasing* the number of constraints *decreases* the computation time because the number of feasible solutions has been reduced. For MIP problems, it is the number of *in-*

teger variables rather than the *total* number of variables that is important, because the continuous variables have almost no effect on the computational effort.

Because IP problems are, in general, much more difficult to solve than linear programming problems, sometimes it is tempting to use the approximate procedure of simply applying the simplex method to the LP relaxation and then *rounding* the noninteger values to integers in the resulting solution. This approach may be adequate for some applications, especially if the values of the variables are quite large so that rounding creates relatively little error. However, you should beware of two pitfalls involved in this approach.

One pitfall is that an optimal linear programming solution is *not necessarily feasible* after it is rounded. Often it is difficult to see in which way the rounding should be done to retain feasibility. It may even be necessary to change the value of some variables by one or more units after rounding. To illustrate, consider the following problem:

Maximize $\quad Z = x_2,$

subject to

$$-x_1 + x_2 \leq \frac{1}{2}$$

$$x_1 + x_2 \leq 3\frac{1}{2}$$

and

$$x_1 \geq 0, \qquad x_2 \geq 0$$
x_1, x_2 are integers.

As Fig. 12.2 shows, the optimal solution for the LP relaxation is $x_1 = 1\frac{1}{2}$, $x_2 = 2$, but it is impossible to round the noninteger variable x_1 to 1 or 2 (or any other integer) and retain feasibility. Feasibility can be retained only by also changing the integer value of x_2. It is easy to imagine how such difficulties can be compounded when there are tens or hundreds of constraints and variables.

Even if an optimal solution for the LP relaxation is rounded successfully, there remains another pitfall. There is no guarantee that this rounded solution will be the optimal integer solution. In fact, it may even be far from optimal in terms of the value of the objective function. This fact is illustrated by the following problem:

Maximize $\quad Z = x_1 + 5x_2,$

subject to

$$\begin{aligned} x_1 + 10x_2 &\leq 20 \\ x_1 \qquad\quad &\leq \ 2 \end{aligned}$$

and

$$x_1 \geq 0, \qquad x_2 \geq 0$$
x_1, x_2 are integers.

Because there are only two decision variables, this problem can be depicted graphically as shown in Fig. 12.3. Either the graph or the simplex method may be used to find that

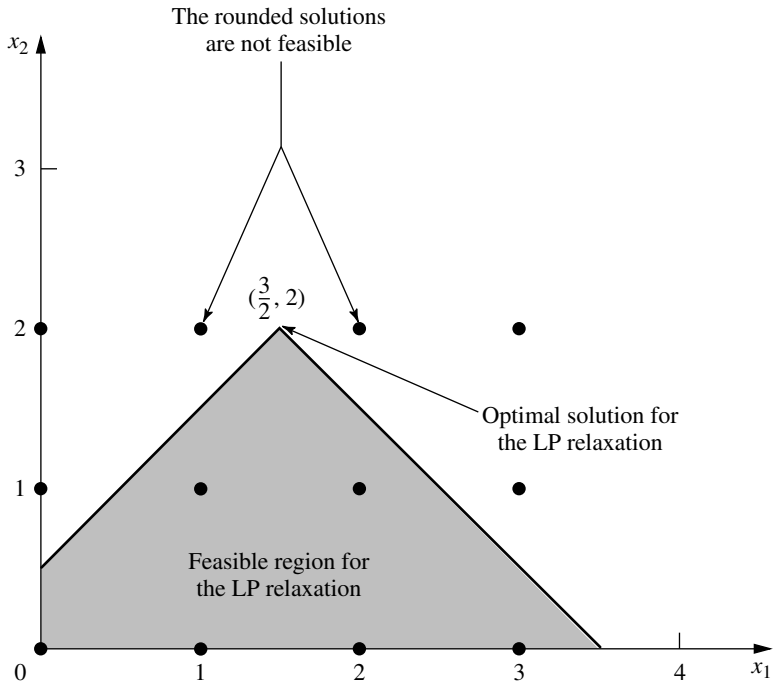


FIGURE 12.2
An example of an IP problem where the optimal solution for the LP relaxation cannot be rounded in any way that retains feasibility.

the optimal solution for the LP relaxation is $x_1 = 2$, $x_2 = \frac{9}{5}$, with $Z = 11$. If a graphical solution were not available (which would be the case with more decision variables), then the variable with the noninteger value $x_2 = \frac{9}{5}$ would normally be rounded in the feasible direction to $x_2 = 1$. The resulting integer solution is $x_1 = 2, x_2 = 1$, which yields $Z = 7$. Notice that this solution is far from the optimal solution $(x_1, x_2) = (0, 2)$, where $Z = 10$.

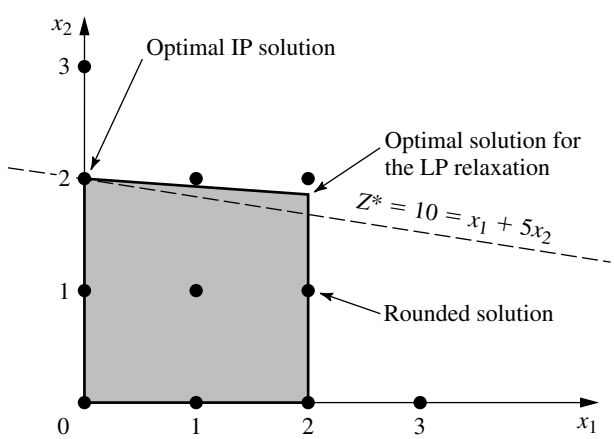


FIGURE 12.3
An example where rounding the optimal solution for the LP relaxation is far from optimal for the IP problem.

Because of these two pitfalls, a better approach for dealing with IP problems that are too large to be solved exactly is to use one of the available *heuristic algorithms.* These algorithms are extremely efficient for large problems, but they are not guaranteed to find an optimal solution. However, they do tend to be considerably more effective than the rounding approach just discussed in finding very good feasible solutions.

One of the particularly exciting developments in OR in recent years has been the rapid progress in developing very effective heuristic algorithms (commonly called *metaheuristics*) for various combinatorial problems such as IP problems. Three prominent types of metaheuristics are tabu search, simulated annealing, and genetic algorithms. All three use innovative concepts that guide a search procedure to move toward an optimal solution. *Tabu search* explores promising areas to hold good solutions by rapidly eliminating unpromising areas that are classified as tabu. *Simulated annealing* conducts the search by using the analog of a physical annealing process. The basic concept underlying the search with *genetic algorithms* is survival of the fittest through natural evolution. These sophisticated metaheuristics (described further in Selected Reference 8) can even be applied to integer *nonlinear* programming problems that have locally optimal solutions that may be far removed from a globally optimal solution.

Returning to integer *linear* programming, for IP problems that are small enough to be solved to optimality, a considerable number of algorithms now are available. However, no IP algorithm possesses computational efficiency that is even nearly comparable to the *simplex method* (except on special types of problems). Therefore, developing IP algorithms has continued to be an active area of research. Fortunately, some exciting algorithmic advances have been made within the last two decades, and additional progress can be anticipated during the coming years. These advances are discussed further in Sec. 12.8.

The most popular mode for IP algorithms is to use the *branch-and-bound technique* and related ideas to *implicitly enumerate* the feasible integer solutions, and we shall focus on this approach. The next section presents the branch-and-bound technique in a general context, and illustrates it with a basic branch-and-bound algorithm for BIP problems. Section 12.7 presents another algorithm of the same type for general MIP problems.

12.6 THE BRANCH-AND-BOUND TECHNIQUE AND ITS APPLICATION TO BINARY INTEGER PROGRAMMING

Because any bounded *pure* IP problem has only a finite number of feasible solutions, it is natural to consider using some kind of *enumeration procedure* for finding an optimal solution. Unfortunately, as we discussed in the preceding section, this finite number can be, and usually is, very large. Therefore, it is imperative that any enumeration procedure be cleverly structured so that only a tiny fraction of the feasible solutions actually need be examined. For example, dynamic programming (see Chap. 11) provides one such kind of procedure for many problems having a finite number of feasible solutions (although it is not particularly efficient for most IP problems). Another such approach is provided by the *branch-and-bound technique.* This technique and variations of it have been applied with some success to a variety of OR problems, but it is especially well known for its application to IP problems.

The basic concept underlying the branch-and-bound technique is to *divide and conquer.* Since the original "large" problem is too difficult to be solved directly, it is divided

into smaller and smaller subproblems until these subproblems can be conquered. The dividing (*branching*) is done by partitioning the entire set of feasible solutions into smaller and smaller subsets. The conquering (*fathoming*) is done partially by *bounding* how good the best solution in the subset can be and then discarding the subset if its bound indicates that it cannot possibly contain an optimal solution for the original problem.

We shall now describe in turn these three basic steps—branching, bounding, and fathoming—and illustrate them by applying a branch-and-bound algorithm to the prototype example (the California Manufacturing Co. problem) presented in Sec. 12.1 and repeated here (with the constraints numbered for later reference).

$$\text{Maximize} \quad Z = 9x_1 + 5x_2 + 6x_3 + 4x_4,$$

subject to

$$\begin{aligned}
&(1) \quad 6x_1 + 3x_2 + 5x_3 + 2x_4 \le 10 \\
&(2) \quad x_3 + x_4 \le 1 \\
&(3) \quad -x_1 + x_3 \le 0 \\
&(4) \quad -x_2 + x_4 \le 0
\end{aligned}$$

and

$$(5) \quad x_j \text{ is binary}, \quad \text{for } j = 1, 2, 3, 4.$$

Branching

When you are dealing with binary variables, the most straightforward way to partition the set of feasible solutions into subsets is to fix the value of one of the variables (say, x_1) at $x_1 = 0$ for one subset and at $x_1 = 1$ for the other subset. Doing this for the prototype example divides the whole problem into the two smaller subproblems shown below.

Subproblem 1:
Fix $x_1 = 0$ so the resulting subproblem is

$$\text{Maximize} \quad Z = 5x_2 + 6x_3 + 4x_4,$$

subject to

$$\begin{aligned}
&(1) \quad 3x_2 + 5x_3 + 2x_4 \le 10 \\
&(2) \quad x_3 + x_4 \le 1 \\
&(3) \quad x_3 \le 0 \\
&(4) \quad -x_2 + x_4 \le 0 \\
&(5) \quad x_j \text{ is binary}, \quad \text{for } j = 2, 3, 4.
\end{aligned}$$

Subproblem 2:
Fix $x_1 = 1$ so the resulting subproblem is

$$\text{Maximize} \quad Z = 9 + 5x_2 + 6x_3 + 4x_4,$$

subject to

$$\begin{aligned}
&(1) \quad 3x_2 + 5x_3 + 2x_4 \le 4 \\
&(2) \quad x_3 + x_4 \le 1 \\
&(3) \quad x_3 \le 1
\end{aligned}$$

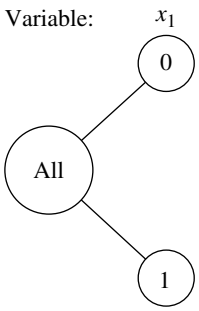


FIGURE 12.4
The solution tree created by the branching for the first iteration of the BIP branch-and-bound algorithm for the example in Sec. 12.1.

(4) $-x_2 + x_4 \leq 0$
(5) x_j is binary, for $j = 2, 3, 4$.

Figure 12.4 portrays this dividing (branching) into subproblems by a *tree* (defined in Sec. 9.2) with *branches* (arcs) from the *All* node (corresponding to the whole problem having *all* feasible solutions) to the two nodes corresponding to the two subproblems. This tree, which will continue "growing branches" iteration by iteration, is referred to as the **solution tree** (or **enumeration tree**) for the algorithm. The variable used to do this branching at any iteration by assigning values to the variable (as with x_1 above) is called the **branching variable.** (Sophisticated methods for selecting branching variables are an important part of some branch-and-bound algorithms but, for simplicity, we always select them in their natural order—$x_1, x_2, \ldots, x_n$—throughout this section.)

Later in the section you will see that one of these subproblems can be conquered (fathomed) immediately, whereas the other subproblem will need to be divided further into smaller subproblems by setting $x_2 = 0$ or $x_2 = 1$.

For other IP problems where the integer variables have more than two possible values, the branching can still be done by setting the branching variable at its respective individual values, thereby creating more than two new subproblems. However, a good alternate approach is to specify a *range* of values (for example, $x_j \leq 2$ or $x_j \geq 3$) for the branching variable for each new subproblem. This is the approach used for the algorithm presented in Sec. 12.7.

Bounding

For each of these subproblems, we now need to obtain a *bound* on how good its best feasible solution can be. The standard way of doing this is to quickly solve a simpler *relaxation* of the subproblem. In most cases, a **relaxation** of a problem is obtained simply by *deleting* ("relaxing") one set of constraints that had made the problem difficult to solve. For IP problems, the most troublesome constraints are those requiring the respective variables to be integer. Therefore, the most widely used relaxation is the *LP relaxation* that deletes this set of constraints.

To illustrate for the example, consider first the whole problem given in Sec. 12.1. Its LP relaxation is obtained by replacing the last line of the model (x_j is binary, for $j = 1, 2, 3, 4$) by the constraints that $x_j \leq 1$ and $x_j \geq 0$ for $j = 1, 2, 3, 4$. Using the simplex method to quickly solve this LP relaxation yields its optimal solution

$$(x_1, x_2, x_3, x_4) = \left(\frac{5}{6}, 1, 0, 1\right), \qquad \text{with } Z = 16\frac{1}{2}.$$

Therefore, $Z \leq 16\frac{1}{2}$ for all feasible solutions for the original BIP problem (since these solutions are a subset of the feasible solutions for the LP relaxation). In fact, as summarized below, this *bound* of $16\frac{1}{2}$ can be rounded down to 16, because all coefficients in the objective function are integer, so all integer solutions must have an integer value for Z.

Bound for whole problem: $Z \leq 16$.

Now let us obtain the bounds for the two subproblems in the same way. Their LP relaxations are obtained from the models in the preceding subsection by replacing the constraints that x_j is binary for $j = 2, 3, 4$ by the constraints $0 \leq x_j \leq 1$ for $j = 2, 3, 4$. Ap-

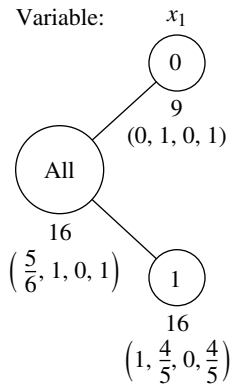


FIGURE 12.5
The results of bounding for the first iteration of the BIP branch-and-bound algorithm for the example in Sec. 12.1.

plying the simplex method then yields their optimal solutions (plus the fixed value of x_1) shown below.

LP relaxation of subproblem 1: $(x_1, x_2, x_3, x_4) = (0, 1, 0, 1)$ with $Z = 9$.

LP relaxation of subproblem 2: $(x_1, x_2, x_3, x_4) = \left(1, \frac{4}{5}, 0, \frac{4}{5}\right)$ with $Z = 16\frac{1}{5}$.

The resulting bounds for the subproblems then are

Bound for subproblem 1: $Z \leq 9$,
Bound for subproblem 2: $Z \leq 16$.

Figure 12.5 summarizes these results, where the numbers given just below the nodes are the bounds and below each bound is the optimal solution obtained for the LP relaxation.

Fathoming

A subproblem can be conquered (fathomed), and thereby dismissed from further consideration, in the three ways described below.

One way is illustrated by the results for subproblem 1 given by the $x_1 = 0$ node in Fig. 12.5. Note that the (unique) optimal solution for its LP relaxation, $(x_1, x_2, x_3, x_4) = (0, 1, 0, 1)$, is an *integer* solution. Therefore, this solution must also be the optimal solution for subproblem 1 itself. This solution should be stored as the first **incumbent** (the best feasible solution found so far) for the whole problem, along with its value of Z. This value is denoted by

Z^* = value of Z for current incumbent,

so $Z^* = 9$ at this point. Since this solution has been stored, there is no reason to consider subproblem 1 any further by branching from the $x_1 = 0$ node, etc. Doing so could only lead to other feasible solutions that are inferior to the incumbent, and we have no interest in such solutions. Because it has been solved, we **fathom** (dismiss) subproblem 1 now.

The above results suggest a second key fathoming test. Since $Z^* = 9$, there is no reason to consider further any subproblem whose *bound* ≤ 9, since such a subproblem cannot have a feasible solution better than the *incumbent.* Stated more generally, a subproblem is fathomed whenever its

Bound $\leq Z^*$.

This outcome does not occur in the current iteration of the example because subproblem 2 has a bound of 16 that is larger than 9. However, it might occur later for **descendants** of this subproblem (new smaller subproblems created by branching on this subproblem, and then perhaps branching further through subsequent "generations"). Furthermore, as new incumbents with larger values of Z^* are found, it will become easier to *fathom* in this way.

The third way of fathoming is quite straightforward. If the simplex method finds that a subproblem's LP relaxation has *no feasible solutions,* then the subproblem itself must have *no feasible solutions,* so it can be dismissed (fathomed).

In all three cases, we are conducting our search for an optimal solution by retaining for further investigation only those subproblems that could possibly have a feasible solution better than the current incumbent.

Summary of Fathoming Tests. A subproblem is *fathomed* (dismissed from further consideration) if

Test 1: Its bound $\leq Z^*$,

or

Test 2: Its LP relaxation has no feasible solutions,

or

Test 3: The optimal solution for its LP relaxation is *integer.* (If this solution is better than the incumbent, it becomes the new incumbent, and test 1 is reapplied to all unfathomed subproblems with the new larger Z^*.)

Figure 12.6 summarizes the results of applying these three tests to subproblems 1 and 2 by showing the current *solution tree.* Only subproblem 1 has been fathomed, by test 3, as indicated by $F(3)$ next to the $x_1 = 0$ node. The resulting incumbent also is identified below this node.

The subsequent iterations will illustrate successful applications of all three tests. However, before continuing the example, we summarize the algorithm being applied to this BIP problem. (This algorithm assumes that all coefficients in the objective function are integer and that the ordering of the variables for branching is $x_1, x_2, \ldots, x_n$.)

Summary of the BIP Branch-and-Bound Algorithm.

Initialization: Set $Z^* = -\infty$. Apply the bounding step, fathoming step, and optimality test described below to the whole problem. If not fathomed, classify this problem as the one remaining "subproblem" for performing the first full iteration below.

Steps for each iteration:

1. *Branching:* Among the *remaining* (unfathomed) subproblems, select the one that was created *most recently.* (Break ties according to which has the *larger bound.*) Branch from the node for this subproblem to create two new subproblems by fixing the next variable (the branching variable) at either 0 or 1.
2. *Bounding:* For each new subproblem, obtain its *bound* by applying the simplex method to its LP relaxation and rounding down the value of Z for the resulting optimal solution.
3. *Fathoming:* For each new subproblem, apply the three fathoming tests summarized above, and discard those subproblems that are fathomed by any of the tests.

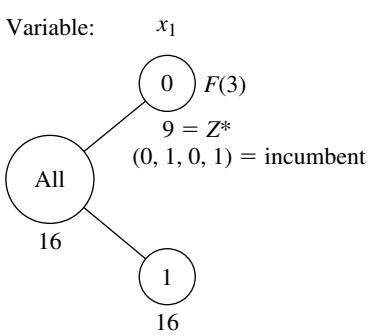


FIGURE 12.6
The solution tree after the first iteration of the BIP branch-and-bound algorithm for the example in Sec. 12.1.

Optimality test: Stop when there are *no remaining* subproblems; the current *incumbent* is optimal.[1] Otherwise, return to perform another iteration.

The branching step for this algorithm warrants a comment as to why the subproblem to branch from is selected in this way. One option not used would have been always to select the remaining subproblem with the *best bound,* because this subproblem would be the most promising one to contain an optimal solution for the whole problem. The reason for instead selecting the *most recently created* subproblem is that *LP relaxations* are being solved in the bounding step. Rather than start the simplex method from scratch each time, each LP relaxation generally is solved by *reoptimization* in large-scale implementations of this algorithm. This reoptimization involves revising the final simplex tableau from the preceding LP relaxation as needed because of the few differences in the model (just as for sensitivity analysis) and then applying a few iterations of perhaps the dual simplex method. This reoptimization tends to be *much* faster than starting from scratch, *provided* the preceding and current models are closely related. The models will tend to be closely related under the branching rule used, but *not* when you are skipping around in the solution tree by selecting the subproblem with the best bound.

Completing the Example

The pattern for the remaining iterations will be quite similar to that for the first iteration described above except for the ways in which fathoming occurs. Therefore, we shall summarize the branching and bounding steps fairly briefly and then focus on the fathoming step.

Iteration 2. The only remaining subproblem corresponds to the $x_1 = 1$ node in Fig. 12.6, so we shall branch from this node to create the two new subproblems given below.

Subproblem 3:
Fix $x_1 = 1$, $x_2 = 0$ so the resulting subproblem is

Maximize $Z = 9 + 6x_3 + 4x_4$,

subject to

$$\begin{aligned}
&(1) \quad 5x_3 + 2x_4 \le 4 \\
&(2) \quad x_3 + x_4 \le 1 \\
&(3) \quad x_3 \le 1 \\
&(4) \quad x_4 \le 0 \\
&(5) \quad x_j \text{ is binary,} \quad \text{for } j = 3, 4.
\end{aligned}$$

Subproblem 4:
Fix $x_1 = 1$, $x_2 = 1$ so the resulting subproblem is

Maximize $Z = 14 + 6x_3 + 4x_4$,

subject to

$$\begin{aligned}
&(1) \quad 5x_3 + 2x_4 \le 1 \\
&(2) \quad x_3 + x_4 \le 1
\end{aligned}$$

[1]If there is no incumbent, the conclusion is that the problem has no feasible solutions.

(3) $x_3 \leq 1$
(4) $x_4 \leq 1$
(5) x_j is binary, for $j = 3, 4$.

The LP relaxations of these subproblems are obtained by replacing the constraints x_j is binary for $j = 3, 4$ by the constraints $0 \leq x_j \leq 1$ for $j = 3, 4$. Their optimal solutions (plus the fixed values of x_1 and x_2) are

LP relaxation of subproblem 3: $(x_1, x_2, x_3, x_4) = \left(1, 0, \frac{4}{5}, 0\right)$ with $Z = 13\frac{4}{5}$,

LP relaxation of subproblem 4: $(x_1, x_2, x_3, x_4) = \left(1, 1, 0, \frac{1}{2}\right)$ with $Z = 16$.

The resulting bounds for the subproblems are

Bound for subproblem 3: $Z \leq 13$,
Bound for subproblem 4: $Z \leq 16$.

Note that both these bounds are larger than $Z^* = 9$, so fathoming test 1 fails in both cases. Test 2 also fails, since both LP relaxations have feasible solutions (as indicated by the existence of an optimal solution). Alas, test 3 fails as well, because both optimal solutions include variables with noninteger values.

Figure 12.7 shows the resulting solution tree at this point. The lack of an *F* to the right of either new node indicates that both remain unfathomed.

Iteration 3. So far, the algorithm has created four subproblems. Subproblem 1 has been fathomed, and subproblem 2 has been replaced by (separated into) subproblems 3 and 4, but these last two remain under consideration. Because they were created simultaneously, but subproblem 4 ($x_1 = 1$, $x_2 = 1$) has the larger *bound* ($16 > 13$), the next branching is done from the $(x_1, x_2) = (1, 1)$ node in the solution tree, which creates the following new subproblems (where constraint 3 disappears because it does not contain x_4).

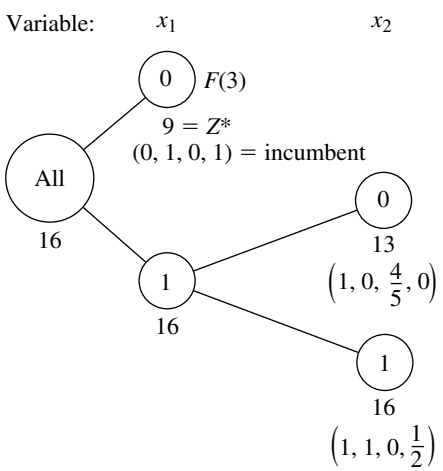


FIGURE 12.7
The solution tree after iteration 2 of the BIP branch-and-bound algorithm for the example in Sec. 12.1.

Subproblem 5:
Fix $x_1 = 1$, $x_2 = 1$, $x_3 = 0$ so the resulting subproblem is

Maximize $Z = 14 + 4x_4$,

subject to

(1) $2x_4 \leq 1$
(2), (4) $x_4 \leq 1$ (twice)
(5) x_4 is binary.

Subproblem 6:
Fix $x_1 = 1$, $x_2 = 1$, $x_3 = 1$ so the resulting subproblem is

Maximize $Z = 20 + 4x_4$,

subject to

(1) $2x_4 \leq -4$
(2) $x_4 \leq 0$
(4) $x_4 \leq 1$
(5) x_4 is binary.

If we form their LP relaxations by replacing constraint 5 by

(5) $0 \leq x_4 \leq 1$,

the following results are obtained:

LP relaxation of subproblem 5: $(x_1, x_2, x_3, x_4) = \left(1, 1, 0, \frac{1}{2}\right)$, with $Z = 16$.
LP relaxation of subproblem 6: No feasible solutions.
Bound for subproblem 5: $Z \leq 16$.

Note how the combination of constraints 1 and 5 in the LP relaxation of subproblem 6 prevents any feasible solutions. Therefore, this subproblem is fathomed by test 2. However, subproblem 5 fails this test, as well as test 1 ($16 > 9$) and test 3 ($x_4 = \frac{1}{2}$ is not integer), so it remains under consideration.

We now have the solution tree shown in Fig. 12.8.

Iteration 4. The subproblems corresponding to nodes (1, 0) and (1, 1, 0) in Fig. 12.8 remain under consideration, but the latter node was created more recently, so it is selected for branching from next. Since the resulting branching variable x_4 is the *last* variable, fixing its value at either 0 or 1 actually creates a *single solution* rather than subproblems requiring fuller investigation. These single solutions are

$x_4 = 0$: $(x_1, x_2, x_3, x_4) = (1, 1, 0, 0)$ is feasible, with $Z = 14$,
$x_4 = 1$: $(x_1, x_2, x_3, x_4) = (1, 1, 0, 1)$ is infeasible.

Formally applying the fathoming tests, we see that the first solution passes test 3 and the second passes test 2. Furthermore, this feasible first solution is better than the incumbent ($14 > 9$), so it becomes the new incumbent, with $Z^* = 14$.

Because a new incumbent has been found, we now reapply fathoming test 1 with the new larger value of Z^* to the only remaining subproblem, the one at node (1, 0).

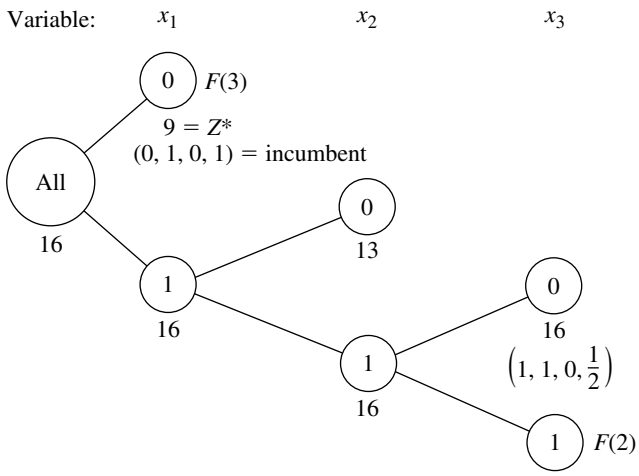


FIGURE 12.8
The solution tree after iteration 3 of the BIP branch-and-bound algorithm for the example in Sec. 12.1.

Subproblem 3:

Bound = 13 ≤ Z^* = 14.

Therefore, this subproblem now is fathomed.

We now have the solution tree shown in Fig. 12.9. Note that there are *no remaining* (unfathomed) subproblems. Consequently, the optimality test indicates that the current incumbent

$$(x_1, x_2, x_3, x_4) = (1, 1, 0, 0)$$

is optimal, so we are done.

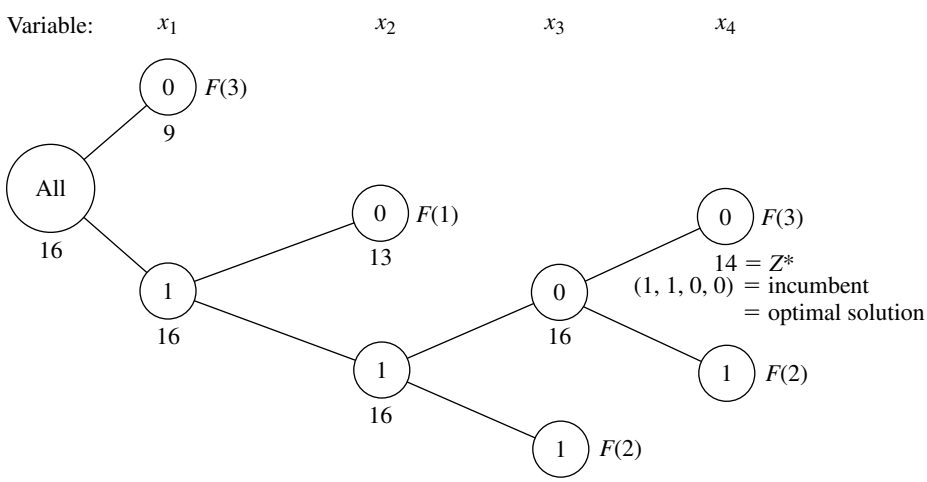


FIGURE 12.9
The solution tree after the final (fourth) iteration of the BIP branch-and-bound algorithm for the example in Sec. 12.1.

Your OR Tutor includes another example of applying this algorithm. Also included in the OR Courseware is an interactive routine for executing this algorithm. As usual, the Excel, LINGO/LINDO, and MPL/CPLEX files for this chapter in your OR Courseware show how the student version of these software packages is applied to the various examples in the chapter. The algorithms they use for BIP problems all are similar to the one described above.[1]

Other Options with the Branch-and-Bound Technique

This section has illustrated the branch-and-bound technique by describing a basic branch-and-bound algorithm for solving BIP problems. However, the general framework of the branch-and-bound technique provides a great deal of flexibility in how to design a specific algorithm for any given type of problem such as BIP. There are many options available, and constructing an efficient algorithm requires tailoring the specific design to fit the specific structure of the problem type.

Every branch-and-bound algorithm has the same three basic steps of *branching, bounding,* and *fathoming.* The flexibility lies in how these steps are performed.

Branching always involves *selecting* one remaining subproblem and *dividing* it into smaller subproblems. The flexibility here is found in the rules for selecting and dividing. Our BIP algorithm selected the *most recently created* subproblem, because this is very efficient for *reoptimizing* each LP relaxation from the preceding one. Selecting the subproblem with the *best bound* is the other most popular rule, because it tends to lead more quickly to better incumbents and so more fathoming. Combinations of the two rules also can be used. The *dividing* typically (but not always) is done by choosing a *branching variable* and assigning it either individual values (e.g., our BIP algorithm) or ranges of values (e.g., the algorithm in the next section). More sophisticated algorithms generally use a rule for strategically choosing a branching variable that should tend to lead to early fathoming.

Bounding usually is done by solving a *relaxation.* However, there are a variety of ways to form relaxations. For example, consider the **Lagrangian relaxation,** where the entire set of functional constraints $\mathbf{Ax} \leq \mathbf{b}$ (in matrix notation) is *deleted* (except possibly for any "convenient" constraints) and then the objective function

$$\text{Maximize} \quad Z = \mathbf{cx},$$

is replaced by

$$\text{Maximize} \quad Z_R = \mathbf{cx} - \boldsymbol{\lambda}(\mathbf{Ax} - \mathbf{b}),$$

where the fixed vector $\boldsymbol{\lambda} \geq \mathbf{0}$. If $\mathbf{x}^*$ is an optimal solution for the original problem, its $Z \leq Z_R$, so solving the Lagrangian relaxation for the optimal value of Z_R provides a valid *bound*. If $\boldsymbol{\lambda}$ is chosen well, this bound tends to be a reasonably tight one (at least comparable to the bound from the LP relaxation). Without any functional constraints, this relaxation also can be solved extremely quickly. The drawbacks are that fathoming tests 2 and 3 (revised) are not as powerful as for the LP relaxation.

[1]In the professional version of LINGO, LINDO, and CPLEX, the BIP algorithm also uses a variety of sophisticated techniques along the lines described in Sec. 12.8.

In general terms, two features are sought in choosing a relaxation: it can be solved relatively quickly, and provides a relatively tight bound. Neither alone is adequate. The LP relaxation is popular because it provides an excellent trade-off between these two factors.

One option occasionally employed is to use a quickly solved relaxation and then, if fathoming is not achieved, to tighten the relaxation in some way to obtain a somewhat tighter bound.

Fathoming generally is done pretty much as described for the BIP algorithm. The three fathoming criteria can be stated in more general terms as follows.

Summary of Fathoming Criteria. A subproblem is *fathomed* if an analysis of its *relaxation* reveals that

Criterion 1: Feasible solutions of the subproblem must have $Z \leq Z^*$, or
Criterion 2: The subproblem has no feasible solutions, or
Criterion 3: An optimal solution of the subproblem has been found.

Just as for the BIP algorithm, the first two criteria usually are applied by solving the relaxation to obtain a bound for the subproblem and then checking whether this bound is $\leq Z^*$ (test 1) or whether the relaxation has no feasible solutions (test 2). If the relaxation differs from the subproblem *only* by the deletion (or loosening) of some constraints, then the third criterion usually is applied by checking whether the optimal solution for the relaxation is *feasible* for the subproblem, in which case it must be *optimal* for the subproblem. For other relaxations (such as the Lagrangian relaxation), additional analysis is required to determine whether the optimal solution for the relaxation is also optimal for the subproblem.

If the original problem involves *minimization* rather than maximization, two options are available. One is to convert to maximization in the usual way (see Sec. 4.6). The other is to convert the branch-and-bound algorithm directly to minimization form, which requires changing the direction of the inequality for fathoming test 1 from

Is the subproblem's bound $\leq Z^*$?

to

Is the subproblem's bound $\geq Z^*$?

So far, we have described how to use the branch-and-bound technique to find only *one* optimal solution. However, in the case of ties for the optimal solution, it is sometimes desirable to identify *all* these optimal solutions so that the final choice among them can be made on the basis of intangible factors not incorporated into the mathematical model. To find them all, you need to make only a few slight alterations in the procedure. First, change the weak inequality for fathoming test 1 (Is the subproblem's bound $\leq Z^*$?) to a strict inequality (Is the subproblem's bound $< Z^*$?), so that fathoming will not occur if the subproblem can have a feasible solution *equal* to the incumbent. Second, if fathoming test 3 passes and the optimal solution for the subproblem has $Z = Z^*$, then store this solution as *another* (tied) incumbent. Third, if test 3 provides a new incumbent (tied or otherwise), then check whether the optimal solution obtained for the *relaxation* is *unique.* If it is not, then identify the other optimal solutions for the relaxation and check whether they are optimal for the subproblem as well, in which case they also become incumbents.

Finally, when the *optimality test* finds that there are *no remaining* (unfathomed) subsets, *all* the current *incumbents* will be the *optimal* solutions.

Finally, note that rather than find an optimal solution, the branch-and-bound technique can be used to find a *nearly optimal* solution, generally with much less computational effort. For some applications, a solution is "good enough" if its Z is "close enough" to the value of Z for an optimal solution (call it Z^{**}). *Close enough* can be defined in either of two ways as either

$$Z^{**} - K \leq Z \qquad \text{or} \qquad (1 - \alpha)Z^{**} \leq Z$$

for a specified (positive) constant K or α. For example, if the second definition is chosen and $\alpha = 0.05$, then the solution is required to be within 5 percent of optimal. Consequently, if it were known that the value of Z for the current incumbent (Z^*) satisfies either

$$Z^{**} - K \leq Z^* \qquad \text{or} \qquad (1 - \alpha)Z^{**} \leq Z^*$$

then the procedure could be terminated immediately by choosing the incumbent as the desired nearly optimal solution. Although the procedure does not actually identify an optimal solution and the corresponding Z^{**}, if this (unknown) solution is feasible (and so optimal) for the subproblem currently under investigation, then fathoming test 1 finds an upper bound such that

$$Z^{**} \leq \text{bound}$$

so that either

$$\text{Bound} - K \leq Z^* \qquad \text{or} \qquad (1 - \alpha)\text{bound} \leq Z^*$$

would imply that the corresponding inequality in the preceding sentence is satisfied. Even if this solution is not feasible for the current subproblem, a valid upper bound is still obtained for the value of Z for the subproblem's optimal solution. Thus, satisfying either of these last two inequalities is sufficient to fathom this subproblem because the incumbent must be "close enough" to the subproblem's optimal solution.

Therefore, to find a solution that is close enough to being optimal, only one change is needed in the usual branch-and-bound procedure. This change is to replace the usual fathoming test 1 for a subproblem

$$\text{Bound} \leq Z^*?$$

by either

$$\text{Bound} - K \leq Z^*?$$

or

$$(1 - \alpha)(\text{bound}) \leq Z^*?$$

and then perform this test *after* test 3 (so that a feasible solution found with $Z > Z^*$ is still kept as the new incumbent). The reason this weaker test 1 suffices is that regardless of how close Z for the subproblem's (unknown) optimal solution is to the subproblem's bound, the incumbent is still close enough to this solution (if the new inequality holds) that the subproblem does not need to be considered further. When there are no remaining subproblems, the current incumbent will be the desired *nearly optimal* solution. However, it is much

easier to fathom with this new fathoming test (in either form), so the algorithm should run much faster. For a large problem, this acceleration may make the difference between finishing with a solution guaranteed to be close to optimal and never terminating.

12.7 A BRANCH-AND-BOUND ALGORITHM FOR MIXED INTEGER PROGRAMMING

We shall now consider the general MIP problem, where *some* of the variables (say, I of them) are restricted to integer values (but not necessarily just 0 and 1) but the rest are ordinary continuous variables. For notational convenience, we shall order the variables so that the first I variables are the *integer-restricted* variables. Therefore, the general form of the problem being considered is

$$\text{Maximize} \qquad Z = \sum_{j=1}^{n} c_j x_j,$$

subject to

$$\sum_{j=1}^{n} a_{ij} x_j \le b_i, \qquad \text{for } i = 1, 2, \ldots, m,$$

and

$$x_j \ge 0, \qquad \text{for } j = 1, 2, \ldots, n,$$
$$x_j \text{ is integer}, \qquad \text{for } j = 1, 2, \ldots, I; \; I \le n.$$

(When $I = n$, this problem becomes the pure IP problem.)

We shall describe a basic branch-and-bound algorithm for solving this problem that, with a variety of refinements, has provided a standard approach to MIP. The structure of this algorithm was first developed by R. J. Dakin,[1] based on a pioneering branch-and-bound algorithm by A. H. Land and A. G. Doig.[2]

This algorithm is quite similar in structure to the BIP algorithm presented in the preceding section. Solving *LP relaxations* again provides the basis for both the *bounding* and *fathoming* steps. In fact, only four changes are needed in the BIP algorithm to deal with the generalizations from *binary* to *general* integer variables and from *pure* IP to *mixed* IP.

One change involves the choice of the *branching variable.* Before, the *next* variable in the natural ordering—$x_1, x_2, \ldots, x_n$—was chosen automatically. Now, the only variables considered are the *integer-restricted* variables that have a *noninteger* value in the optimal solution for the LP relaxation of the current subproblem. Our rule for choosing among these variables is to select the *first* one in the natural ordering. (Production codes generally use a more sophisticated rule.)

[1]R. J. Dakin, "A Tree Search Algorithm for Mixed Integer Programming Problems," *Computer Journal,* **8**(3): 250–255, 1965.

[2]A. H. Land and A. G. Doig, "An Automatic Method of Solving Discrete Programming Problems," *Econometrica,* **28:** 497–520, 1960.

The second change involves the values assigned to the branching variable for creating the new smaller subproblems. Before, the *binary* variable was fixed at 0 and 1, respectively, for the two new subproblems. Now, the *general* integer-restricted variable could have a very large number of possible integer values, and it would be inefficient to create *and* analyze *many* subproblems by fixing the variable at its individual integer values. Therefore, what is done instead is to create just *two* new subproblems (as before) by specifying two *ranges* of values for the variable.

To spell out how this is done, let x_j be the current branching variable, and let x_j^* be its (noninteger) value in the optimal solution for the LP relaxation of the current subproblem. Using square brackets to denote

$$[x_j^*] = \text{greatest integer} \leq x_j^*,$$

we have for the range of values for the two new subproblems

$$x_j \leq [x_j^*] \qquad \text{and} \qquad x_j \geq [x_j^*] + 1,$$

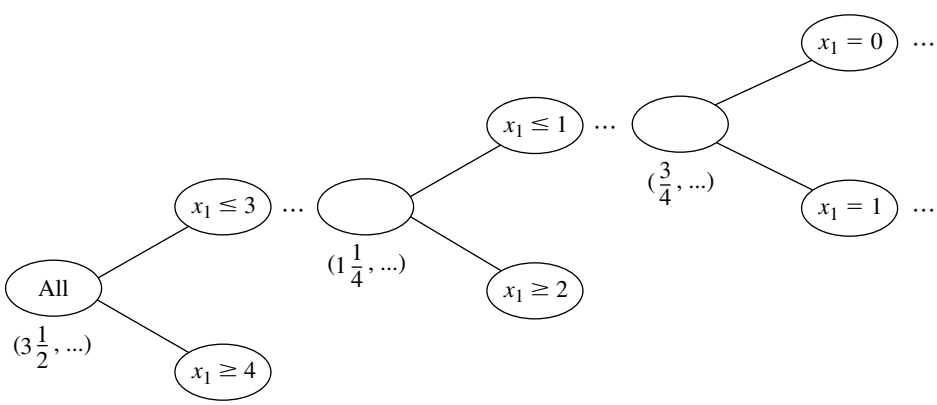
respectively. Each inequality becomes an *additional constraint* for that new subproblem. For example, if $x_j^* = 3\frac{1}{2}$, then

$$x_j \leq 3 \qquad \text{and} \qquad x_j \geq 4$$

are the respective additional constraints for the new subproblem.

When the two changes to the BIP algorithm described above are combined, an interesting phenomenon of a *recurring branching variable* can occur. To illustrate, as shown in Fig. 12.10, let $j = 1$ in the above example where $x_j^* = 3\frac{1}{2}$, and consider the new subproblem where $x_1 \leq 3$. When the LP relaxation of a descendant of this subproblem is solved, suppose that $x_1^* = 1\frac{1}{4}$. Then x_1 *recurs* as the branching variable, and the two new subproblems created have the additional constraint $x_1 \leq 1$ and $x_1 \geq 2$, respectively (as well as the previous additional constraint $x_1 \leq 3$). Later, when the LP relaxation for a descendant of, say, the $x_1 \leq 1$ subproblem is solved, suppose that $x_1^* = \frac{3}{4}$. Then x_1 *recurs* again as the branching variable, and the two new subproblems created have $x_1 = 0$ (be-

FIGURE 12.10
Illustration of the phenomenon of a *recurring branching variable,* where here x_1 becomes a branching variable three times because it has a noninteger value in the optimal solution for the LP relaxation at three nodes.



cause of the new $x_1 \leq 0$ constraint and the nonnegativity constraint on x_1) and $x_1 = 1$ (because of the new $x_1 \geq 1$ constraint and the previous $x_1 \leq 1$ constraint).

The third change involves the *bounding step.* Before, with a *pure* IP problem and integer coefficients in the objective function, the value of Z for the optimal solution for the subproblem's LP relaxation was *rounded down* to obtain the bound, because any feasible solution for the subproblem must have an *integer* Z. Now, with some of the variables *not* integer-restricted, the bound is the value of Z *without* rounding down.

The fourth (and final) change to the BIP algorithm to obtain our MIP algorithm involves fathoming test 3. Before, with a *pure* IP problem, the test was that the optimal solution for the subproblem's LP relaxation is *integer,* since this ensures that the solution is feasible, and therefore optimal, for the subproblem. Now, with a *mixed* IP problem, the test requires only that the *integer-restricted* variables be *integer* in the optimal solution for the subproblem's LP relaxation, because this suffices to ensure that the solution is feasible, and therefore optimal, for the subproblem.

Incorporating these four changes into the summary presented in the preceding section for the BIP algorithm yields the following summary for the new algorithm for MIP.

Summary of the MIP Branch-and-Bound Algorithm.

Initialization: Set $Z^* = -\infty$. Apply the bounding step, fathoming step, and optimality test described below to the whole problem. If not fathomed, classify this problem as the one remaining subproblem for performing the first full iteration below.

Steps for each iteration:

1. *Branching:* Among the *remaining* (unfathomed) subproblems, select the one that was created *most recently.* (Break ties according to which has the *larger bound.*) Among the *integer-restricted* variables that have a *noninteger* value in the optimal solution for the LP relaxation of the subproblem, choose the *first one* in the natural ordering of the variables to be the *branching variable.* Let x_j be this variable and x_j^* its value in this solution. Branch from the node for the subproblem to create two new subproblems by adding the respective constraints $x_j \leq [x_j^*]$ and $x_j \geq [x_j^*] + 1$.
2. *Bounding:* For each new subproblem, obtain its bound by applying the simplex method (or the dual simplex method when reoptimizing) to its LP relaxation and using the value of Z for the resulting optimal solution.
3. *Fathoming:* For each new subproblem, apply the three fathoming tests given below, and discard those subproblems that are fathomed by any of the tests.
   Test 1: Its bound $\leq Z^*$, where Z^* is the value of Z for the current *incumbent.*
   Test 2: Its LP relaxation has no feasible solutions.
   Test 3: The optimal solution for its LP relaxation has *integer* values for the *integer-restricted* variables. (If this solution is better than the incumbent, it becomes the new incumbent and test 1 is reapplied to all unfathomed subproblems with the new larger Z^*.)

Optimality test: Stop when there are no remaining subproblems; the current *incumbent* is optimal.[1] Otherwise, perform another iteration.

[1]If there is no incumbent, the conclusion is that the problem has no feasible solutions.

An MIP Example. We will now illustrate this algorithm by applying it to the following MIP problem:

Maximize $Z = 4x_1 - 2x_2 + 7x_3 - x_4$,

subject to

$$\begin{aligned} x_1 \qquad + 5x_3 \qquad &\le 10 \\ x_1 + x_2 - x_3 \qquad &\le 1 \\ 6x_1 - 5x_2 \qquad &\le 0 \\ -x_1 \qquad + 2x_3 - 2x_4 &\le 3 \end{aligned}$$

and

$x_j \ge 0$, for $j = 1, 2, 3, 4$
x_j is an integer, for $j = 1, 2, 3$.

Note that the number of integer-restricted variables is $I = 3$, so x_4 is the only continuous variable.

Initialization. After setting $Z^* = -\infty$, we form the LP relaxation of this problem by *deleting* the set of constraints that x_j is an integer for $j = 1, 2, 3$. Applying the simplex method to this LP relaxation yields its optimal solution below.

LP relaxation of whole problem: $(x_1, x_2, x_3, x_4) = \left(\frac{5}{4}, \frac{3}{2}, \frac{7}{4}, 0\right)$, with $Z = 14\frac{1}{4}$.

Because it has *feasible* solutions and this optimal solution has *noninteger* values for its integer-restricted variables, the whole problem is not fathomed, so the algorithm continues with the first full iteration below.

Iteration 1. In this optimal solution for the LP relaxation, the *first* integer-restricted variable that has a noninteger value is $x_1 = \frac{5}{4}$, so x_1 becomes the branching variable. Branching from the *All* node (*all* feasible solutions) with this branching variable then creates the following two subproblems:

Subproblem 1:
Original problem plus additional constraint

$x_1 \le 1$.

Subproblem 2:
Original problem plus additional constraint

$x_1 \ge 2$.

Deleting the set of integer constraints again and solving the resulting LP relaxations of these two subproblems yield the following results.

LP relaxation of subproblem 1: $(x_1, x_2, x_3, x_4) = \left(1, \frac{6}{5}, \frac{9}{5}, 0\right)$, with $Z = 14\frac{1}{5}$.

Bound for subproblem 1: $Z \le 14\frac{1}{5}$.

LP relaxation of subproblem 2: No feasible solutions.

This outcome for subproblem 2 means that it is fathomed by test 2. However, just as for the whole problem, subproblem 1 fails all fathoming tests.

These results are summarized in the solution tree shown in Fig. 12.11.

Iteration 2. With only one remaining subproblem, corresponding to the $x_1 \leq 1$ node in Fig. 12.11, the next branching is from this node. Examining its LP relaxation's optimal solution given below, we see that this node reveals that the *branching variable* is x_2, because $x_2 = \frac{6}{5}$ is the first integer-restricted variable that has a noninteger value. Adding one of the constraints $x_2 \leq 1$ or $x_2 \geq 2$ then creates the following two new subproblems.

Subproblem 3:
Original problem plus additional constraints

$$x_1 \leq 1, \qquad x_2 \leq 1.$$

Subproblem 4:
Original problem plus additional constraints

$$x_1 \leq 1, \qquad x_2 \geq 2.$$

Solving their LP relaxations gives the following results.

LP relaxation of subproblem 3: $(x_1, x_2, x_3, x_4) = \left(\frac{5}{6}, 1, \frac{11}{6}, 0\right)$, with $Z = 14\frac{1}{6}$.

Bound for subproblem 3: $Z \leq 14\frac{1}{6}$.

LP relaxation of subproblem 4: $(x_1, x_2, x_3, x_4) = \left(\frac{5}{6}, 2, \frac{11}{6}, 0\right)$, with $Z = 12\frac{1}{6}$.

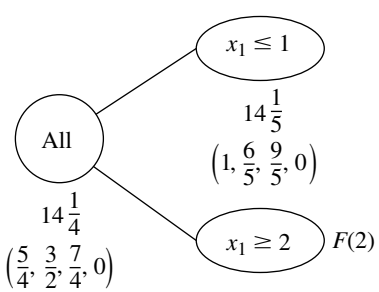
Bound for subproblem 4: $Z \leq 12\frac{1}{6}$.

Because both solutions exist (feasible solutions) and have noninteger values for integer-restricted variables, neither subproblem is fathomed. (Test 1 still is not operational, since $Z^* = -\infty$ until the first incumbent is found.)

The solution tree at this point is given in Fig. 12.12.

Iteration 3. With two remaining subproblems (3 and 4) that were created simultaneously, the one with the larger bound (subproblem 3, with $14\frac{1}{6} > 12\frac{1}{6}$) is selected for the

FIGURES 12.11
The solution tree after the first iteration of the MIP branch-and-bound algorithm for the MIP example.



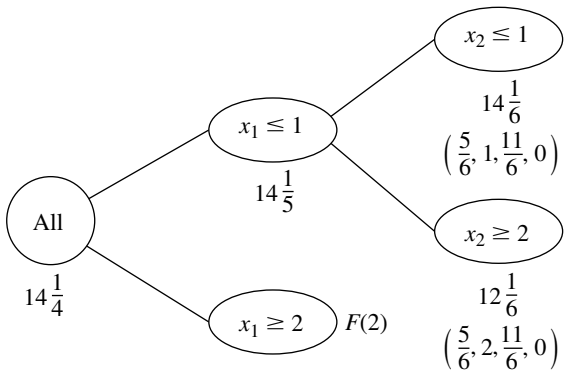


FIGURE 12.12
The solution tree after the second iteration of the MIP branch-and-bound algorithm for the MIP example.

next branching. Because $x_1 = \frac{5}{6}$ has a noninteger value in the optimal solution for this subproblem's LP relaxation, x_1 becomes the branching variable. (Note that x_1 now is a *recurring* branching variable, since it also was chosen at iteration 1.) This leads to the following new subproblems.

Subproblem 5:
Original problem plus additional constraints

$x_1 \le 1$
$x_2 \le 1$
$x_1 \le 0$ (so $x_1 = 0$).

Subproblem 6:
Original problem plus additional constraints

$x_1 \le 1$
$x_2 \le 1$
$x_1 \ge 1$ (so $x_1 = 1$).

The results from solving their LP relaxations are given below.

LP relaxation of subproblem 5: $(x_1, x_2, x_3, x_4) = \left(0, 0, 2, \frac{1}{2}\right)$, with $Z = 13\frac{1}{2}$.

Bound for subproblem 5: $Z \le 13\frac{1}{2}$.

LP relaxation of subproblem 6: No feasible solutions.

Subproblem 6 is immediately fathomed by test 2. However, note that subproblem 5 also can be fathomed. Test 3 passes because the optimal solution for its LP relaxation has integer values ($x_1 = 0$, $x_2 = 0$, $x_3 = 2$) for all three integer-restricted variables. (It does not matter that $x_4 = \frac{1}{2}$, since x_4 is not integer-restricted.) This *feasible* solution for the original problem becomes our first incumbent:

Incumbent $= \left(0, 0, 2, \frac{1}{2}\right)$ with $Z^* = 13\frac{1}{2}$.

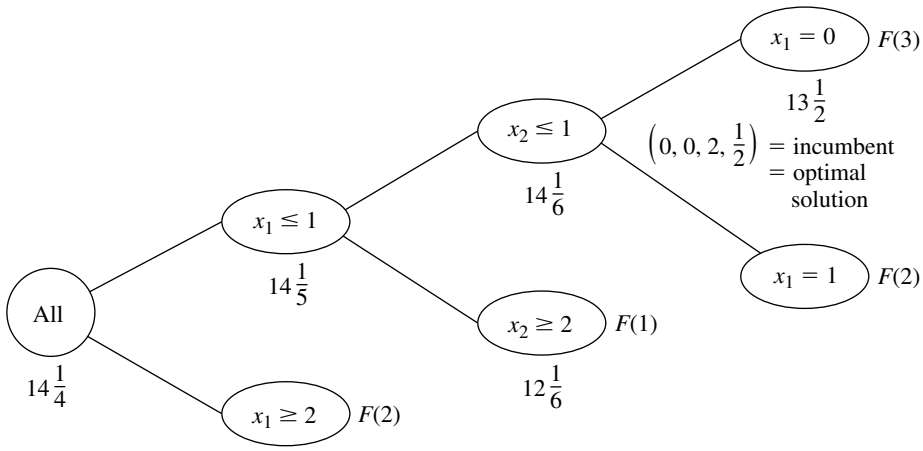


FIGURE 12.13
The solution tree after the final (third) iteration of the MIP branch-and-bound algorithm for the MIP example.

Using this Z^* to reapply fathoming test 1 to the only other subproblem (subproblem 4) is successful, because its bound $12\frac{1}{6} \le Z^*$.

This iteration has succeeded in fathoming subproblems in all three possible ways. Furthermore, there now are no remaining subproblems, so the current incumbent is optimal.

$$\text{Optimal solution} = \left(0, 0, 2, \frac{1}{2}\right) \qquad \text{with } Z = 13\frac{1}{2}.$$

These results are summarized by the final solution tree given in Fig. 12.13.

Another example of applying the MIP algorithm is presented in your OR Tutor. The OR Courseware also includes an interactive routine for executing this algorithm.

12.8 OTHER DEVELOPMENTS IN SOLVING BIP PROBLEMS

Integer programming has been an especially exciting area of OR since the mid-1980s because of the dramatic progress being made in its solution methodology.

Background

To place this progress into perspective, consider the historical background. One big breakthrough had come in the 1960s and early 1970s with the development and refinement of the branch-and-bound approach. But then the state of the art seemed to hit a plateau. Relatively small problems (well under 100 variables) could be solved very efficiently, but even a modest increase in problem size might cause an explosion in computation time beyond feasible limits. Little progress was being made in overcoming this exponential growth in computation time as the problem size was increased. Many important problems arising in practice could not be solved.

Then came the next breakthrough in the mid-1980s, as reported largely in four papers published in 1983, 1985, 1987, and 1991. (See Selected References 3, 6, 10, and 5.)

In the 1983 paper, Harlan Crowder, Ellis Johnson, and Manfred Padberg presented a new algorithmic approach to solving *pure* BIP problems that had successfully solved problems with no apparent special structure having up to 2,756 variables! This paper won the Lanchester Prize, awarded by the Operations Research Society of America for the most notable publication in operations research during 1983. In the 1985 paper, Ellis Johnson, Michael Kostreva, and Uwe Suhl further refined this algorithmic approach.

However, both of these papers were limited to *pure* BIP. For IP problems arising in practice, it is quite common for all the integer-restricted variables to be *binary,* but a large proportion of these problems are *mixed* BIP problems. What was critically needed was a way of extending this same kind of algorithmic approach to *mixed* BIP. This came in the 1987 paper by Tony Van Roy and Laurence Wolsey of Belgium. Once again, problems of *very* substantial size (up to nearly 1,000 binary variables and a larger number of continuous variables) were being solved successfully. And once again, this paper won a very prestigious award, the Orchard-Hays Prize given triannually by the Mathematical Programming Society.

In the 1991 paper, Karla Hoffman and Manfred Padberg followed up on the 1983 and 1985 papers by developing improved techniques for solving pure BIP problems. Using the name **branch-and-cut algorithm** for this algorithmic approach, they reported successfully solving problems with as many as 6,000 variables!

We do need to add one note of caution. This algorithmic approach cannot consistently solve *all* pure BIP problems with a few thousand variables, or even a few hundred variables. The very large pure BIP problems solved had *sparse* **A** matrices; i.e., the percentage of coefficients in the functional constraints that were *nonzeros* was quite small (perhaps less than 5 percent). In fact, the approach depends heavily upon this sparsity. (Fortunately, this kind of sparsity is typical in large practical problems.) Furthermore, there are other important factors besides sparsity and size that affect just how difficult a given IP problem will be to solve. IP formulations of fairly substantial size should still be approached with considerable caution.

On the other hand, each new algorithmic breakthrough in OR always generates a flurry of new research and development activity to try to refine the new approach further. We have seen substantial effort to develop sophisticated software packages for widespread use. For example, the kinds of IP techniques discussed above have been incorporated into the IP module of IBM's Optimization Subroutine Library (OSL). The developers of CPLEX have an ongoing project to maintain a fully state-of-the-art IP module. Theoretical research also continues.

Throughout the 1990s, we have seen further fruits of these intensified research and development activities in integer programming. Larger and larger problems are being solved. For example, at the end of that decade, CPLEX 6.5 successfully used a sophisticated branch-and-cut algorithm to solve a real-world problem with over 4,000 functional constraints and over 120,000 binary variables! MIP problems with thousands of general integer variables, along with numerous continuous variables and binary variables, also were being solved. (Selected Reference 2 provides details.)

Although it would be beyond the scope and level of this book to fully describe the algorithmic approach discussed above, we will now give a brief overview. (You are encouraged to read Selected References 2, 3, 5, 6, and 10 for further information.) This overview is limited to *pure* BIP, so *all* variables introduced later in this section are *binary* variables.

The approach mainly uses a combination of three kinds[1] of techniques: *automatic problem preprocessing,* the *generation of cutting planes,* and clever *branch-and-bound* techniques. You already are familiar with branch-and-bound techniques, and we will not elaborate further on the more advanced versions incorporated here. An introduction to the other two kinds of techniques is given below.

Automatic Problem Preprocessing for Pure BIP

Automatic problem preprocessing involves a "computer inspection" of the user-supplied formulation of the IP problem in order to spot reformulations that make the problem quicker to solve without eliminating any feasible solutions. These reformulations fall into three categories:

1. *Fixing variables:* Identify variables that can be fixed at one of their possible values (either 0 or 1) because the other value cannot possibly be part of a solution that is both feasible and optimal.
2. *Eliminating redundant constraints:* Identify and eliminate *redundant constraints* (constraints that automatically are satisfied by solutions that satisfy all the other constraints).
3. *Tightening constraints:* Tighten some constraints in a way that reduces the feasible region for the LP relaxation without eliminating any feasible solutions for the BIP problem.

These categories are described in turn.

Fixing Variables. One general principle for fixing variables is the following.

> If one value of a variable cannot satisfy a certain constraint, even when the other variables equal their best values for trying to satisfy the constraint, then that variable should be fixed at its other value.

For example, *each* of the following $\leq$ constraints would enable us to fix x_1 at $x_1 = 0$, since $x_1 = 1$ with the best values of the other variables (0 with a nonnegative coefficient and 1 with a negative coefficient) would violate the constraint.

$$\begin{aligned} 3x_1 &\leq 2 \quad \Rightarrow \quad x_1 = 0, \quad \text{since} \quad 3(1) > 2. \\ 3x_1 + x_2 &\leq 2 \quad \Rightarrow \quad x_1 = 0, \quad \text{since} \quad 3(1) + 1(0) > 2. \\ 5x_1 + x_2 - 2x_3 &\leq 2 \quad \Rightarrow \quad x_1 = 0, \quad \text{since} \quad 5(1) + 1(0) - 2(1) > 2. \end{aligned}$$

The general procedure for checking any $\leq$ constraint is to identify the variable with the *largest positive coefficient,* and if the *sum* of *that coefficient* and any *negative coefficients* exceeds the right-hand side, then that variable should be fixed at 0. (Once the variable has been fixed, the procedure can be repeated for the variable with the next largest positive coefficient, etc.)

An analogous procedure with $\geq$ constraints can enable us to fix a variable at 1 instead, as illustrated below three times.

$$\begin{aligned} 3x_1 &\geq 2 \quad \Rightarrow \quad x_1 = 1, \quad \text{since} \quad 3(0) < 2. \\ 3x_1 + x_2 &\geq 2 \quad \Rightarrow \quad x_1 = 1, \quad \text{since} \quad 3(0) + 1(1) < 2. \\ 3x_1 + x_2 - 2x_3 &\geq 2 \quad \Rightarrow \quad x_1 = 1, \quad \text{since} \quad 3(0) + 1(1) - 2(0) < 2. \end{aligned}$$

[1]As discussed briefly in Sec. 12.4, still another technique that has played a significant role in the recent progress has been the use of *heuristics* for quickly finding good feasible solutions.

A $\geq$ constraint also can enable us to fix a variable at 0, as illustrated next.

$$x_1 + x_2 - 2x_3 \geq 1 \quad \Rightarrow \quad x_3 = 0, \quad \text{since} \quad 1(1) + 1(1) - 2(1) < 1.$$

The next example shows a $\geq$ constraint fixing one variable at 1 and another at 0.

$$\begin{aligned} 3x_1 + x_2 - 3x_3 \geq 2 \quad &\Rightarrow \quad x_1 = 1, \quad \text{since} \quad 3(0) + 1(1) - 3(0) < 2 \\ \textit{and} \quad &\Rightarrow \quad x_3 = 0, \quad \text{since} \quad 3(1) + 1(1) - 3(1) < 2. \end{aligned}$$

Similarly, a $\leq$ constraint with a *negative* right-hand side can result in either 0 or 1 becoming the fixed value of a variable. For example, both happen with the following constraint.

$$\begin{aligned} 3x_1 - 2x_2 \leq -1 \quad &\Rightarrow \quad x_1 = 0, \quad \text{since} \quad 3(1) - 2(1) > -1 \\ \textit{and} \quad &\Rightarrow \quad x_2 = 1, \quad \text{since} \quad 3(0) - 2(0) > -1. \end{aligned}$$

Fixing a variable from one constraint can sometimes generate a chain reaction of then being able to fix other variables from other constraints. For example, look at what happens with the following three constraints.

$$3x_1 + x_2 - 2x_3 \geq 2 \quad \Rightarrow \quad x_1 = 1 \quad \text{(as above)}.$$

Then

$$x_1 + x_4 + x_5 \leq 1 \quad \Rightarrow \quad x_4 = 0, \quad x_5 = 0.$$

Then

$$-x_5 + x_6 \leq 0 \quad \Rightarrow \quad x_6 = 0.$$

In some cases, it is possible to combine one or more *mutually exclusive alternatives* constraints with another constraint to fix a variable, as illustrated below,

$$\left.\begin{aligned} 8x_1 - 4x_2 - 5x_3 + 3x_4 &\leq 2 \\ x_2 + x_3 &\leq 1 \end{aligned}\right\} \quad \Rightarrow \quad x_1 = 0,$$

$$\text{since} \quad 8(1) - \max\{4, 5\}(1) + 3(0) > 2.$$

There are additional techniques for fixing variables, including some involving optimality considerations, but we will not delve further into this topic.

Fixing variables can have a dramatic impact on reducing the size of a problem. One example is the problem with 2,756 variables reported in Selected Reference 3. A major factor in being able to solve this problem is that the algorithm succeeded in fixing 1,341 variables, thereby eliminating essentially half of the problem's variables from further consideration.

Eliminating Redundant Constraints. Here is one easy way to detect a redundant constraint.

> If a functional constraint satisfies even the most challenging binary solution, then it has been made redundant by the binary constraints and can be eliminated from further consideration. For a $\leq$ constraint, the most challenging binary solution has variables equal to 1 when they have nonnegative coefficients and other variables equal to 0. (Reverse these values for a $\geq$ constraint.)

Some examples are given below.

$3x_1 + 2x_2 \leq 6$ is redundant, since $3(1) + 2(1) \leq 6$.
$3x_1 - 2x_2 \leq 3$ is redundant, since $3(1) - 2(0) \leq 3$.
$3x_1 - 2x_2 \geq -3$ is redundant, since $3(0) - 2(1) \geq -3$.

In most cases where a constraint has been identified as redundant, it was not redundant in the original model but became so after fixing some variables. Of the 11 examples of fixing variables given above, *all* but the last one left a constraint that then was redundant.

Tightening Constraints.[1] Consider the following problem.

Maximize $Z = 3x_1 + 2x_2$,

subject to

$2x_1 + 3x_2 \leq 4$

and

x_1, x_2 binary.

This BIP problem has just three feasible solutions—(0, 0), (1, 0), and (0, 1)—where the optimal solution is (1, 0) with $Z = 3$. The feasible region for the LP relaxation of this problem is shown in Fig. 12.14. The optimal solution for this LP relaxation is $(1, \frac{2}{3})$ with $Z = 4\frac{1}{3}$, which is not very close to the optimal solution for the BIP problem. A branch-and-bound algorithm would have some work to do to identify the optimal BIP solution.

[1]Also commonly called *coefficient reduction.*

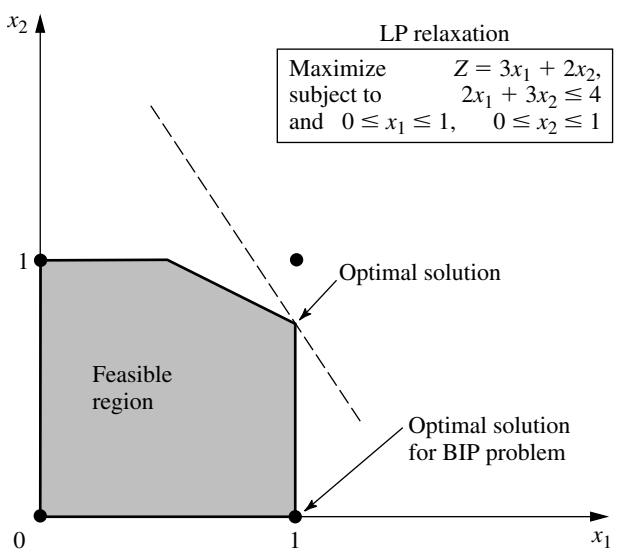


FIGURE 12.14
The LP relaxation (including its feasible region and optimal solution) for the BIP example used to illustrate tightening a constraint.

Now look what happens when the functional constraint $2x_1 + 3x_2 \leq 4$ is replaced by

$$x_1 + x_2 \leq 1.$$

The feasible solutions for the BIP problem remain exactly the same—(0, 0), (1, 0), and (0, 1)—so the optimal solution still is (1, 0). However, the feasible region for the LP relaxation has been greatly reduced, as shown in Fig. 12.15. In fact, this feasible region has been reduced so much that the optimal solution for the LP relaxation now is (1, 0), so the optimal solution for the BIP problem has been found without needing any additional work.

This is an example of tightening a constraint in a way that reduces the feasible region for the LP relaxation without eliminating any feasible solutions for the BIP problem. It was easy to do for this tiny two-variable problem that could be displayed graphically. However, with application of the same principles for tightening a constraint without eliminating any feasible BIP solutions, the following algebraic procedure can be used to do this for any $\leq$ constraint with any number of variables.

Procedure for Tightening a $\leq$ Constraint

Denote the constraint by $a_1x_1 + a_2x_2 + \cdots + a_nx_n \leq b$.

1. Calculate S = sum of the *positive* a_j.
2. Identify any $a_j \neq 0$ such that $S < b + |a_j|$.
   (*a*) If none, stop; the constraint cannot be tightened further.
   (*b*) If $a_j > 0$, go to step 3.
   (*c*) If $a_j < 0$, go to step 4.
3. ($a_j > 0$) Calculate $\overline{a}_j = S - b$ and $\overline{b} = S - a_j$. Reset $a_j = \overline{a}_j$ *and* $b = \overline{b}$. Return to step 1.
4. ($a_j < 0$) Increase a_j to $a_j = b - S$. Return to step 1.

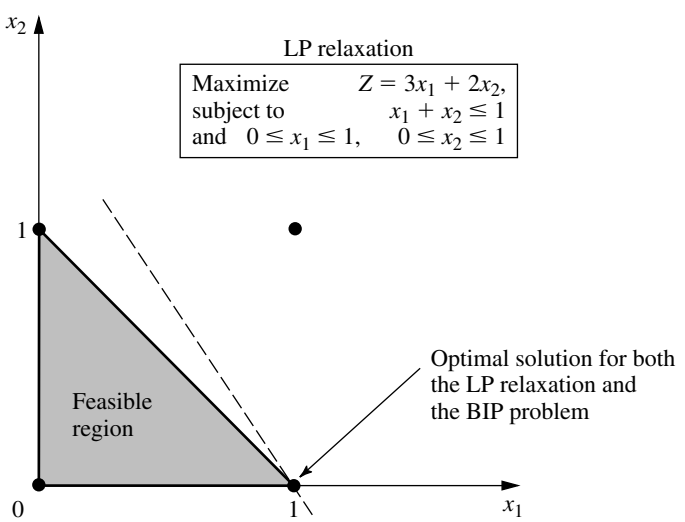


FIGURE 12.15
The LP relaxation after tightening the constraint, $2x_1 + 3x_2 \leq 4$, to $x_1 + x_2 \leq 1$ for the example of Fig. 12.14.

Applying this procedure to the functional constraint in the above example flows as follows:

The constraint is $2x_1 + 3x_2 \leq 4 \qquad (a_1 = 2, a_2 = 3, b = 4)$.

1. $S = 2 + 3 = 5$.
2. a_1 satisfies $S < b + |a_1|$, since $5 < 4 + 2$. Also a_2 satisfies $S < b + |a_2|$, since $5 < 4 + 3$. Choose a_1 arbitrarily.
3. $\overline{a}_1 = 5 - 4 = 1$ and $\overline{b} = 5 - 2 = 3$, so reset $a_1 = 1$ and $b = 3$. The new tighter constraint is

$$x_1 + 3x_2 \leq 3 \qquad (a_1 = 1, a_2 = 3, b = 3).$$

1. $S = 1 + 3 = 4$.
2. a_2 satisfies $S < b + |a_2|$, since $4 < 3 + 3$.
3. $\overline{a}_2 = 4 - 3 = 1$ and $\overline{b} = 4 - 3 = 1$, so reset $a_2 = 1$ and $b = 1$. The new tighter constraint is

$$x_1 + x_2 \leq 1 \qquad (a_1 = 1, a_2 = 1, b = 1).$$

1. $S = 1 + 1 = 2$.
2. No $a_j \neq 0$ satisfies $S < b + |a_j|$, so stop; $x_1 + x_2 \leq 1$ is the desired tightened constraint.

If the first execution of step 2 in the above example had chosen a_2 instead, then the first tighter constraint would have been $2x_1 + x_2 \leq 2$. The next series of steps again would have led to $x_1 + x_2 \leq 1$.

In the next example, the procedure tightens the constraint on the left to become the one on its right and then tightens further to become the second one on the right.

$$\begin{aligned} 4x_1 - 3x_2 + x_3 + 2x_4 \leq 5 \quad &\Rightarrow \quad 2x_1 - 3x_2 + x_3 + 2x_4 \leq 3 \\ &\Rightarrow \quad 2x_1 - 2x_2 + x_3 + 2x_4 \leq 3. \end{aligned}$$

(Problem 12.8-5 asks you to apply the procedure to confirm these results.)

A constraint in $\geq$ form can be converted to $\leq$ form (by multiplying through both sides by -1) to apply this procedure directly.

Generating Cutting Planes for Pure BIP

A **cutting plane** (or **cut**) for any IP problem is a new functional constraint that reduces the feasible region for the LP relaxation without eliminating any feasible solutions for the IP problem. In fact, you have just seen one way of generating cutting planes for pure BIP problems, namely, apply the above procedure for tightening constraints. Thus, $x_1 + x_2 \leq 1$ is a cutting plane for the BIP problem considered in Fig. 12.14, which leads to the reduced feasible region for the LP relaxation shown in Fig. 12.15.

In addition to this procedure, a number of other techniques have been developed for generating cutting planes that will tend to accelerate how quickly a branch-and-bound algorithm can find an optimal solution for a pure BIP problem. We will focus on just one of these techniques.

To illustrate this technique, consider the California Manufacturing Co. pure BIP problem presented in Sec. 12.1 and used to illustrate the BIP branch-and-bound algorithm in

Sec. 12.6. The optimal solution for its LP relaxation is given in Fig. 12.5 as $(x_1, x_2, x_3, x_4) = (\frac{5}{6}, 1, 0, 1)$. One of the functional constraints is

$$6x_1 + 3x_2 + 5x_3 + 2x_4 \leq 10.$$

Now note that the binary constraints and this constraint together imply that

$$x_1 + x_2 + x_4 \leq 2.$$

This new constraint is a *cutting plane.* It eliminates part of the feasible region for the LP relaxation, including what had been the optimal solution, $(\frac{5}{6}, 1, 0, 1)$, but it does not eliminate any feasible *integer* solutions. Adding just this one cutting plane to the original model would improve the performance of the BIP branch-and-bound algorithm in Sec. 12.6 (see Fig. 12.9) in two ways. First, the optimal solution for the new (tighter) LP relaxation would be $(1, 1, \frac{1}{5}, 0)$, with $Z = 15\frac{1}{5}$, so the bounds for the *All* node, $x_1 = 1$ node, and $(x_1, x_2) = (1, 1)$ node now would be 15 instead of 16. Second, one less iteration would be needed because the optimal solution for the LP relaxation at the $(x_1, x_2, x_3) = (1, 1, 0)$ node now would be (1, 1, 0, 0), which provides a new *incumbent* with $Z^* = 14$. Therefore, on the *third* iteration (see Fig. 12.8), this node would be fathomed by test 3, and the $(x_1, x_2) = (1, 0)$ node would be fathomed by test 1, thereby revealing that this incumbent is the optimal solution for the original BIP problem.

Here is the general procedure used to generate this cutting plane.

A Procedure for Generating Cutting Planes

1. Consider any functional constraint in $\leq$ form with only nonnegative coefficients.
2. Find a group of variables (called a **minimum cover** of the constraint) such that
   (*a*) The constraint is violated if every variable in the group equals 1 and all other variables equal 0.
   (*b*) But the constraint becomes satisfied if the value of *any one* of these variables is changed from 1 to 0.
3. By letting N denote the number of variables in the group, the resulting cutting plane has the form

   Sum of variables in group $\leq N - 1$.

Applying this procedure to the constraint $6x_1 + 3x_2 + 5x_3 + 2x_4 \leq 10$, we see that the group of variables $\{x_1, x_2, x_4\}$ is a *minimal cover* because

(*a*) (1, 1, 0, 1) violates the constraint.
(*b*) But the constraint becomes satisfied if the value of *any one* of these three variables is changed from 1 to 0.

Since $N = 3$ in this case, the resulting cutting plane is $x_1 + x_2 + x_4 \leq 2$.

This same constraint also has a second minimal cover $\{x_1, x_3\}$, since (1, 0, 1, 0) violates the constraint but both (0, 0, 1, 0) and (1, 0, 0, 0) satisfy the constraint. Therefore, $x_1 + x_3 \leq 1$ is another valid cutting plane.

The new algorithmic approach presented in Selected References 3, 6, 10, 5, and 2 involves generating *many* cutting planes in a similar manner before then applying clever branch-and-bound techniques. The results of including the cutting planes can be quite

dramatic in tightening the LP relaxations. For example, for the test problem with 2,756 binary variables considered in Selected Reference 3,326 cutting planes were generated. The result was that the *gap* between Z for the optimal solution for the LP relaxation of the whole BIP problem and Z for this problem's optimal solution was reduced by 98 percent. Similar results were obtained on about half of the problems considered in Selected Reference 3.

Ironically, the very first algorithms developed for integer programming, including Ralph Gomory's celebrated algorithm announced in 1958, were based on cutting planes (generated in a different way), but this approach proved to be unsatisfactory in practice (except for special classes of problems). However, these algorithms relied solely on cutting planes. We now know that judiciously *combining* cutting planes and branch-and-bound techniques (along with automatic problem preprocessing) provides a powerful algorithmic approach for solving large-scale BIP problems. This is one reason that the name *branch-and-cut algorithm* has been given to this new approach.

12.9 CONCLUSIONS

IP problems arise frequently because some or all of the decision variables must be restricted to integer values. There also are many applications involving yes-or-no decisions (including combinatorial relationships expressible in terms of such decisions) that can be represented by binary (0–1) variables. These factors have made integer programming one of the most widely used OR techniques.

IP problems are more difficult than they would be without the integer restriction, so the algorithms available for integer programming are generally much less efficient than the simplex method. The most important determinants of computation time are the *number of integer variables* and whether the problem has some *special structure* that can be exploited. For a fixed number of integer variables, BIP problems generally are much easier to solve than problems with general integer variables, but adding continuous variables (MIP) may not increase computation time substantially. For special types of BIP problems containing a special structure that can be exploited by a *special-purpose algorithm,* it may be possible to solve very large problems (thousands of binary variables) routinely. Other much smaller problems without such special structure may not be solvable.

Computer codes for IP algorithms now are commonly available in mathematical programming software packages. Traditionally, these algorithms usually have been based on the *branch-and-bound* technique and variations thereof.

A new era in IP solution methodology has now been ushered in by a series of landmark papers since the mid-1980s. The new *branch-and-cut* algorithmic approach involves combining automatic problem preprocessing, the generation of cutting planes, and clever branch-and-bound techniques. Research in this area is continuing, along with the development of sophisticated new software packages that incorporate these new techniques.

In recent years, there has been considerable investigation into the development of algorithms (including heuristic algorithms) for integer *nonlinear* programming, and this area continues to be a very active area of research.

SELECTED REFERENCES

1. Beasley, J. E. (ed).: *Advances in Linear and Integer Programming,* Oxford University Press, Oxford, England, 1996.
2. Bixby, R. E., M. Fenelon, Z. Gu, E. Rothberg, and R. Wunderling, "MIP: Theory and Practice Closing the Gap," *Proceedings of IFIP TC7 Conference, Cambridge 1999.*
3. Crowder, H., E. L. Johnson, and M. Padberg: "Solving Large-Scale Zero-One Linear Programming Problems," *Operations Research,* **31:** 803–834, 1983.
4. Hillier, F. S., M. S. Hillier, and G. J. Lieberman: *Introduction to Management Science: A Modeling and Case Studies Approach with Spreadsheets,* Irwin/McGraw-Hill, Burr Ridge, IL, 2000, chaps. 9, 10.
5. Hoffman, K. L., and M. Padberg: "Improving LP-Representations of Zero-One Linear Programs for Branch-and-Cut," *ORSA Journal on Computing,* **3:** 121–134, 1991.
6. Johnson, E. L., M. M. Kostreva, and U. H. Suhl: "Solving 0-1 Integer Programming Problems Arising from Large Scale Planning Models," *Operations Research,* **33:** 803–819, 1985.
7. Nemhauser, G. L., and L. A. Wolsey: *Integer and Combinatorial Optimization,* Wiley, New York, 1988.
8. Rayward-Smith, V. J., I. H. Osman, C. R. Reeves, and G. D. Smith (eds.): *Modern Heuristic Search Methods,* Wiley, New York, 1997.
9. Schriver, A.: *Theory of Linear and Integer Programming,* Wiley, New York, 1986.
10. Van Roy, T. J., and L. A. Wolsey: "Solving Mixed 0-1 Programs by Automatic Reformulation," *Operations Research,* **35:** 45–57, 1987.
11. Williams, H. P.: *Model Building in Mathematical Programming,* 3d ed., Wiley, New York, 1990.

LEARNING AIDS FOR THIS CHAPTER IN YOUR OR COURSEWARE

Demonstration Examples in OR Tutor:

Binary Integer Programming Branch-and-Bound Algorithm
Mixed Integer Programming Branch-and-Bound Algorithm

Interactive Routines:

Enter or Revise an Integer Programming Model
Solve Binary Integer Program Interactively
Solve Mixed Integer Program Interactively

An Excel Add-in:

Premium Solver

"Ch. 12—Integer Programming" Files for Solving the Examples:

Excel File
LINGO/LINDO File
MPL/CPLEX File

See Appendix 1 for documentation of the software.

PROBLEMS

The symbols to the left of some of the problems (or their parts) have the following meaning:

D: The corresponding demonstration example listed above may be helpful.
I: We suggest that you use the corresponding interactive routine listed above (the printout records your work).
C: Use the computer with any of the software options available to you (or as instructed by your instructor) to solve the problem.

An asterisk on the problem number indicates that at least a partial answer is given in the back of the book.

12.1-1. Reconsider the California Manufacturing Co. example presented in Sec. 12.1. The mayor of San Diego now has contacted the company's president to try to persuade him to build a factory and perhaps a warehouse in that city. With the tax incentives being offered the company, the president's staff estimates that the net present value of building a factory in San Diego would be $7 million and the amount of capital required to do this would be $4 million. The net present value of building a warehouse there would be $5 million and the capital required would be $3 million. (This option would be considered only if a factory also is being built there.)

The company president now wants the previous OR study revised to incorporate these new alternatives into the overall problem. The objective still is to find the feasible combination of investments that maximizes the total net present value, given that the amount of capital available for these investments is $10 million.
(a) Formulate a BIP model for this problem.
(b) Display this model on an Excel spreadsheet.
C **(c)** Use the computer to solve this model.

12.1-2* A young couple, Eve and Steven, want to divide their main household chores (marketing, cooking, dishwashing, and laundering) between them so that each has two tasks but the total time they spend on household duties is kept to a minimum. Their efficiencies on these tasks differ, where the time each would need to perform the task is given by the following table:

	Time Needed per Week			
	Marketing	**Cooking**	**Dishwashing**	**Laundry**
Eve	4.5 hours	7.8 hours	3.6 hours	2.9 hours
Steven	4.9 hours	7.2 hours	4.3 hours	3.1 hours

(a) Formulate a BIP model for this problem.
(b) Display this model on an Excel spreadsheet.
C **(c)** Use the computer to solve this model.

12.1-3. A real estate development firm, Peterson and Johnson, is considering five possible development projects. The following table shows the estimated long-run profit (net present value) that each project would generate, as well as the amount of investment required to undertake the project, in units of millions of dollars.

	Development Project				
	1	**2**	**3**	**4**	**5**
Estimated profit	1	1.8	1.6	0.8	1.4
Capital required	6	12	10	4	8

The owners of the firm, Dave Peterson and Ron Johnson, have raised $20 million of investment capital for these projects. Dave and Ron now want to select the combination of projects that will maximize their total estimated long-run profit (net present value) without investing more that $20 million.
(a) Formulate a BIP model for this problem.
(b) Display this model on an Excel spreadsheet.
C **(c)** Use the computer to solve this model.

12.1-4. The board of directors of General Wheels Co. is considering seven large capital investments. Each investment can be made only once. These investments differ in the estimated long-run profit (net present value) that they will generate as well as in the amount of capital required, as shown by the following table (in units of millions of dollars):

	Investment Opportunity						
	1	**2**	**3**	**4**	**5**	**6**	**7**
Estimated profit	17	10	15	19	7	13	9
Capital required	43	28	34	48	17	32	23

The total amount of capital available for these investments is $100 million. Investment opportunities 1 and 2 are mutually exclusive, and so are 3 and 4. Furthermore, neither 3 nor 4 can be undertaken unless one of the first two opportunities is undertaken. There are no such restrictions on investment opportunities 5, 6, and 7. The objective is to select the combination of capital investments that will maximize the total estimated long-run profit (net present value).
(a) Formulate a BIP model for this problem.
C **(b)** Use the computer to solve this model.

12.1-5. Reconsider Prob. 8.3-4, where a swim team coach needs to assign swimmers to the different legs of a 200-yard medley relay team. Formulate a BIP model for this problem. Identify the groups of mutually exclusive alternatives in this formulation.

12.1-6. Vincent Cardoza is the owner and manager of a machine shop that does custom order work. This Wednesday afternoon, he has received calls from two customers who would like to place rush orders. One is a trailer hitch company which would like some custom-made heavy-duty tow bars. The other is a mini-car-carrier company which needs some customized stabilizer bars. Both customers would like as many as possible by the end of the week (two working days). Since both products would require the use of the same two machines, Vincent needs to decide and inform the customers this afternoon about how many of each product he will agree to make over the next two days.

Each tow bar requires 3.2 hours on machine 1 and 2 hours on machine 2. Each stabilizer bar requires 2.4 hours on machine 1 and 3 hours on machine 2. Machine 1 will be available for 16 hours over the next two days and machine 2 will be available for 15 hours. The profit for each tow bar produced would be $130 and the profit for each stabilizer bar produced would be $150.

Vincent now wants to determine the mix of these production quantities that will maximize the total profit.

(a) Formulate an IP model for this problem.

(b) Use a graphical approach to solve this model.

C **(c)** Use the computer to solve the model.

12.1-7. Pawtucket University is planning to buy new copier machines for its library. Three members of its Operations Research Department are analyzing what to buy. They are considering two different models: Model A, a high-speed copier, and Model B, a lower-speed but less expensive copier. Model A can handle 20,000 copies a day, and costs $6,000. Model B can handle 10,000 copies a day, but costs only $4,000. They would like to have at least six copiers so that they can spread them throughout the library. They also would like to have at least one high-speed copier. Finally, the copiers need to be able to handle a capacity of at least 75,000 copies per day. The objective is to determine the mix of these two copiers which will handle all these requirements at minimum cost.

(a) Formulate an IP model for this problem.

(b) Use a graphical approach to solve this model.

C **(c)** Use the computer to solve the model.

12.1-8. Reconsider Prob. 8.2-23 involving a contractor (Susan Meyer) who needs to arrange for hauling gravel from two pits to three building sites.

Susan now needs to hire the trucks (and their drivers) to do the hauling. Each truck can only be used to haul gravel from a single pit to a single site. In addition to the hauling and gravel costs specified in Prob. 8.2-23, there now is a fixed cost of $50 associated with hiring each truck. A truck can haul 5 tons, but it is not required to go full. For each combination of pit and site, there are now two decisions to be made: the number of trucks to be used and the amount of gravel to be hauled.

(a) Formulate an MIP model for this problem.

C **(b)** Use the computer to solve this model.

12.2-1. Select one of the actual applications of BIP by a company or governmental agency mentioned in Sec. 12.2. Read the article describing the application in the referenced issue of *Interfaces.* Write a two-page summary of the application and its benefits.

12.2-2. Select three of the actual applications of BIP by a company or governmental agency mentioned in Sec. 12.2. Read the articles describing the applications in the referenced issues of *Interfaces.* For each one, write a one-page summary of the application and its benefits.

12.3-1.* The Research and Development Division of the Progressive Company has been developing four possible new product lines. Management must now make a decision as to which of these four products actually will be produced and at what levels. Therefore, an operations research study has been requested to find the most profitable product mix.

A substantial cost is associated with beginning the production of any product, as given in the first row of the following table. Management's objective is to find the product mix that maximizes the total profit (total net revenue minus start-up costs).

	Product			
	1	2	3	4
Start-up cost	$50,000	$40,000	$70,000	$60,000
Marginal revenue	$ 70	$ 60	$ 90	$ 80

Let the continuous decision variables x_1, x_2, x_3, and x_4 be the production levels of products 1, 2, 3, and 4, respectively. Management has imposed the following policy constraints on these variables:

1. No more than two of the products can be produced.
2. Either product 3 or 4 can be produced only if either product 1 or 2 is produced.
3. Either $5x_1 + 3x_2 + 6x_3 + 4x_4 \leq 6{,}000$
   or $4x_1 + 6x_2 + 3x_3 + 5x_4 \leq 6{,}000.$

(a) Introduce auxiliary binary variables to formulate a mixed BIP model for this problem.

C **(b)** Use the computer to solve this model.

12.3-2. Suppose that a mathematical model fits linear programming except for the restriction that $|x_1 - x_2| = 0$, or 3, or 6. Show how to reformulate this restriction to fit an MIP model.

12.3-3. Suppose that a mathematical model fits linear programming except for the restrictions that

1. At least one of the following two inequalities holds:

$$x_1 + x_2 + x_3 + x_4 \leq 4$$
$$3x_1 - x_2 - x_3 + x_4 \leq 3.$$

2. At least two of the following four inequalities holds:

$$5x_1 + 3x_2 + 3x_3 - x_4 \leq 10$$
$$2x_1 + 5x_2 - x_3 + 3x_4 \leq 10$$
$$-x_1 + 3x_2 + 5x_3 + 3x_4 \leq 10$$
$$3x_1 - x_2 + 3x_3 + 5x_4 \leq 10.$$

Show how to reformulate these restrictions to fit an MIP model.

12.3-4. The Toys-R-4-U Company has developed two new toys for possible inclusion in its product line for the upcoming Christmas season. Setting up the production facilities to begin production would cost \$50,000 for toy 1 and \$80,000 for toy 2. Once these costs are covered, the toys would generate a unit profit of \$10 for toy 1 and \$15 for toy 2.

The company has two factories that are capable of producing these toys. However, to avoid doubling the start-up costs, just one factory would be used, where the choice would be based on maximizing profit. For administrative reasons, the same factory would be used for both new toys if both are produced.

Toy 1 can be produced at the rate of 50 per hour in factory 1 and 40 per hour in factory 2. Toy 2 can be produced at the rate of 40 per hour in factory 1 and 25 per hour in factory 2. Factories 1 and 2, respectively, have 500 hours and 700 hours of production time available before Christmas that could be used to produce these toys.

It is not known whether these two toys would be continued after Christmas. Therefore, the problem is to determine how many units (if any) of each new toy should be produced before Christmas to maximize the total profit.

(a) Formulate an MIP model for this problem.

C **(b)** Use the computer to solve this model.

12.3-5.* Northeastern Airlines is considering the purchase of new long-, medium-, and short-range jet passenger airplanes. The purchase price would be \$67 million for each long-range plane, \$50 million for each medium-range plane, and \$35 million for each short-range plane. The board of directors has authorized a maximum commitment of \$1.5 billion for these purchases. Regardless of which airplanes are purchased, air travel of all distances is expected to be sufficiently large that these planes would be utilized at essentially maximum capacity. It is estimated that the net annual profit (after capital recovery costs are subtracted) would be \$4.2 million per long-range plane, \$3 million per medium-range plane, and \$2.3 million per short-range plane.

It is predicted that enough trained pilots will be available to the company to crew 30 new airplanes. If only short-range planes were purchased, the maintenance facilities would be able to handle 40 new planes. However, each medium-range plane is equivalent to $1\frac{1}{3}$ short-range planes, and each long-range plane is equivalent to $1\frac{2}{3}$ short-range planes in terms of their use of the maintenance facilities.

The information given here was obtained by a preliminary analysis of the problem. A more detailed analysis will be conducted subsequently. However, using the preceding data as a first approximation, management wishes to know how many planes of each type should be purchased to maximize profit.

(a) Formulate an IP model for this problem.

C **(b)** Use the computer to solve this problem.

(c) Use a binary representation of the variables to reformulate the IP model in part (*a*) as a BIP problem.

C **(d)** Use the computer to solve the BIP model formulated in part (*c*). Then use this optimal solution to identify an optimal solution for the IP model formulated in part (*a*).

12.3-6. Consider the two-variable IP example discussed in Sec. 12.5 and illustrated in Fig. 12.3.

(a) Use a binary representation of the variables to reformulate this model as a BIP problem.

C **(b)** Use the computer to solve this BIP problem. Then use this optimal solution to identify an optimal solution for the original IP model.

12.3-7. The Fly-Right Airplane Company builds small jet airplanes to sell to corporations for the use of their executives. To meet the needs of these executives, the company's customers sometimes order a custom design of the airplanes being purchased. When this occurs, a substantial start-up cost is incurred to initiate the production of these airplanes.

Fly-Right has recently received purchase requests from three customers with short deadlines. However, because the company's production facilities already are almost completely tied up filling previous orders, it will not be able to accept all three orders. Therefore, a decision now needs to be made on the number of airplanes the company will agree to produce (if any) for each of the three customers.

The relevant data are given in the next table. The first row gives the start-up cost required to initiate the production of the airplanes for each customer. Once production is under way, the marginal net revenue (which is the purchase price minus the marginal production cost) from each airplane produced is shown in the second row. The third row gives the percentage of the available pro-

duction capacity that would be used for each airplane produced. The last row indicates the maximum number of airplanes requested by each customer (but less will be accepted).

	Customer		
	1	2	3
Start-up cost	$3 million	$2 million	0
Marginal net revenue	$2 million	$3 million	$0.8 million
Capacity used per plane	20%	40%	20%
Maximum order	3 planes	2 planes	5 planes

Fly-Right now wants to determine how many airplanes to produce for each customer (if any) to maximize the company's total profit (total net revenue minus start-up costs).

(a) Formulate a model with both integer variables and binary variables for this problem.

C **(b)** Use the computer to solve this model.

12.4-1. Reconsider the Fly-Right Airplane Co. problem introduced in Prob. 12.3-7. A more detailed analysis of the various cost and revenue factors now has revealed that the potential profit from producing airplanes for each customer cannot be expressed simply in terms of a *start-up cost* and a fixed *marginal net revenue* per airplane produced. Instead, the profits are given by the following table.

Airplanes Produced	Profit from Customer		
	1	2	3
0	0	0	0
1	−$1 million	$1 million	$1 million
2	$2 million	$5 million	$3 million
3	$4 million		$5 million
4			$6 million
5			$7 million

(a) Formulate a BIP model for this problem that includes constraints for *mutually exclusive alternatives.*

C **(b)** Use the computer to solve the model formulated in part (*a*). Then use this optimal solution to identify the optimal number of airplanes to produce for each customer.

(c) Formulate another BIP model for this model that includes constraints for *contingent decisions.*

C **(d)** Repeat part (*b*) for the model formulated in part (*c*).

12.4-2. Reconsider the Wyndor Glass Co. problem presented in Sec. 3.1. Management now has decided that only one of the two new products should be produced, and the choice is to be made on the basis of maximizing profit. Introduce *auxiliary binary variables* to formulate an MIP model for this new version of the problem.

12.4-3.* Reconsider Prob. 3.1-11, where the management of the Omega Manufacturing Company is considering devoting excess production capacity to one or more of three products. (See the Partial Answers to Selected Problems in the back of the book for additional information about this problem.) Management now has decided to add the restriction that no more than two of the three prospective products should be produced.

(a) Introduce *auxiliary binary variables* to formulate an MIP model for this new version of the problem.

C **(b)** Use the computer to solve this model.

12.4-4. Consider the following integer nonlinear programming problem.

$$\text{Maximize} \qquad Z = 4x_1^2 - x_1^3 + 10x_2^2 - x_2^4,$$

subject to

$$x_1 + x_2 \leq 3$$

and

$$x_1 \geq 0, \qquad x_2 \geq 0$$
$$x_1 \text{ and } x_2 \text{ are integers.}$$

This problem can be reformulated in two different ways as an equivalent pure BIP problem (with a linear objective function) with six binary variables (y_{1j} and y_{2j} for $j = 1, 2, 3$), depending on the interpretation given the binary variables.

(a) Formulate a BIP model for this problem where the binary variables have the interpretation,

$$y_{ij} = \begin{cases} 1 & \text{if } x_i = j \\ 0 & \text{otherwise.} \end{cases}$$

C **(b)** Use the computer to solve the model formulated in part (*a*), and thereby identify an optimal solution for (x_1, x_2) for the original problem.

(c) Formulate a BIP model for this problem where the binary variables have the interpretation,

$$y_{ij} = \begin{cases} 1 & \text{if } x_i \geq j \\ 0 & \text{otherwise.} \end{cases}$$

C **(d)** Use the computer to solve the model formulated in part (*c*), and thereby identify an optimal solution for (x_1, x_2) for the original problem.

12.4-5. Consider the following discrete nonlinear programming problem.

$$\text{Maximize} \qquad Z = 2x_1 - x_1^2 + 3x_2 - 3x_2^2,$$

subject to

$$x_1 + x_2 \leq 0.75$$

and

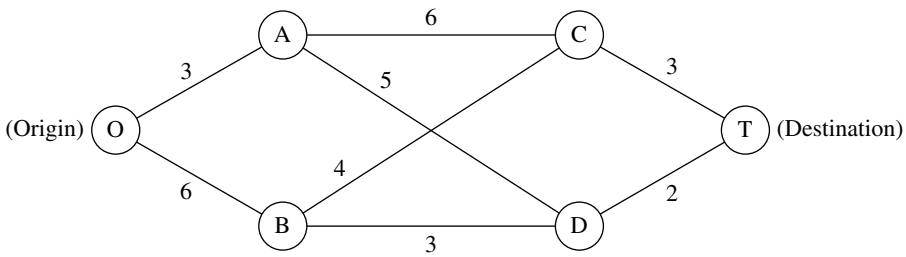
each variable is restricted to the values: $\frac{1}{2}, \frac{1}{3}, \frac{1}{4}, \frac{1}{5}$.

(Continue in the next column.)

(a) Reformulate this problem as a pure binary integer linear programming problem.

C **(b)** Use the computer to solve the model formulated in part (*a*), and thereby identify an optimal solution for (x_1, x_2) for the original problem.

12.4-6.* Consider the following special type of *shortest-path problem* (see Sec. 9.3) where the nodes are in columns and the only paths considered always move forward one column at a time.



The numbers along the links represent distances, and the objective is to find the shortest path from the origin to the destination.

This problem also can be formulated as a BIP model involving both mutually exclusive alternatives and contingent decisions.

(a) Formulate this model. Identify the constraints that are for mutually exclusive alternatives and that are for contingent decisions.

C **(b)** Use the computer to solve this problem.

12.4-7. Consider the project network for a PERT-type system shown in Prob. 11.2-3. Formulate a BIP model for the problem of finding a *critical path* (i.e., a *longest path*) for this project network.

12.4-8. Speedy Delivery provides two-day delivery service of large parcels across the United States. Each morning at each collection center, the parcels that have arrived overnight are loaded onto several trucks for delivery throughout the area. Since the competitive battlefield in this business is speed of delivery, the parcels are divided among the trucks according to their geographical destinations to minimize the average time needed to make the deliveries.

On this particular morning, the dispatcher for the Blue River Valley Collection Center, Sharon Lofton, is hard at work. Her three drivers will be arriving in less than an hour to make the day's deliveries. There are nine parcels to be delivered, all at locations many miles apart. As usual, Sharon has loaded these locations into her computer. She is using her company's special software package, a decision support system called Dispatcher. The first thing Dispatcher does is use these locations to generate a considerable number of attractive possible routes for the individual delivery trucks. These routes are shown in the following table (where the numbers in each column indicate the order of the deliveries), along with the estimated time required to traverse the route.

	Attractive Possible Route									
Delivery Location	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
A	1				1				1	
B		2		1		2			2	2
C			3	3			3		3	
D	2					1		1		
E			2	2		3				
F		1			2					
G	3						1	2		3
H			1		3					1
I		3		4			2			
Time (in hours)	6	4	7	5	4	6	5	3	7	6

Dispatcher is an interactive system that shows these routes to Sharon for her approval or modification. (For example, the computer may not know that flooding has made a particular route infeasible.) After Sharon approves these routes as attractive possibilities with reasonable time estimates, Dispatcher next formulates and solves a BIP model for selecting three routes that minimize their total time while including each delivery location on exactly one route. This morning, Sharon does approve all the routes.

(a) Formulate this BIP model.

C **(b)** Use the computer to solve this model.

12.4-9. An increasing number of Americans are moving to a warmer climate when they retire. To take advantage of this trend, Sunny Skies Unlimited is undertaking a major real estate development project. The project is to develop a completely new retirement community (to be called Pilgrim Haven) that will cover several square miles. One of the decisions to be made is where to locate the two fire stations that have been allocated to the community. For planning purposes, Pilgrim Haven has been divided into five tracts, with no more than one fire station to be located in any given tract. Each station is to respond to *all* the fires that occur in the tract in which it is located as well as in the other tracts that are assigned to this station. Thus, the decisions to be made consist of (1) the tracts to receive a fire station and (2) the assignment of each of the other tracts to one of the fire stations. The objective is to minimize the overall average of the *response times* to fires.

The following table gives the average response time to a fire in each tract (the columns) if that tract is served by a station in a given tract (the rows). The bottom row gives the forecasted average number of fires that will occur in each of the tracts per day.

Assigned Station Located in Tract	Response Times (in minutes) Fire in Tract				
	1	**2**	**3**	**4**	**5**
1	5	12	30	20	15
2	20	4	15	10	25
3	15	20	6	15	12
4	25	15	25	4	10
5	10	25	15	12	5
Average frequency of fires	2 per day	1 per day	3 per day	1 per day	3 per day

Formulate a BIP model for this problem. Identify any constraints that correspond to mutually exclusive alternatives or contingent decisions.

12.4-10. Reconsider Prob. 12.4-9. The management of Sunny Skies Unlimited now has decided that the decision on the locations of the fire stations should be based mainly on costs.

The cost of locating a fire station in a tract is \$200,000 for tract 1, \$250,000 for tract 2, \$400,000 for tract 3, \$300,000 for tract 4, and \$500,000 for tract 5. Management's objective now is the following:

> Determine which tracts should receive a station to minimize the total cost of stations while ensuring that each tract has at least one station close enough to respond to a fire in no more than 15 minutes (on the average).

In contrast to the original problem, note that the total number of fire stations is no longer fixed. Furthermore, if a tract without a station has more than one station within 15 minutes, it is no longer necessary to assign this tract to just one of these stations.

(a) Formulate a complete pure BIP model with 5 binary variables for this problem.

(b) Is this a *set covering problem?* Explain, and identify the relevant sets.

C **(c)** Use the computer to solve the model formulated in part (*a*).

12.4-11. Suppose that a state sends R persons to the U.S. House of Representatives. There are D counties in the state ($D > R$), and the state legislature wants to group these counties into R distinct electoral districts, each of which sends a delegate to Congress. The total population of the state is P, and the legislature wants to form districts whose population approximates $p = P/R$. Suppose that the appropriate legislative committee studying the electoral districting problem generates a long list of N *candidates* to be districts ($N > R$). Each of these candidates contains contiguous counties and a total population p_j ($j = 1, 2, \ldots, N$) that is acceptably close to p. Define $c_j = |p_j - p|$. Each county i ($i = 1, 2, \ldots, D$) is included in at least one candidate and typically will be included in a considerable number of candidates (in order to provide many feasible ways of selecting a set of R candidates that includes each county exactly once). Define

$$a_{ij} = \begin{cases} 1 & \text{if county } i \text{ is included in candidate } j \\ 0 & \text{if not.} \end{cases}$$

Given the values of the c_j and the a_{ij}, the objective is to select R of these N possible districts such that each county is contained in a single district and such that the largest of the associated c_j is as small as possible.

Formulate a BIP model for this problem.

12.4-12. A U.S. professor will be spending a short sabbatical leave at the University of Iceland. She wishes to bring all needed items with her on the airplane. After collecting the professional items that she must have, she finds that airline regulations on space and weight for checked luggage will severely limit the clothes she can take. (She plans to carry on a warm coat and then purchase a warm Icelandic sweater upon arriving in Iceland.) Clothes under consideration for checked luggage include 3 skirts, 3 slacks, 4 tops, and 3 dresses. The professor wants to maximize the number of outfits she will have in Iceland (including the special dress she will wear on the airplane). Each dress constitutes an outfit. Other outfits consist of a combination of a top and either a skirt or slacks. However, certain combinations are not fashionable and so will not qualify as an outfit.

In the following table, the combinations that will make an outfit are marked with an x.

		Top				
		1	2	3	4	Icelandic Sweater
Skirt	1	x	x			x
	2	x			x	
	3		x	x	x	x
Slacks	1	x		x		
	2	x	x		x	x
	3			x	x	x

The weight (in grams) and volume (in cubic centimeters) of each item are shown in the following table:

		Weight	Volume
Skirt	1	600	5,000
	2	450	3,500
	3	700	3,000
Slacks	1	600	3,500
	2	550	6,000
	3	500	4,000
Top	1	350	4,000
	2	300	3,500
	3	300	3,000
	4	450	5,000
Dress	1	600	6,000
	2	700	5,000
	3	800	4,000
Total allowed		4,000	32,000

Formulate a BIP model to choose which items of clothing to take. (*Hint:* After using binary decision variables to represent the individual items, you should introduce *auxiliary* binary variables to represent outfits involving combinations of items. Then use constraints and the objective function to ensure that these auxiliary variables have the correct values, given the values of the decision variables.)

12.5-1.* Consider the following IP problem.

Maximize $Z = 5x_1 + x_2$,

subject to

$$\begin{aligned} -x_1 + 2x_2 &\le 4 \\ x_1 - x_2 &\le 1 \\ 4x_1 + x_2 &\le 12 \end{aligned}$$

and

$$x_1 \ge 0, \quad x_2 \ge 0$$
x_1, x_2 are integers.

(a) Solve this problem graphically.

(b) Solve the LP relaxation graphically. Round this solution to the *nearest* integer solution and check whether it is feasible. Then enumerate *all* the rounded solutions by rounding this solution for the LP relaxation in *all* possible ways (i.e., by rounding each noninteger value both up and down). For each rounded solution, check for feasibility and, if feasible, calculate Z. Are any of these feasible rounded solutions optimal for the IP problem?

12.5-2. Follow the instructions of Prob. 12.5-1 for the following IP problem.

Maximize $Z = 220x_1 + 80x_2$,

subject to

$$\begin{aligned} 5x_1 + 2x_2 &\le 16 \\ 2x_1 - x_2 &\le 4 \\ -x_1 + 2x_2 &\le 4 \end{aligned}$$

and

$$x_1 \ge 0, \quad x_2 \ge 0$$
x_1, x_2 are integers.

12.5-3. Follow the instructions of Prob. 12.5-1 for the following BIP problem.

Maximize $Z = 2x_1 + 5x_2$,

subject to

$$\begin{aligned} 10x_1 + 30x_2 &\le 30 \\ 95x_1 - 30x_2 &\le 75 \end{aligned}$$

and

x_1, x_2 are binary.

12.5-4. Follow the instructions of Prob. 12.5-1 for the following BIP problem.

Maximize $Z = -5x_1 + 25x_2$,

subject to

$$\begin{aligned} -3x_1 + 30x_2 &\le 27 \\ 3x_1 + x_2 &\le 4 \end{aligned}$$

and

x_1, x_2 are binary.

12.5-5. Label each of the following statements as True or False, and then justify your answer by referring to specific statements (with page citations) in the chapter.

(a) Linear programming problems are generally much easier to solve than IP problems.
(b) For IP problems, the number of integer variables is generally more important in determining the computational difficulty than is the number of functional constraints.
(c) To solve an IP problem with an approximate procedure, one may apply the simplex method to the LP relaxation problem and then round each noninteger value to the nearest integer. The result will be a feasible but not necessarily optimal solution for the IP problem.

D,I **12.6-1.*** Use the BIP branch-and-bound algorithm presented in Sec. 12.6 to solve the following problem interactively.

Maximize $Z = 2x_1 - x_2 + 5x_3 - 3x_4 + 4x_5$,

subject to

$$3x_1 - 2x_2 + 7x_3 - 5x_4 + 4x_5 \leq 6$$
$$x_1 - x_2 + 2x_3 - 4x_4 + 2x_5 \leq 0$$

and

x_j is binary, for $j = 1, 2, \ldots, 5$.

D,I **12.6-2.** Use the BIP branch-and-bound algorithm presented in Sec. 12.6 to solve the following problem interactively.

Minimize $Z = 5x_1 + 6x_2 + 7x_3 + 8x_4 + 9x_5$,

subject to

$$3x_1 - x_2 + x_3 + x_4 - 2x_5 \geq 2$$
$$x_1 + 3x_2 - x_3 - 2x_4 + x_5 \geq 0$$
$$-x_1 - x_2 + 3x_3 + x_4 + x_5 \geq 1$$

and

x_j is binary, for $j = 1, 2, \ldots, 5$.

D,I **12.6-3.** Use the BIP branch-and-bound algorithm presented in Sec. 12.6 to solve the following problem interactively.

Maximize $Z = 5x_1 + 5x_2 + 8x_3 - 2x_4 - 4x_5$,

subject to

$$-3x_1 + 6x_2 - 7x_3 + 9x_4 + 9x_5 \geq 10$$
$$x_1 + 2x_2 - x_4 - 3x_5 \leq 0$$

and

x_j is binary, for $j = 1, 2, \ldots, 5$.

D,I **12.6-4.** Reconsider Prob. 12.3-6(a). Use the BIP branch-and-bound algorithm presented in Sec. 12.6 to solve this BIP model interactively.

D,I **12.6-5.** Reconsider Prob. 12.4-10(a). Use the BIP algorithm presented in Sec. 12.6 to solve this problem interactively.

12.6-6. Consider the following statements about any pure IP problem (in maximization form) and its LP relaxation. Label each of the statements as True or False, and then justify your answer.
(a) The feasible region for the LP relaxation is a subset of the feasible region for the IP problem.
(b) If an optimal solution for the LP relaxation is an integer solution, then the optimal value of the objective function is the same for both problems.
(c) If a noninteger solution is feasible for the LP relaxation, then the nearest integer solution (rounding each variable to the nearest integer) is a feasible solution for the IP problem.

12.6-7.* Consider the assignment problem with the following cost table:

		Task				
		1	**2**	**3**	**4**	**5**
Assignee	1	39	65	69	66	57
	2	64	84	24	92	22
	3	49	50	61	31	45
	4	48	45	55	23	50
	5	59	34	30	34	18

(a) Design a branch-and-bound algorithm for solving such assignment problems by specifying how the branching, bounding, and fathoming steps would be performed. (*Hint:* For the assignees not yet assigned for the current subproblem, form the relaxation by deleting the constraints that each of these assignees must perform exactly one task.)
(b) Use this algorithm to solve this problem.

12.6-8. Five jobs need to be done on a certain machine. However, the setup time for each job depends upon which job immediately preceded it, as shown by the following table:

		Setup Time				
		Job				
		1	**2**	**3**	**4**	**5**
Immediately Preceding Job	None	4	5	8	9	4
	1	—	7	12	10	9
	2	6	—	10	14	11
	3	10	11	—	12	10
	4	7	8	15	—	7
	5	12	9	8	16	—

The objective is to schedule the *sequence* of jobs that minimizes the sum of the resulting setup times.

(a) Design a branch-and-bound algorithm for sequencing problems of this type by specifying how the branch, bound, and fathoming steps would be performed.
(b) Use this algorithm to solve this problem.

12.6-9.* Consider the following *nonlinear* BIP problem.

$$\text{Maximize} \quad Z = 80x_1 + 60x_2 + 40x_3 + 20x_4 - (7x_1 + 5x_2 + 3x_3 + 2x_4)^2,$$

subject to

$$x_j \text{ is binary,} \qquad \text{for } j = 1, 2, 3, 4.$$

Given the value of the first k variables $x_1, \ldots, x_k$, where $k = 0$, 1, 2, or 3, an upper bound on the value of Z that can be achieved by the corresponding feasible solutions is

$$\sum_{j=1}^{k} c_j x_j - \left(\sum_{j=1}^{k} d_j x_j\right)^2 + \sum_{j=k+1}^{4} \max\left\{0,\ c_j - \left[\left(\sum_{i=1}^{k} d_i x_i + d_j\right)^2 - \left(\sum_{i=1}^{k} d_i x_i\right)^2\right]\right\},$$

where $c_1 = 80$, $c_2 = 60$, $c_3 = 40$, $c_4 = 20$, $d_1 = 7$, $d_2 = 5$, $d_3 = 3$, $d_4 = 2$. Use this bound to solve the problem by the branch-and-bound technique.

12.6-10. Consider the Lagrangian relaxation described near the end of Sec. 12.6.
(a) If $\mathbf{x}$ is a feasible solution for an MIP problem, show that $\mathbf{x}$ also must be a feasible solution for the corresponding Lagrangian relaxation.
(b) If $\mathbf{x}^*$ is an optimal solution for an MIP problem, with an objective function value of Z, show that $Z \leq Z_R^*$, where Z_R^* is the optimal objective function value for the corresponding Lagrangian relaxation.

12.7-1.* Consider the following IP problem.

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

subject to

$$5x_1 - 7x_2 \geq 3$$

and

$$x_j \leq 3$$
$$x_j \geq 0$$
$$x_j \text{ is integer,} \qquad \text{for } j = 1, 2.$$

(a) Solve this problem graphically.
(b) Use the MIP branch-and-bound algorithm presented in Sec. 12.7 to solve this problem by hand. For each subproblem, solve its LP relaxation *graphically*.
(c) Use the binary representation for integer variables to reformulate this problem as a BIP problem.
D,I **(d)** Use the BIP branch-and-bound algorithm presented in Sec. 12.6 to solve the problem as formulated in part (*c*) interactively.

12.7-2. Follow the instructions of Prob. 12.7-1 for the following IP model.

$$\text{Minimize} \quad Z = 2x_1 + 3x_2,$$

subject to

$$x_1 + x_2 \geq 3$$
$$x_1 + 3x_2 \geq 6$$

and

$$x_1 \geq 0, \qquad x_2 \geq 0$$
$$x_1, x_2 \text{ are integers.}$$

12.7-3. Reconsider the IP model of Prob. 12.5-1.
(a) Use the MIP branch-and-bound algorithm presented in Sec. 12.7 to solve this problem by hand. For each subproblem, solve its LP relaxation *graphically.*
D,I **(b)** Now use the interactive routine for this algorithm in your OR Courseware to solve this problem.
C **(c)** Check your answer by using an automatic routine to solve the problem.

12.7-4. Follow the instructions of Prob. 12.7-3 for the IP model of Prob. 12.5-2.

D,I **12.7-5.** Consider the IP example discussed in Sec. 12.5 and illustrated in Fig. 12.3. Use the MIP branch-and-bound algorithm presented in Sec. 12.7 to solve this problem interactively.

D,I **12.7-6.** Reconsider Prob. 12.3-5*a*. Use the MIP branch-and-bound algorithm presented in Sec. 12.7 to solve this IP problem interactively.

12.7-7. A machine shop makes two products. Each unit of the first product requires 3 hours on machine 1 and 2 hours on machine 2. Each unit of the second product requires 2 hours on machine 1 and 3 hours on machine 2. Machine 1 is available only 8 hours per day and machine 2 only 7 hours per day. The profit per unit sold is 16 for the first product and 10 for the second. The amount of each product produced per day must be an integral multiple of 0.25. The objective is to determine the mix of production quantities that will maximize profit.
(a) Formulate an IP model for this problem.
(b) Solve this model graphically.
(c) Use graphical analysis to apply the MIP branch-and-bound algorithm presented in Sec. 12.7 to solve this model.
D,I **(d)** Now use the interactive routine for this algorithm in your OR Courseware to solve this model.
C **(e)** Check your answers in parts (*b*), (*c*), and (*d*) by using an automatic routine to solve the model.

D,I **12.7-8.** Use the MIP branch-and-bound algorithm presented in Sec. 12.7 to solve the following MIP problem interactively.

$$\text{Maximize} \quad Z = 5x_1 + 4x_2 + 4x_3 + 2x_4,$$

subject to

$$\begin{aligned} x_1 + 3x_2 + 2x_3 + x_4 &\le 10 \\ 5x_1 + x_2 + 3x_3 + 2x_4 &\le 15 \\ x_1 + x_2 + x_3 + x_4 &\le 6 \end{aligned}$$

and

$$x_j \ge 0, \quad \text{for } j = 1, 2, 3, 4$$
$$x_j \text{ is integer}, \quad \text{for } j = 1, 2, 3.$$

D,I **12.7-9.** Use the MIP branch-and-bound algorithm presented in Sec. 12.7 to solve the following MIP problem interactively.

$$\text{Maximize} \quad Z = 3x_1 + 4x_2 + 2x_3 + x_4 + 2x_5,$$

subject to

$$\begin{aligned} 2x_1 - x_2 + x_3 + x_4 + x_5 &\le 3 \\ -x_1 + 3x_2 + x_3 - x_4 - 2x_5 &\le 2 \\ 2x_1 + x_2 - x_3 + x_4 + 3x_5 &\le 1 \end{aligned}$$

and

$$x_j \ge 0, \quad \text{for } j = 1, 2, 3, 4, 5$$
$$x_j \text{ is binary}, \quad \text{for } j = 1, 2, 3.$$

D,I **12.7-10.** Use the MIP branch-and-bound algorithm presented in Sec. 12.7 to solve the following MIP problem interactively.

$$\text{Minimize} \quad Z = 5x_1 + x_2 + x_3 + 2x_4 + 3x_5,$$

subject to

$$\begin{aligned} x_2 - 5x_3 + x_4 + 2x_5 &\ge -2 \\ 5x_1 - x_2 + x_5 &\ge 7 \\ x_1 + x_2 + 6x_3 + x_4 &\ge 4 \end{aligned}$$

and

$$x_j \ge 0, \quad \text{for } j = 1, 2, 3, 4, 5$$
$$x_j \text{ is integer}, \quad \text{for } j = 1, 2, 3.$$

12.7-11. Reconsider the discrete nonlinear programming problem given in Prob. 12.4-5.

(a) Use the following outline in designing the main features of a branch-and-bound algorithm for solving this problem (and similar problems) directly without reformulation.

(i) Specify the tightest possible nonlinear programming relaxation that has only continuous variables and so can be solved efficiently by nonlinear programming techniques. (The next chapter will describe how such nonlinear programming problems can be solved efficiently.)

(ii) Specify the fathoming tests.

(iii) Specify a branching procedure that involves specifying two ranges of values for a single variable.

(b) Use the algorithm designed in part (*a*) to solve this problem by using an available software package to solve the *quadratic programming* relaxation at each iteration. (As described in Sec. 13.7, Excel, LINDO, LINGO, and MPL/CPLEX all are able to solve quadratic programming problems.)

12.8-1.* For each of the following constraints of pure BIP problems, use the constraint to fix as many variables as possible.

(a) $4x_1 + x_2 + 3x_3 + 2x_4 \le 2$

(b) $4x_1 - x_2 + 3x_3 + 2x_4 \le 2$

(c) $4x_1 - x_2 + 3x_3 + 2x_4 \ge 7$

12.8-2. For each of the following constraints of pure BIP problems, use the constraint to fix as many variables as possible.

(a) $20x_1 - 7x_2 + 5x_3 \le 10$

(b) $10x_1 - 7x_2 + 5x_3 \ge 10$

(c) $10x_1 - 7x_2 + 5x_3 \le -1$

12.8-3. Use the following set of constraints for the *same* pure BIP problem to fix as many variables as possible. Also identify the constraints which become redundant because of the fixed variables.

$$\begin{aligned} 3x_3 - x_5 + x_7 &\le 1 \\ x_2 + x_4 + x_6 &\le 1 \\ x_1 - 2x_5 + 2x_6 &\ge 2 \\ x_1 + x_2 - x_4 &\le 0 \end{aligned}$$

12.8-4. For each of the following constraints of pure BIP problems, identify which ones are made redundant by the binary constraints. Explain why each one is, or is not, redundant.

(a) $2x_1 + x_2 + 2x_3 \le 5$

(b) $3x_1 - 4x_2 + 5x_3 \le 5$

(c) $x_1 + x_2 + x_3 \ge 2$

(d) $3x_1 - x_2 - 2x_3 \ge -4$

12.8-5. In Sec. 12.8, at the end of the subsection on tightening constraints, we indicated that the constraint $4x_1 - 3x_2 + x_3 + 2x_4 \le 5$ can be tightened to $2x_1 - 3x_2 + x_3 + 2x_4 \le 3$ and then to $2x_1 - 2x_2 + x_3 + 2x_4 \le 3$. Apply the procedure for tightening constraints to confirm these results.

12.8-6. Apply the procedure for *tightening constraints* to the following constraint for a pure BIP problem.

$$3x_1 - 2x_2 + x_3 \le 3.$$

12.8-7. Apply the procedure for *tightening constraints* to the following constraint for a pure BIP problem.

$$x_1 - x_2 + 3x_3 + 4x_4 \ge 1.$$

12.8-8. Apply the procedure for *tightening constraints* to each of the following constraints for a pure BIP problem.
(a) $x_1 + 3x_2 - 4x_3 \leq 2$.
(b) $3x_1 - x_2 + 4x_3 \geq 1$.

12.8-9. In Sec. 12.8, a pure BIP example with the constraint, $2x_1 + 3x_2 \leq 4$, was used to illustrate the procedure for tightening constraints. Show that applying the procedure for generating cutting planes to this constraint yields the same new constraint, $x_1 + x_2 \leq 1$.

12.8-10. One of the constraints of a certain pure BIP problem is

$$x_1 + 3x_2 + 2x_3 + 4x_4 \leq 5.$$

Identify all the minimal covers for this constraint, and then give the corresponding cutting planes.

12.8-11. One of the constraints of a certain pure BIP problem is

$$3x_1 + 4x_2 + 2x_3 + 5x_4 \leq 7.$$

Identify all the minimal covers for this constraint, and then give the corresponding cutting planes.

12.8-12. Generate as many cutting planes as possible from the following constraint for a pure BIP problem.

$$3x_1 + 5x_2 + 4x_3 + 8x_4 \leq 10.$$

12.8-13. Generate as many cutting planes as possible from the following constraint for a pure BIP problem.

$$5x_1 + 3x_2 + 7x_3 + 4x_4 + 6x_5 \leq 9.$$

12.8-14. Consider the following BIP problem.

$$\text{Maximize} \quad Z = 2x_1 + 3x_2 + x_3 + 4x_4 + 3x_5 + 2x_6 + 2x_7 + x_8 + 3x_9,$$

subject to

$$\begin{aligned} 3x_2 + x_4 + x_5 &\geq 3 \\ x_1 + x_2 &\leq 1 \\ x_2 + x_4 - x_5 - x_6 &\leq -1 \\ x_2 + 2x_6 + 3x_7 + x_8 + 2x_9 &\geq 4 \\ -x_3 + 2x_5 + x_6 + 2x_7 - 2x_8 + x_9 &\leq 5 \end{aligned}$$

and

$$\text{all } x_j \text{ binary.}$$

Develop the tightest possible formulation of this problem by using the techniques of automatic problem reprocessing (fixing variables, deleting redundant constraints, and tightening constraints). Then use this tightened formulation to determine an optimal solution by inspection.

CASE 12.1 CAPACITY CONCERNS

Bentley Hamilton throws the business section of the *New York Times* onto the conference room table and watches as his associates jolt upright in their overstuffed chairs.

Mr. Hamilton wants to make a point.

He throws the front page of the *Wall Street Journal* on top of the *New York Times* and watches as his associates widen their eyes once heavy with boredom.

Mr. Hamilton wants to make a big point.

He then throws the front page of the *Financial Times* on top of the newspaper pile and watches as his associates dab the fine beads of sweat off their brows.

Mr. Hamilton wants his point indelibly etched into his associates' minds.

"I have just presented you with three leading financial newspapers carrying today's top business story," Mr. Hamilton declares in a tight, angry voice. "My dear associates, our company is going to hell in a hand basket! Shall I read you the headlines? From the *New York Times,* 'CommuniCorp stock drops to lowest in 52 weeks.' From the *Wall Street Journal,* 'CommuniCorp loses 25 percent of the pager market in only one year.' Oh and my favorite, from the *Financial Times,* 'CommuniCorp cannot CommuniCate: CommuniCorp stock drops because of internal communications disarray.' How did our company fall into such dire straits?"

Mr. Hamilton throws a transparency showing a line sloping slightly upward onto the overhead projector. "This is a graph of our productivity over the last 12 months. As you can see from the graph, productivity in our pager production facility has increased steadily over the last year. Clearly, productivity is not the cause of our problem."

Mr. Hamilton throws a second transparency showing a line sloping steeply upward onto the overhead projector. "This is a graph of our missed or late orders over the last 12 months." Mr. Hamilton hears an audible gasp from his associates. "As you can see from the graph, our missed or late orders have increased steadily and significantly over the past 12 months. I think this trend explains why we have been losing market share, causing our stock to drop to its lowest level in 52 weeks. We have angered and lost the business of retailers, our customers who depend upon on-time deliveries to meet the demand of consumers."

"Why have we missed our delivery dates when our productivity level should have allowed us to fill all orders?" Mr. Hamilton asks. "I called several departments to ask this question."

"It turns out that we have been producing pagers for the hell of it!" Mr. Hamilton says in disbelief. "The marketing and sales departments do not communicate with the manufacturing department, so manufacturing executives do not know what pagers to produce to fill orders. The manufacturing executives want to keep the plant running, so they produce pagers regardless of whether the pagers have been ordered. Finished pagers are sent to the warehouse, but marketing and sales executives do not know the number and styles of pagers in the warehouse. They try to communicate with warehouse executives to determine if the pagers in inventory can fill the orders, but they rarely receive answers to their questions."

Mr. Hamilton pauses and looks directly at his associates. "Ladies and gentlemen, it seems to me that we have a serious internal communications problem. I intend to correct this problem immediately. I want to begin by installing a companywide computer network to ensure that all departments have access to critical documents and are able to easily communicate with each other through e-mail. Because this intranet will represent a large change from the current communications infrastructure, I expect some bugs in the system and some resistance from employees. I therefore want to phase in the installation of the intranet."

Mr. Hamilton passes the following timeline and requirements chart to his associates (IN = Intranet).

Month 1	Month 2	Month 3	Month 4	Month 5
IN Education	Install IN in Sales	Install IN in Manufacturing	Install IN in Warehouse	Install IN in Marketing

Department	Number of Employees
Sales	60
Manufacturing	200
Warehouse	30
Marketing	75

Mr. Hamilton proceeds to explain the timeline and requirements chart. "In the first month, I do not want to bring any department onto the intranet; I simply want to disseminate information about it and get buy-in from employees. In the second month, I want to bring the sales department onto the intranet since the sales department receives all critical information from customers. In the third month, I want to bring the manufacturing department onto the intranet. In the fourth month, I want to install the intranet at the warehouse, and in the fifth and final month, I want to bring the marketing department onto the intranet. The requirements chart under the timeline lists the number of employees requiring access to the intranet in each department."

Mr. Hamilton turns to Emily Jones, the head of Corporate Information Management. "I need your help in planning for the installation of the intranet. Specifically, the company needs to purchase servers for the internal network. Employees will connect to company servers and download information to their own desktop computers."

Mr. Hamilton passes Emily the following chart detailing the types of servers available, the number of employees each server supports, and the cost of each server.

Type of Server	Number of Employees Server Supports	Cost of Server
Standard Intel Pentium PC	Up to 30 employees	$ 2,500
Enhanced Intel Pentium PC	Up to 80 employees	$ 5,000
SGI Workstation	Up to 200 employees	$10,000
Sun Workstation	Up to 2,000 employees	$25,000

"Emily, I need you to decide what servers to purchase and when to purchase them to minimize cost and to ensure that the company possesses enough server capacity to follow the intranet implementation timeline," Mr. Hamilton says. "For example, you may decide to buy one large server during the first month to support all employees, or buy several small servers during the first month to support all employees, or buy one small server each month to support each new group of employees gaining access to the intranet."

"There are several factors that complicate your decision," Mr. Hamilton continues. "Two server manufacturers are willing to offer discounts to CommuniCorp. SGI is willing to give you a discount of 10 percent off each server purchased, but only if you purchase servers in the first or second month. Sun is willing to give you a 25 percent discount off all servers purchased in the first two months. You are also limited in the amount of money you can spend during the first month. CommuniCorp has already al-

located much of the budget for the next two months, so you only have a total of $9,500 available to purchase servers in months 1 and 2. Finally, the Manufacturing Department requires at least one of the three more powerful servers. Have your decision on my desk at the end of the week."

(a) Emily first decides to evaluate the number and type of servers to purchase on a month-to-month basis. For each month, formulate an IP model to determine which servers Emily should purchase in that month to minimize costs in that month and support the new users. How many and which types of servers should she purchase in each month? How much is the total cost of the plan?
(b) Emily realizes that she could perhaps achieve savings if she bought a larger server in the initial months to support users in the final months. She therefore decides to evaluate the number and type of servers to purchase over the entire planning period. Formulate an IP model to determine which servers Emily should purchase in which months to minimize total cost and support all new users. How many and which types of servers should she purchase in each month? How much is the total cost of the plan?
(c) Why is the answer using the first method different from that using the second method?
(d) Are there other costs that Emily is not accounting for in her problem formulation? If so, what are they?
(e) What further concerns might the various departments of CommuniCorp have regarding the intranet?

CASE 12.2 ASSIGNING ART

It had been a dream come true for Ash Briggs, a struggling artist living in the San Francisco Bay Area. He had made a trip to the corner grocery store late one Friday afternoon to buy some milk, and on impulse, he had also purchased a California lottery ticket. One week later, he was a millionaire.

Ash did not want to squander his winnings on materialistic, trivial items. Instead he wanted to use his money to support his true passion: art. Ash knew all too well the difficulties of gaining recognition as an artist in this postindustrial, technological society where artistic appreciation is rare and financial support even rarer. He therefore decided to use the money to fund an exhibit of up-and-coming modern artists at the San Francisco Museum of Modern Art.

Ash approached the museum directors with his idea, and the directors became excited immediately after he informed them that he would fund the entire exhibit in addition to donating $1 million to the museum. Celeste McKenzie, a museum director, was assigned to work with Ash in planning the exhibit. The exhibit was slated to open one year from the time Ash met with the directors, and the exhibit pieces would remain on display for two months.

Ash began the project by combing the modern art community for potential artists and pieces. He presented the following list of artists, their pieces, and the price of displaying each piece[1] to Celeste.

Artist	Piece	Description of Piece	Price
Colin Zweibell	"Perfection"	A wire mesh sculpture of the human body	$300,000
	"Burden"	A wire mesh sculpture of a mule	$250,000
	"The Great Equalizer"	A wire mesh sculpture of a gun	$125,000
Rita Losky	"Chaos Reigns"	A series of computer-generated drawings	$400,000
	"Who Has Control?"	A computer-generated drawing intermeshed with lines of computer code	$500,000
	"Domestication"	A pen-and-ink drawing of a house	$400,000
	"Innocence"	A pen-and-ink drawing of a child	$550,000
Norm Marson	"Aging Earth"	A sculpture of trash covering a larger globe	$700,000
	"Wasted Resources"	A collage of various packaging materials	$575,000
Candy Tate	"Serenity"	An all blue watercolor painting	$200,000
	"Calm Before the Storm"	A painting with an all blue watercolor background and a black watercolor center	$225,000
Robert Bayer	"Void"	An all black oil painting	$150,000
	"Sun"	An all yellow oil painting	$150,000
David Lyman	"Storefront Window"	A photo-realistic painting of a jewelry store display window	$850,000
	"Harley"	A photo-realistic painting of a Harley-Davidson motorcycle	$750,000
Angie Oldman	"Consumerism"	A collage of magazine advertisements	$400,000
	"Reflection"	A mirror (considered a sculpture)	$175,000
	"Trojan Victory"	A wooden sculpture of a condom	$450,000

[1]The display price includes the cost of paying the artist for loaning the piece to the museum, transporting the piece to San Francisco, constructing the display for the piece, insuring the piece while it is on display, and transporting the piece back to its origin.

Artist	Piece	Description of Piece	Price
Rick Rawls	"Rick"	A photo-realistic self-portrait (painting)	$500,000
	"Rick II"	A cubist self-portrait (painting)	$500,000
	"Rick III"	An expressionist self-portrait (painting)	$500,000
Bill Reynolds	"Beyond"	A science fiction oil painting depicting Mars colonization	$650,000
	"Pioneers"	An oil painting of three astronauts aboard the space shuttle	$650,000
Bear Canton	"Wisdom"	A pen-and-ink drawing of an Apache chieftain	$250,000
	"Superior Powers"	A pen-and-ink drawing of a traditional Native American rain dance	$350,000
	"Living Land"	An oil painting of the Grand Canyon	$450,000
Helen Row	"Study of a Violin"	A cubist painting of a violin	$400,000
	"Study of a Fruit Bowl"	A cubist painting of a bowl of fruit	$400,000
Ziggy Lite	"My Namesake"	A collage of Ziggy cartoons	$300,000
	"Narcissism"	A collage of photographs of Ziggy Lite	$300,000
Ash Briggs	"All That Glitters"	A watercolor painting of the Golden Gate Bridge	$50,000*
	"The Rock"	A watercolor painting of Alcatraz	$ 50,000
	"Winding Road"	A watercolor painting of Lombard Street	$ 50,000
	"Dreams Come True"	A watercolor painting of the San Francisco Museum of Modern Art	$ 50,000

*Ash does not require personal compensation, and the cost for moving his pieces to the museum from his home in San Francisco is minimal. The cost of displaying his pieces therefore only includes the cost of constructing the display and insuring the pieces.

Ash possesses certain requirements for the exhibit. He believes the majority of Americans lack adequate knowledge of art and artistic styles, and he wants the exhibit to educate Americans. Ash wants visitors to become aware of the collage as an art form, but he believes collages require little talent. He therefore decides to include only one collage. Additionally, Ash wants viewers to compare the delicate lines in a three-dimensional wire mesh sculpture to the delicate lines in a two-dimensional computer-

generated drawing. He therefore wants at least one wire mesh sculpture displayed if a computer-generated drawing is displayed. Alternatively, he wants at least one computer-generated drawing displayed if a wire mesh sculpture is displayed. Furthermore, Ash wants to expose viewers to all painting styles, but he wants to limit the number of paintings displayed to achieve a balance in the exhibit between paintings and other art forms. He therefore decides to include at least one photo-realistic painting, at least one cubist painting, at least one expressionist painting, at least one watercolor painting, and at least one oil painting. At the same time, he wants the number of paintings to be no greater than twice the number of other art forms.

Ash wants all his own paintings included in the exhibit since he is sponsoring the exhibit and since his paintings celebrate the San Francisco Bay Area, the home of the exhibit.

Ash possesses personal biases for and against some artists. Ash is currently having a steamy affair with Candy Tate, and he wants both of her paintings displayed. Ash counts both David Lyman and Rick Rawls as his best friends, and he does not want to play favorites among these two artists. He therefore decides to display as many pieces from David Lyman as from Rick Rawls and to display at least one piece from each of them. Although Ziggy Lite is very popular within art circles, Ash believes Ziggy makes a mockery of art. Ash will therefore only accept one display piece from Ziggy, if any at all.

Celeste also possesses her own agenda for the exhibit. As a museum director, she is interested in representing a diverse population of artists, appealing to a wide audience, and creating a politically correct exhibit. To advance feminism, she decides to include at least one piece from a female artist for every two pieces included from a male artist. To advance environmentalism, she decides to include either one or both of the pieces "Aging Earth" and "Wasted Resources." To advance Native American rights, she decides to include at least one piece by Bear Canton. To advance science, she decides to include at least one of the following pieces: "Chaos Reigns," "Who Has Control," "Beyond," and "Pioneers."

Celeste also understands that space is limited at the museum. The museum only has enough floor space for four sculptures and enough wall space for 20 paintings, collages, and drawings.

Finally, Celeste decides that if "Narcissism" is displayed, "Reflection" should also be displayed since "Reflection" also suggests narcissism.

Please explore the following questions independently except where otherwise indicated.

(a) Ash decides to allocate $4 million to fund the exhibit. Given the pieces available and the specific requirements from Ash and Celeste, formulate and solve a BIP model to maximize the number of pieces displayed in the exhibit without exceeding the budget. How many pieces are displayed? Which pieces are displayed?

(b) To ensure that the exhibit draws the attention of the public, Celeste decides that it must include at least 20 pieces. Formulate and solve a BIP model to minimize the cost of the exhibit while displaying at least 20 pieces and meeting the requirements set by Ash and Celeste. How much does the exhibit cost? Which pieces are displayed?

(c) An influential patron of Rita Losky's work who chairs the Museum Board of Directors learns that Celeste requires at least 20 pieces in the exhibit. He offers to pay the minimum amount required on top of Ash's $4 million to ensure that exactly 20 pieces are displayed in the exhibit and that all of Rita's pieces are displayed. How much does the patron have to pay? Which pieces are displayed?

CASE 12.3 STOCKING SETS

Daniel Holbrook, an expeditor at the local warehouse for Furniture City, sighed as he moved boxes and boxes of inventory to the side in order to reach the shelf where the particular item he needed was located. He dropped to his hands and knees and squinted at the inventory numbers lining the bottom row of the shelf. He did not find the number he needed. He worked his way up the shelf until he found the number matching the number on the order slip. Just his luck! The item was on the top row of the shelf! Daniel walked back through the warehouse to find a ladder, stumbling over boxes of inventory littering his path. When he finally climbed the ladder to reach the top shelf, his face crinkled in frustration. Not again! The item he needed was not in stock! All he saw above the inventory number was an empty space covered with dust!

Daniel trudged back through the warehouse to make the dreadful phone call. He dialed the number of Brenda Sims, the saleswoman on the kitchen showroom floor of Furniture City, and informed her that the particular light fixture the customer had requested was not in stock. He then asked her if she wanted him to look for the rest of the items in the kitchen set. Brenda told him that she would talk to the customer and call him back.

Brenda hung up the phone and frowned. Mr. Davidson, her customer, would not be happy. Ordering and receiving the correct light fixture from the regional warehouse would take at least two weeks.

Brenda then paused to reflect upon business during the last month and realized that over 80 percent of the orders for kitchen sets could not be filled because items needed to complete the sets were not in stock at the local warehouse. She also realized that Furniture City was losing customer goodwill and business because of stockouts. The furniture megastore was gaining a reputation for slow service and delayed deliveries, causing customers to turn to small competitors that sold furniture directly from the showroom floor.

Brenda decided to investigate the inventory situation at the local warehouse. She walked the short distance to the building next door and gasped when she stepped inside the warehouse. What she saw could only be described as chaos. Spaces allocated for some items were overflowing into the aisles of the warehouse while other spaces were completely bare. She walked over to one of the spaces overflowing with inventory to discover the item that was overstocked. She could not believe her eyes! The warehouse had at least 30 rolls of pea-green wallpaper! No customer had ordered pea-green wallpaper since 1973!

Brenda marched over to Daniel demanding an explanation. Daniel said that the warehouse had been in such a chaotic state since his arrival one year ago. He said the inventory problems occurred because management had a policy of stocking every furniture item on the showroom floor in the local warehouse. Management only replenished inventory every three months, and when inventory was replenished, management ordered every item regardless of if it had been sold. Daniel also said that he had tried to make management aware of the problems with overstocking unpopular items and understocking popular items, but that management would not listen to him because he was simply an expeditor.

Brenda understood that Furniture City required a new inventory policy. Not only was the megastore losing money by making customers unhappy with delivery delays, but it was also losing money by wasting warehouse space. By changing the inventory policy to stock only popular items and replenish them immediately when they are sold, Furniture City would ensure that the majority of customers receive their furniture immediately and that the valuable warehouse space was utilized effectively.

Brenda needed to sell her inventory policy to management. Using her extensive sales experience, she decided that the most effective sales strategy would be to use her kitchen department as a model for the new inventory policy. She would identify all kitchen sets comprising 85 percent of customers orders. Given the fixed amount of warehouse space allocated to the kitchen department, she would identify the items Furniture City should stock in order to satisfy the greatest number of customer orders. She would then calculate the revenue from satisfying customer orders under the new inventory policy, using the bottom line to persuade management to accept her policy.

Brenda analyzed her records over the past three years and determined that 20 kitchen sets were responsible for 85 percent of the customer orders. These 20 kitchen sets were composed of up to eight features in a variety of styles. Brenda listed each feature and its popular styles:

Floor Tile	Wallpaper	Light Fixtures	Cabinets
(T1) White textured tile	(W1) Plain ivory paper	(L1) One large rectangular frosted fixture	(C1) Light solid wood cabinets
(T2) Ivory textured tile	(W2) Ivory paper with dark brown pinstripes	(L2) Three small square frosted fixtures	(C2) Dark solid wood cabinets
(T3) White checkered tile with blue trim	(W3) Blue paper with marble texture	(L3) One large oval frosted fixture	(C3) Light wood cabinets with glass doors
(T4) White checkered tile with light yellow trim	(W4) Light yellow paper with marble texture	(L4) Three small frosted globe fixtures	(C4) Dark wood cabinets with glass doors

Countertops	Dishwashers	Sinks	Ranges
(O1) Plain light wood countertops	(D1) White energy-saving dishwasher	(S1) Sink with separate hot and cold water taps	(R1) White electric oven
(O2) Stained light wood countertops	(D2) Ivory energy-saving dishwasher	(S2) Divided sink with separate hot and cold water taps and garbage disposal	(R2) Ivory electric oven
(O3) White lacquer-coated countertops		(S3) Sink with one hot and cold water tap	(R3) White gas oven
(O4) Ivory lacquer-coated countertops		(S4) Divided sink with one hot and cold water tap and garbage disposal	(R4) Ivory gas oven

Brenda then created a table showing the 20 kitchen sets and the particular features composing each set. To simplify the table, she used the codes shown in parentheses above to represent the particular feature and style. The table is given below. For example, kitchen set 1 consists of floor tile T2, wallpaper W2, light fixture L4, cabinet C2, countertop O2, dishwasher D2, sink S2, and range R2. Notice that sets 14 through 20 do not contain dishwashers.

Brenda knew she had only a limited amount of warehouse space allocated to the kitchen department. The warehouse could hold 50 square feet of tile and 12 rolls of wallpaper in the inventory bins. The inventory shelves could hold two light fixtures, two cabinets, three countertops, and two sinks. Dishwashers and ranges are similar in size, so Furniture City stored them in similar locations. The warehouse floor could hold a total of four dishwashers and ranges.

Every kitchen set always includes exactly 20 square feet of tile and exactly five rolls of wallpaper. Therefore, 20 square feet of a particular style of tile and five rolls of a particular style of wallpaper are required for the styles to be in stock.

(a) Formulate and solve a BIP model to maximize the total number of kitchen sets (and thus the number of customer orders) Furniture City stocks in the local warehouse. Assume that when a customer orders a kitchen set, all the particular items composing that kitchen set are replenished at the local warehouse immediately.

(b) How many of each feature and style should Furniture City stock in the local warehouse? How many different kitchen sets are in stock?

(c) Furniture City decides to discontinue carrying nursery sets, and the warehouse space previously allocated to the nursery department is divided between the existing departments at Furniture City. The kitchen department receives enough additional space to allow it to stock both styles of dishwashers and three of the four styles of ranges. How does the optimal inventory policy for the kitchen department change with this additional warehouse space?

(d) Brenda convinces management that the kitchen department should serve as a testing ground for future inventory policies. To provide adequate space for testing, management decides to allocate all the space freed by the nursery department to the kitchen department. The extra

	T1	T2	T3	T4	W1	W2	W3	W4	L1	L2	L3	L4	C1	C2	C3	C4	O1	O2	O3	O4	D1	D2	S1	S2	S3	S4	R1	R2	R3	R4
Set 1		X				X						X		X						X		X		X				X		
Set 2		X			X				X							X				X		X				X	X	X		
Set 3	X						X			X			X				X				X				X				X	
Set 4			X				X				X				X				X		X		X				X			
Set 5				X				X	X				X					X				X				X		X		
Set 6		X				X				X						X				X		X		X						X
Set 7	X						X					X			X			X			X		X				X			
Set 8		X			X						X		X				X					X			X				X	
Set 9		X			X					X					X			X				X		X				X		
Set 10	X				X				X				X				X				X					X			X	
Set 11			X		X						X				X		X				X		X						X	
Set 12		X				X			X					X				X				X				X		X		
Set 13				X				X			X				X		X				X			X					X	
Set 14				X				X				X	X						X				X				X			
Set 15			X				X		X				X				X								X				X	
Set 16			X				X					X	X						X					X			X			
Set 17	X							X		X					X				X						X				X	
Set 18		X					X				X			X						X			X				X			
Set 19		X						X				X				X				X				X						X
Set 20		X					X		X				X				X								X					X

space means that the kitchen department can store not only the dishwashers and ranges from part (*c*), but also all sinks, all countertops, three of the four light fixtures, and three of the four cabinets. How much does the additional space help?

(e) How would the inventory policy be affected if the items composing a kitchen set could not be replenished immediately? Under what conditions is the assumption of immediate replenishment nevertheless justified?

CASE 12.4 ASSIGNING STUDENTS TO SCHOOLS (REVISITED AGAIN)

Reconsider Case 4.3.

The Springfield School Board now has made the decision to prohibit the splitting of residential areas among multiple schools. Thus, each of the six areas must be assigned to a single school.

(a) Formulate a BIP model for this problem under the current policy of providing bussing for all middle school students who must travel more than approximately a mile.

(b) Referring to part (*a*) of Case 4.3, explain why that linear programming model and the BIP model just formulated are so different when they are dealing with nearly the same problem.

(c) Solve the BIP model formulated in part (*a*).

(d) Referring to part (*c*) of Case 4.3, determine how much the total bussing cost increases because of the decision to prohibit the splitting of residential areas among multiple schools.

(e, f, g, h) Repeat parts (*e, f, g, h*) of Case 4.3 under the new school board decision to prohibit splitting residential areas among multiple schools.