Contents

Chapter 7

Modeling Integer and Combinatorial Programs

This is Chapter 7 of "Junior Level Web-Book *Optimization Models for decision Making*" by Katta G. Murty.

7.1 Types of Integer Programs, an Example Puzzle Problem, and a Classical Solution Method

So far, we considered continuous variable optimization models. In this chapter we will discuss modeling **discrete** or **mixed discrete optimization problems** in which all or some of the decision variables are restricted to assume values within specified discrete sets, and **combinatorial optimization problems** in which an optimum combination/arrangement out of a possible set of combinations/arrangements has to be determined. Many of these problems can be modeled as LPs with additional integer restrictions on some, or all, of the variables. LP models with additional integer restrictions on decision variables are called **integer linear programming problems** or just **integer programs**. They can be classified into the following types.

Pure (or, all) integer programs: These are integer programs in which all the decision variables are restricted to assume only integer values.

0−1 pure integer programs: These are pure integer programs, and in addition, all decision variables are bounded variables with lower bound 0, and upper bound 1; i.e., in effect, every decision variable in them is required to be either 0 or 1.

Mixed integer programs or MIPs: Integer programs in which there are some continuous decision variables and some integer decision variables.

0−1 mixed integer programs (0-1 MIPs): These are MIPs in which all the integer decision variables are 0−1 variables.

Integer feasibility problems: Mathematical models in which it is required to find an integer solution to a given system of linear constraints, without any optimization.

0−1 integer feasibility problems: Integer feasibility problems to find a 0−1 solution to a given system of linear constraints.

Many problems involve various **yes - no decisions**, which can be considered as the 0−1 values of integer variables so constrained. Variables which are restricted to the values 0 or 1 are called **0−1 variables** or **binary variables** or **boolean variables**. That's why 0−1 integer programs are also called **binary (or boolean) variable optimization problems**.

And in many practical problems, activities and resources (like machines, ships, and operators) are indivisible, leading to integer decision variables in models involving them.

Many puzzles, riddles, and diversions in recreational mathematics and mathematical games are combinatorial problems that can be formulated as integer programs, or plain integer feasibility problems. We now provide a 0−1 integer feasibility formulation for a problem discussed in the superbly entertaining book [R. M. Smullyan, 1978].

Example 7.1.1: **An Integer program in the play *Merchant of Venice* by William Shakespeare**

The setting of this problem from [R. M. Smullyan, 1978] is William Shakespeare's play "The Merchant of Venice" written in the year 1597. In this play, a girl named Portia is the lead female character. She was a law graduate with an obsession for highly intelligent boys. Her sole concern was with "intelligence", completely ignoring other characteristics that girls usually associate with desirable life-mates. Her life's ambition was to marry an extremely intelligent boy, no matter how he looks or behaves, or how wealthy he is. For achieving this goal she devised a very clever scheme to choose her fiance.

She purchased three caskets, one of gold, silver, and lead, and hid a stunningly beautiful portrait of herself in one of them. The suitor was asked to identify the casket containing the portrait. If his choice is correct, he can claim Portia as his bride; otherwise he will be permanently banished to guarantee that he won't appear for the test again. To help the suitor choose intelligently, Portia put inscriptions on the caskets as in Figure 7.1. And she explained that at most one of the three inscriptions was true. She reasoned that only an intelligent boy could identify the casket containing the portrait with these clues.

The portrait is in this casket	The portrait is not in this casket	The portrait is not in the gold casket
1 = Gold	2 = Silver	3 = Lead

Figure 7.1

We will now show that the problem of identifying the casket containing the portrait, can be modeled as a 0−1 integer feasibility problem. The model uses 0−1 variables known as **combinatorial choice variables** corresponding to the various possibilities in this problem, defined above. These decision variables are, for $j = 1, 2, 3$,

$$x_j = \begin{cases} 1, & \text{if the } j\text{th casket contains the portrait} \\ 0, & \text{otherwise} \end{cases}$$

$$y_j = \begin{cases} 1, & \text{if the inscription on the } j\text{th casket is true} \\ 0, & \text{otherwise} \end{cases} \tag{7.1.1}$$

These variables have to satisfy the following constraints

$$\begin{array}{rrrrrrcl} x_1 & +x_2 & +x_3 & & & & = & 1 \\ -x_1 & & & +y_1 & & & = & 0 \\ & x_2 & & & +y_2 & & = & 1 \\ x_1 & & & & & +y_3 & = & 1 \\ & & & y_1 & +y_2 & +y_3 & \leq & 1 \end{array} \tag{7.1.2}$$

$$x_j, y_j \quad = \quad 0 \quad \text{or} \quad 1 \quad \text{for all } j$$

The first constraint in (7.1.2) must hold because only one casket contains the portrait. The second, third, fourth constraints must hold because of the inscriptions on caskets 1, 2, 3, and the definitions of the variables x_1, y_1; x_2, y_2; x_3, y_3. The fifth constraint must hold because at most one inscription was true.

In (7.1.2), the $0-1$ values for each variable denote the two distinct possibilities associated with that variable. Fractional values for any of the variables in (7.1.2) do not represent anything in the model, and hence do not make any sense for the problem. Also, given the definitions of the variables in (7.1.1), we cannot claim that a fractional value like 0.99 for one of these variables is closer to 1 than 0 for this variable. Due to this, we cannot take a fractional solution to the system consisting of the top 5 constraints in (7.1.2), and somehow try to round it to satisfy the integer requirements on the values of the variables also. The usual technique of rounding a fractional solution to a nearest integer point does not make any sense at all when dealing with integer models involving combinatorial choice variables like those in this problem.

Since there is no objective function to be optimized in (7.1.2), it is a 0−1 integer feasibility problem. It is a formulation for Portia's casket problem involving 0−1 integer variables.

A Classical Solution Method, Total Enumeration

In a pure 0−1 problem with three 0−1 variables $x = (x_1, x_2, x_3)$, the only vectors that could be solution vectors are (0, 0, 0), (0, 0, 1), (0,1, 0), (0,1,1), (1,0,0), (1,0,1), (1,1,0), (1, 1, 1). So, by checking each of these $2^3 = 8$ vectors we can identify the set of all feasible solutions, and also the set of optimum solutions for the problem.

In the same way, an optimum solution of a 0−1 problem in binary variables $x = (x_1, \ldots, x_n)$ can be found by checking the set of all 0−1 vectors x which number 2^n. Since the spirit of this method is to check all possible vectors for the optimum solution, this classical method is called the **total enumeration method**.

In the same way, when dealing with integer variables (or variables that can take only values in discrete sets), the set of all vectors that could be solutions is a discrete set which can be enumerated one by one to check and identify the best solution. This is the total enumeration method. The name of the method refers to the fact that the method examines every possible solution and selects the best feasible solution among them.

In a pure integer program there are only a finite number of solutions if all the variables have finite lower and upper bounds specified for them. A solution is obtained by giving each variable an integer value within its bounds. This solution is a feasible solution to the problem if it satisfies all the other equality and inequality constraints in the system; and if it is feasible, we evaluate the objective function at it. By examining each possible integer solution this way, and then comparing the objective values at the feasible solutions among them, we can find an optimum feasible solution if the problem has a feasible solution.

If there is no upper or lower bound on the value of one or more variables in a pure integer program, the number of solutions to examine in the above method could be infinite. But still they could be evaluated one by one as discussed above.

In MIPs there are some integer and some continuous decision variables. The total enumeration method uses the following enumeration scheme to solve an MIP. First, all the integer variables are given specific integer values within their bounds. The remaining problem, consisting of only the continuous decision variables, is a linear program, and can be solved by the methods discussed earlier. If this LP is feasible, this yields the best possible feasible solution for the original problem with all the integer variables fixed at their current values. Repeat this process with each possible choice of integer values for the integer variables, and select the best among all the feasible solutions obtained as the optimum solution for the original problem.

Solution of Portia's problem by total enumeration: As an example, we will illustrate how the total enumeration method solves Portia's casket problem. The only possible choices for the vector $x = (x_1, x_2, x_3)^T$ to satisfy the first and sixth constraints in (7.2) are $(1, 0, 0)^T$, $(0, 1, 0)^T$, and $(0, 0, 1)^T$. We try each of these choices and see whether we can generate a vector $y = (y_1, y_2, y_3)^T$ which together with this x satisfies the remaining constraints in (7.2).

If $x = (1, 0, 0)^T$, by the second and third constraints in (7.1.2), we get $y_1 = 1$, $y_2 = 1$, and the fifth constraint will be violated. So, $x = (1, 0, 0)^T$ cannot lead to a feasible solution to (7.1.2).

If $x = (0, 1, 0)^T$, from the second, third, and fourth constraints in (7.1.2), we get $y_1 = 0, y_2 = 0$, and $y_3 = 1$, and we verify that $x = (0, 1, 0)^T$, $y = (0, 0, 1)^T$ satisfies all the constraints in the problem, hence it is a feasible solution to (7.1.2).

In the same way we verify that $x = (0, 0, 1)^T$ does not lead to a feasible solution to (7.1.2).

Hence the unique solution of (7.1.2) is $x = (0, 1, 0)^T$, $y = (0, 0, 1)^T$; which by the definition of the variables implies that casket 2 (silver casket) must contain Portia's portrait.

Thus total enumeration involves checking every possibility. It is an extremely simple method, and it works well if the number of possibilities to be examined is small. In fact, this is the method used to solve problems of Category 1 discussed in Chapter 1, 2. Unfortunately, in real world applications of integer programming and combinatorial optimization, the number of possibilities to check tends to be so huge, that even using the fastest and most sophisticated computers available today, the answer to the problem cannot be obtained by total enumeration within the lifetime of the decision maker, making it impractical.

Thus for solving large scale problems of Category 2 discussed in Chapter 1, total enumeration is not a practical approach. We need more efficient algorithms to handle these problems. An improved approach based on partial enumeration is presented in Chapter 8.

Conclusion of Portia's Story

As R. Smullyan reports in his 1978 book, an intelligent, nice, and handsome suitor showed up for Portia's test. He chose correctly and claimed Portia's hand in marriage, and they lived happily for a while. The sequel to this story is stated in the following Exercise 7.1.1.

Exercises: Some puzzle problems to model as integer programs: Below are some puzzle problems that can be modeled using combinatorial choice variables by direct arguments. Somewhat more difficult puzzle problems that need the use of the modeling tool called *indicator variables* or given later.

7.1.1: This exercise is also adapted from the excellent 1978 book of R. Smullyan. As

The portrait is not in the silver casket	The portrait is not in this casket	The portrait is in this casket
1 = Gold	2 = Silver	3 = Lead

Figure 7.2

discussed under the marriage problem in Chapter 3, familiarity breeds contempt, and after a brief blissful period of married life, Portia was haunted by the following thought: "My husband displayed intelligence by solving my casket problem correctly, but that problem was quite easy. I could have posed a much harder problem, and gotten a more intelligent husband." Because of her craving for intelligence, she could not continue her married life with this thought, and being a lawyer, she was able to secure a divorce easily. This time she wanted to find a more intelligent husband by the casket method again, and had the inscriptions put on the caskets as shown in Figure 7.2.

She explained to the suitors that at least one of the three inscriptions is true, and at least one of them is false.

Formulate the problem of identifying the casket containing the portrait in this situation as a $0-1$ integer feasibility problem, and solve it by total enumeration.

P.S. To complete the story, the first man who solved this casket problem turned out to be Portia's ex-husband. So, they got married again. He took her home, and being not only intelligent but also worldly-wise he was able to convince her that he is the right man for her, and they lived happily ever after.

7.1.2: Four persons, one of whom has committed a terrible crime, made the following statements when questioned by the police. Anita: "Kitty did it." Kitty: "Robin did it." Ved: "I didn't do it." Robin: "Kitty lied."

If only one of these four statements is true, formulate the problem of finding the guilty person as a $0-1$ feasibility problem, and find its solution by total enumeration.

Who is the guilty person if only one of the four statements is false? Formulate this as a $0-1$ feasibility problem and solve it.

7.1.3: Lady or Tigers-1: (Adopted from R. Smullyan, 1982) In trying to win the hand of his beloved, a man becomes a prisoner, and is faced with a decision where he must open the door of one of four rooms. Each room may be either empty, or hiding a tiger, or his beloved. Each door has a sign bearing a statement that may be true or false. The statements on the four doors are:

Door 1: The lady is in an odd-numbered room

Door 2: This room is empty

Door 3: The sign on door 1 is false

Door 4: The lady is not in room 1

The prisoner is given several clues: He is informed that the lady is in one of the rooms. The sign on the door hiding the lady is true, the signs on all doors hiding tigers are false, and the signs on the doors of empty rooms can be either true or false. Also, either the sign on door 2 is right, or that on door 4 is wrong.

If the prisoner opens the door to find the lady, he can marry her; if he opens the door to find a tiger he will be eaten alive; and if he opens an empty room, he will be banished alone to a distant foreign land.

Help the prisoner by formulating the problem of finding all the doors which may be hiding his beloved, subject to all the clues.

7.1.4: Color of the mule: (From R. Smullyan, 1997)

The setting of this puzzle is the "Tales of the Arabian Nights" with Scheherazade entertaining her husband, the King, with some puzzles, so he will stay her execution until next day. This is the puzzle she told him on the 1003rd night.

"A certain Sheik named Hassan once met three boys and told them about his mule.

"What color is he?" asked one boy. "Well he is either brown, black, or gray. Make some guesses."

"He is black" one boy said. "He is either brown or gray" said another. "He is brown" the third said.

Hassan said "at least one of you guessed right, and at least one of you guessed wrong".

Formulate the problem of finding the mule's color as an integer feasibility problem, and solve it using total enumeration.

7.1.5: (From R. Smullyan, 1997)

Abdul has a shop which was robbed, but the loot was recovered. A, I and H were the three suspects. They made the following statements. These may be true or lies.

A: I did not commit the robbery.

I: H certainly didn't.

H: Yes, I did.

Later two of them confessed to having lied, and this is correct. It is reqiured to find who committed the robbery. Formulate this as an integer feasibility problem, and solve it using total enumeration. ⋈

In the following sections we show how a variety of combinatorial conditions arising in practical applications can be modeled through linear constraints involving $0-1$ integer variables. And we present several integer programming models that appear often in applications.

7.2 The Knapsack Problems

Knapsack problems (or **one dimensional knapsack problems** to be specific, see later on) are single constraint pure integer programs. The knapsack model refers to the following situation. Articles of n different types are available. Each article of type i has weight w_i kg. and value $\$v_i$. A knapsack that can hold a weight of at most w kg. is available for loading as many of these objects as will fit into it, so as to maximize the value of the articles included subject to the knapsack's weight capacity. Articles cannot be broken, only a nonnegative integer number of articles of each type can be loaded. For $j = 1$ to n define

$$x_j \quad = \quad \text{number of articles of type } j \text{ included in the knapsack}$$

In terms of these decision variables, the problem is

$$\begin{aligned} \text{Maximize} \quad z(x) = \sum_{j=1}^{n} v_j x_j & \\ \text{subject to} \quad \sum_{j=1}^{n} w_j x_j \ \leq \ w & \qquad (7.2.1) \\ x_j \geq 0 \quad \text{and integer} \qquad \text{for all} \quad j & \end{aligned}$$

This is known as the **nonnegative integer knapsack problem**. It is characterized by a single inequality constraint of "$\leq$" type, and all positive integral data elements.

If the last condition in (7.2.1) is replaced by "$x_j = 0$ or 1 for all j", the problem becomes the **0−1 knapsack problem**.

The knapsack problem is the simplest integer program, but it has many applications in capital budgeting, project selection, etc. It also appears as a subproblem in algorithms for cutting stock problems and other integer programming algorithms.

Example 7.2.1: Journal Subscription Problem

We now present an application for the knapsack model to a problem that arose at the University of Michigan Engineering Library in Ann Arbor. At that time the library was subscribing to 1200 serial journals, and the annual subscription budget was about $300,000. The unending battle to balance the serials budget, and the exorbitant price increases for subscriptions to scholarly journals, have made it essential for the library to consider a reduction in acquisitions. This led to the problem of determining which subscriptions to renew and which to cancel, in order to bring the total serials subscription expenditure to within the specified budget. To protect the library's traditional strengths as a research facility, the librarian has set the goal of making these renewal/cancellation decisions in order to provide the greatest number of patrons the most convenient access to the serial literature they require within allotted budget. Anticipating this problem, the library staff has been gathering data on the use of journals for about four years. We constructed a sample problem to illustrate this application, using the data from 8 different journals. The value or the readership of a journal given in the following table is the average number of uses per year per title.

Suppose the total budget available for subscriptions to these 8 journals is $670. Defining for $j = 1$ to 8

$$x_j = \begin{cases} 1, & \text{if subscription to journal } j \text{ is renewed} \\ 0, & \text{otherwise} \end{cases}$$

Journal j	Subscription \$/year, w_j	Readership, v_j
1	80	7840
2	95	6175
3	115	8510
4	165	15015
5	125	7375
6	78	1794
7	69	897
8	99	8316

we get the following 0−1 integer programming formulation for the library's problem of determining which journal subscriptions to renew, to maximize readership subject to the budget constraint.

$$\begin{aligned} \text{Max.} \quad z(x) \quad = \quad & 7840x_1 + 6175x_2 + 8510x_3 + 15015x_4 + 7375x_5 + 1794x_6 + 897x_7 \\ & +8316x_8 \\ \text{s. to} \quad & 80x_1 + 95x_2 + 115x_3 + 165x_4 + 125x_5 + 78x_6 + 69x_7 + 99x_8 \le 670 \\ & x_j = 0 \text{ or } 1 \text{ for all } j \end{aligned}$$

Clearly, this problem is a 0−1 knapsack problem.

The Multidimensional Knapsack Problem

Consider the knapsack problem involving n articles. Suppose we are given the value of each article, and both its weight as well as volume. And assume that the knapsack has a capacity on both the weight and the volume that it can hold. Then the problem of determining the optimum number of articles of each type to load into the knapsack, to maximize the total value loaded subject to both the weight and volume constraints, is a problem of the form (7.3) with two constraints instead of one. A problem of this form is in called a **multidimensional knapsack problem**. A general multidimensional knapsack problem is the following problem

$$\begin{array}{rll} \text{Maximize} \quad z(x) = cx & & \\ \text{subject to} \quad Ax & \leq & b \\ x \geq 0 & & \text{and integer} \end{array} \tag{7.2.2}$$

where A is an $m \times n$ matrix, and A, b, c are all > 0 and integer.

0−1 Multidimensional Knapsack Problem with Additional Multiple Choice Constraints

Consider a multidimensional knapsack problem involving n articles, in which at most one copy of each article is available for packing into the knapsack. So, the decision variables in this problem are, for $j = 1$ to n

$$x_j = \begin{cases} 1, & \text{if } j\text{th article is packed into the knapsack} \\ 0, & \text{otherwise} \end{cases}$$

Let c_j be the value of article j, so $z(x) = \sum c_j x_j$ is the objective function to be maximized in this problem. Let $Ax \leq b$ be the system of m multidimensional knapsack constraints in this problem.

In addition, suppose the n articles are partitioned into p disjoint subsets $\{1, \ldots, n_1\}, \{n_1+1, \ldots, n_1+n_2\}, \ldots, \{n_1+\ldots+n_{p-1}+1, \ldots, n_1+\ldots+n_p\}$ consisting of n_1, n_2, …, n_p articles respectively, where $n_1+\ldots+n_p = n$, and it is specified that precisely one article must be selected from each of these subsets. These additional requirements impose the following constraints

$$\begin{array}{llllr} x_1 + \ldots + x_{n_1} & & & & = 1 \\ & x_{n_1+1} + \ldots + x_{n_1+n_2} & & & = 1 \\ & & \ddots & & \vdots \\ & & & x_{n_1+\ldots+n_{p-1}+1} + \ldots + x_n & = 1 \end{array}$$

A system of equality constraints of this type in 0−1 variables is called a system of **multiple choice constraints**. Each constraint among these specifies that a single variable among a subset of variables has to be set equal to 1, while all the other variables in that subset are set equal to 0. The combined problem is the following

$$
\begin{array}{lllll}
\text{Max.} & z(x) = cx & & & \\
\text{s. to} & Ax \leq b & & & \\
 & & & & \\
x_1 + \ldots + x_{n_1} & & & & = 1 \\
 & x_{n_1+1} + \ldots + x_{n_1+n_2} & & & = 1 \\
 & & \ddots & & \vdots \\
 & & & x_{n_1+\ldots+n_{p-1}+1} + \ldots + x_n & = 1 \\
 & x_j = 0 \text{ or } 1 \text{ for all } j & & &
\end{array}
$$

This is the general 0−1 multidimensional knapsack problem with additional multiple choice constraints.

Exercises

7.2.1: Girlscout fruit problem: (Lisa Schaefer) A school girl is raising money for Girlscouts by selling fruit in the neighborhood of her school. She picks the fruit at her school in a knapsack, and carries it around door to door in the neighborhood trying to sell. Her carrying capacity is 25 lbs of fruit, and wants to earn the maximum possible amount in each round. The following fruit is available, we also provide the selling price/piece. It is required to determine the highest total value that can be packed in the knapsack subject to the 25 lbs weight limit. Formulate this problem.

Fruit	Per piece		Quantity
	Weight(lbs)	Selling price ($)	available
Cantalopue	3	2	4
Watermelon	6	8	3
Honeydew	4	5	6
Apple	0.5	0.95	15
Grapefruit	1	0.75	10
Orange	0.7	0.6	15
Bunch of 3 bananas	0.9	1	6
Mango	1.2	2	10

7.2.2: Formulate the Girlscout Fruit Problem in Exercise 7.2.1 as a $0-1$ knapsack problem (Hint: Introduce groups of new variables, all variables in each group associated with the same data).

7.2.3: A Capital Budgeting Problem: There is a total of $w_0 =$ \$35 mil. available to invest. There are 8 independent investment possibilities, with the jth one costing w_j in mil.\$, and yielding an annual payoff of v_j in units of \$10,000, $j = 1$ to 8. The following table provides this data. Each investment possibility requires full participation, partial investments are not acceptable. The problem is to select a subset of these possibilities to invest, to maximize the total annual payoff (measured in units of \$10,000) subject to the constraint on available funds. Formulate as a knapsack problem.

Investment possibility j	cost w_j in \$mil.	Annual payoff v_j in \$10,000 units
1	3	12
2	4	12
3	3	9
4	3	15
5	15	90
6	13	26
7	16	112
8	12	62

7.2.4: A Multiperiod Capital Budgeting Problem: An investor who is expecting to receive sizable income annually over the next three years is investigating 8 independent projects to invest the spare income. Each project requires full participation, no partial participation is allowed. If selected, a project may require cash contributions yearly over the next 3 years, as in the following table. At the end of the 4th year, the investor expects to sell off all the selected projects at the expected prices given in the following table. The investor needs to determine the subset of projects to invest in, to maximize the total expected amount obtained by selling off the projects at the end of the

4th year, subject to the constraint on available funds in years 1, 2, 3. Formulate as a multidimensional knapsack problem.

Project	Investment in $\$10^4$ units needed in year			Expected selling price in 4th year in $\$10^4$ units
	1	2	3	
1	20	30	10	70
2	40	20	0	75
3	50	30	10	110
4	25	25	35	105
5	15	25	30	85
6	7	22	23	65
7	23	23	23	82
8	13	28	15	70
Funds available	95	70	65	

7.2.5: Problem with Multiple Choice Constraints: Consider the investment problem discussed in Exercise 7.2.3. Projects 1, 2 there deal with fertilizer manufacturing; projects 3, 4 deal with tractor leasing; and projects 5, 6, 7, 8 are miscellaneous projects. The investor would like to invest in one fertilizer project, one tractor leasing project, and at least one miscellaneous project. Derive a formulation of the problem that includes these additional constraints.

7.3 Set Covering, Set Packing, and Set Partitioning Problems

Consider the following problem faced by the US Senate. They have various committees having responsibility for carrying out the senate's work, or pursuing various investigations. Membership in committees brings prestige and visibility to the senators, and they are quite vigorously contested. We present an example dealing with that of forming a senate committee to investigate a political problem. There are 10 senators, numbered 1 to 10, who are eligible to serve on this committee. They belong to the following groups.

Group	Eligible senators in this group
Southern senators	{1, 2, 3, 4, 5}
Northern senators	{6, 7, 8, 9, 10}
Liberals	{2, 3, 8, 9, 10}
Conservatives	{1, 5, 6, 7}
Democrats	{3, 4, 5, 6, 7, 9}
Republicans	{1, 2, 8, 10}

It is required to form the smallest size committee which contains at least one representative from each of the above groups. Notice that here the groups are not disjoint, and the same person may belong to several groups. Each person selected as a member of the committee will be counted as representing each group to which s/he belongs. So, the size of an optimum committee may be smaller that the number of groups.

We will give a 0−1 integer formulation for this problem. For $j = 1$ to 10, define

$$x_j = \begin{cases} 1, & \text{if senator } j \text{ is selected for the committee} \\ 0, & \text{otherwise} \end{cases}$$

Then the committee size is $\sum x_j$ which has to be minimized. From the definition of the decision variables, we see that the number of senators selected for the committee, from group 1 is $x_1 + x_2 + x_3 + x_4 + x_5$ and this is required to be ≥ 1. Continuing in the same way, we get the following integer programming formulation for this problem

$$\text{Minimize} \quad z(x) = \sum_{j=1}^{10} x_j$$

$$\begin{aligned} \text{subject to} \quad x_1 + x_2 + x_3 + x_4 + x_5 &\geq 1 \\ x_6 + x_7 + x_8 + x_9 + x_{10} &\geq 1 \\ x_2 + x_3 + x_8 + x_9 + x_{10} &\geq 1 \end{aligned}$$

$$\begin{aligned} x_1 + x_5 + x_6 + x_7 &\geq 1 \\ x_3 + x_4 + x_5 + x_6 + x_7 + x_9 &\geq 1 \\ x_1 + x_2 + x_8 + x_{10} &\geq 1 \\ x_j = 0 \ \text{ or } 1 \text{ for all } \ & j \end{aligned}$$

This is a pure 0−1 integer program in which all the constraints are ≥ inequalities, all the right hand side constants are 1, and the matrix of coefficients is a 0−1 matrix. Each constraint corresponds to a group, and when a 0−1 solution satisfies it, the associated committee has at least one member from this group. A 0−1 integer program of this form is called a **set covering problem**. The general set covering problem is of the following form

$$\begin{aligned} \text{Minimize} \quad & z(x) = cx \\ \text{subject to} \quad & Ax \geq e \\ & x_j = 0 \ \text{ or } \ 1 \qquad \text{for all } j \end{aligned} \tag{7.3.1}$$

where A is a 0−1 matrix of order $m \times n$, and e is the vector of all 1s in R^m. The set covering model has many applications, we now provide examples of some of its important applications.

Example 7.3.1: **Application in Delivery and Routing Problems using a Column Generation Approach:**

These problems are also called **truck dispatching** or **truck scheduling problems**. A **warehouse** (or sometimes referred to as a **depot**) with a fleet of trucks has to make deliveries to m customers in a region. In Figure 7.3, the warehouse location is marked by "□", and customer locations are marked by an "x". The problem is to make up routes for trucks which begin at the depot, visit customers to make deliveries, and return to the depot at the end. The input data consists of the cost (either distance or driving time) for traveling between every pair of locations among the depot and the customers, the quantity to be delivered to each customer, sayin tons, and the capacity of each truck in tons. A single truck route covering all the customers may not be

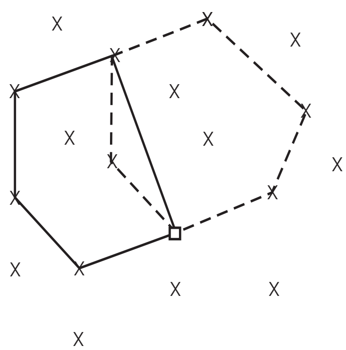

Figure 7.3: "□" marks the location of the depot. Each customers location is marked by an "x". Two feasible truck routes are shown, one in dashed lines, another in solid lines.

feasible if the total quantity to be delivered to all the customers exceeds the truck capacity, or if the total distance or time of the route exceeds the distance or time that a truck driver can work per day by union regulations or company policy. So, the problem is to partition the set of all customers into subsets each of which can be handled by a truck in a feasible manner, and the actual route to be followed by each truck (i.e., the order in which the truck will visit the customers in its subset), so as to minimize the total cost incurred in making all the deliveries.

One approach for solving this problem generates a **List** consisting of a large number of feasible routes which are good (i.e., have good cost performance for the deliveries they make) one after the other using appropriate heuristic rules, and selects a subset of them to implement using a set covering model. In Figure 7.3 we show two routes, one with dashed lines, and the other with continuous lines. Let n denote the list size, i.e., the total number of routes generated, and c_j the cost of route j, $j = 1$ to n. Each customer may lie on several of the routes generated, in fact the larger n is, the better the final output. Let $\mathbf{F}_i$ denote the subset of routes in the list, which contain the ith customer, $i = 1$ to m. Since each of the m customers has to be visited, at least one of the routes from the set $\mathbf{F}_i$ has to be implemented, for $i = 1$ to m. Define

$$x_j = \begin{cases} 1, & \text{if the } j \text{ route is implemented by a truck} \\ 0, & \text{otherwise} \end{cases}$$

Then the problem of finding the best subset of routes to implement is

$$\begin{array}{lll} \text{Minimize} & \sum_{j=1}^{n} c_j x_j & \\ \text{subject to} & \sum_{j \in \mathbf{F}_i} x_j \geq 1, & \text{for each } i = 1 \text{ to } m \\ & x_j = 0 \text{ or } 1 & \text{for all } j = 1 \text{ to } n \end{array} \qquad (7.3.2)$$

(7.3.2) is a set covering problem. If $\bar{x}$ is an optimum solution of (7.3.2), the routes to implement are those in the set $\{j\text{: } \bar{x}_j = 1\}$. If only one route from this set contains a customer i on it, the truck following that route makes the delivery to this customer. If two or more routes from this set contain customer i on them, we select any one of the trucks following these routes to make the delivery to customer i, and the other trucks pass through customer i's location on their route without stopping.

Define the **incidence vector** of a route in the list as a column vector of order m where for each $i = 1$ to m the entry in position i is 1 if this route visits customer i, 0 otherwise. For example, if there are 10 customers in all numbered 1 to 10, and if a route visits customers 2, 6, 8, 9; them its incidence vector will be $(0, 1, 0, 0, 0, 1, 0, 1, 1, 0)^T$.

Then we verify that the incidence vector of route j in the list is actually the column vector of the corresponding variable x_j in the model (7.3.2). Hence, each route in the list generated corresponds to a variable in the set covering model, and the column vector of that variable is actually the incidence vector of that route.

That's why this type of approach for modeling problems is commonly known as the **column generation approach**. For handling large scale routing and other allocation problems, column generation works with a generated list of good solution components, and selects a subset of them to implement using a set covering or some other 0−1 model.

In applications, there are many problems that are too complex and not amenable to a direct mathematical model to find an optimum solution for them. In some of these problems, the solution consists of several components that need to cover or span a given set of requirements. Column generation is a common approach used to handle such

Route no.	Route	Cost
R_1	0-3-8-0	6
R_2	0-1-3-7-0	8
R_3	0-2-4-1-5-0	9
R_4	0-4-6-8-0	10
R_5	0-5-7-6-0	7
R_6	0-8-2-7-0	11
R_7	0-1-8-6-0	8
R_8	0-8-4-2-0	7
R_9	0-3-5-0	7

x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	
0	1	1	0	0	0	1	0	0	≥ 1
0	0	1	0	0	1	0	1	0	≥ 1
1	1	0	0	0	0	0	0	1	≥ 1
0	0	1	1	0	0	0	1	0	≥ 1
0	0	1	0	1	0	0	0	1	≥ 1
0	0	0	1	1	0	1	0	0	≥ 1
0	1	0	0	1	1	0	0	0	≥ 1
1	0	0	1	0	1	1	1	0	≥ 1
6	8	9	10	7	11	8	7	7	$= z$, min.

$x_j = 0$ or 1 for all j.

problems. It involves enumerating several (i.e., typically as many as possible) good components based on practical considerations, and then selecting the best subset of them to implement using either an integer programming model (like the set covering model discussed in this example) or a linear programming model and some heuristic procedures.

Here is a numerical example from a delivery problem. A depot numbered 0 has to make deliveries to customers at locations numbered 1 to 8. A set of 9 good routes for delivery vehicles have been generated and given in the table above. The first route 0-3-8-0 means that the vehicle starts at the depot 0, visits customer 3 first, from there goes to visit customer 8, and from there returns to the depot. The cost of the route given below is its expected driving time in hours. The integer programming formulation for determining which of these 9 routes should be implemented so as to minimize the total driving time of all the vehicles used to make the deliveries uses binary decision variables x_j for $j = 1$ to 9; where x_j takes the value 1 if the route R_j is implemented, 0 otherwise; is given after the table.

The i-th contraint in the model corresponds to the i-th customer location for $i = 1$ to 8. For example the first constraint $x_2 + x_3 + x_7 \geq 1$ requires that among the set of routes $\{R_2, R_3, R_7\}$ at least one must be implemented, as these are the only routes in the generated list that pass through customer location 1.

Example 7.3.2: **Locating Fire Hydrants, a Node (or Vertex) Covering Model:**

Given a network of traffic centers (nodes, also called vertices, in a network), and street segments (edges in the network, each edge joining a pair of nodes), this problem is to find a subset of nodes for locating fire hydrants so that each street segment contains at least one fire hydrant. Suppose there are n nodes numbered 1 to n, and let c_j be the cost of locating a fire hydrant at node j. A subset of nodes in the network satisfying the property that every edge in the network contains at least one node from the subset is called a **node** (or **vertex cover)** for the network. The constraint requires that the subset of nodes where fire hydrants are located should be a node (or vertex) cover, covering all

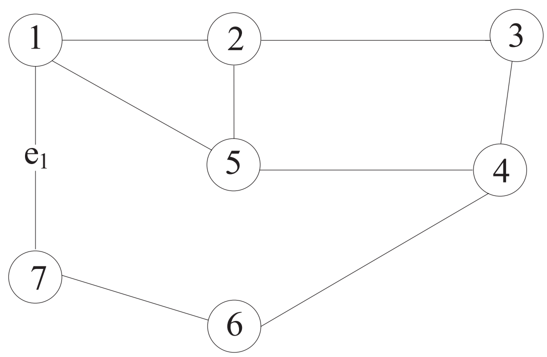

Figure 7.4: Street network for fire hydrant location problem. Nodes are traffic centers. Edges are street segments.

$$
\begin{array}{lllllllll}
\text{Minimize} & \multicolumn{7}{c}{\sum_{j=1}^{7} c_j x_j} & \\
\text{subject to} & x_1 & & & & & & +x_7 & \geq 1 \\
& x_1 & +x_2 & & & & & & \geq 1 \\
& x_1 & & & & +x_5 & & & \geq 1 \\
& & x_2 & +x_3 & & & & & \geq 1 \\
& & x_2 & & & +x_5 & & & \geq 1 \\
& & & x_3 & +x_4 & & & & \geq 1 \\
& & & & x_4 & +x_5 & & & \geq 1 \\
& & & & x_4 & & +x_6 & & \geq 1 \\
& & & & & & x_6 & +x_7 & \geq 1
\end{array}
$$

$$x_j = 0 \text{ or } 1 \text{ for all } j$$

the edges. Thus the problem is to determine a minimum cost node cover. Define for $j = 1$ to n

$$x_j = \begin{cases} 1, & \text{if a fire hydrant is located at node } j \\ 0, & \text{otherwise} \end{cases}$$

Then the problem is to determine a $0-1$ vector x to minimize $\sum c_j x_j$ subject to the constraints that $x_i + x_j \geq 1$ for every edge $(i;j)$ in the network. For the network in Figure 7.4, the problem is the one following the figure.

Each constraint in this model corresponds to an edge in the network. For example, the first constraint requires that a fire hydrant should be located at at least one of the two nodes 1, 7 on the edge e_1 in the network in Figure 7.4. This is a set covering problem in which each constraint involves exactly two variables. Such problems are known as **node covering** or **vertex covering problems** in networks.

Example 7.3.3: **Facility Location Problems:** These problems have the following features. A region is partitioned into m neighborhoods, each of which requires the use of some facility (fire stations, snow removal equipment banks, etc.). There are n possible locations in the region for building these facilities. d_{ij} = the distance in miles between neighborhood i and location j, is given for all $i = 1$ to m, $j =$ 1 to n. (A neighborhood could be a large area; the distance between a location and a neighborhood is usually defined to be the distance between the location and the population center of the neighborhood.) c_j = the cost of building a facility at location j, is given for $j = 1$ to n. There is a state restriction that every neighborhood must be within a distance of at most d miles from its nearest facility. The problem is to select a minimum cost subset of locations to build the facilities that meets the state's restrictions. For $i = 1$ to m define $\mathbf{F}_i = \{\text{location } j\text{: } d_{ij} \leq d\}$. Let

$$x_j = \begin{cases} 1, & \text{if a facility is built at location } j \\ 0, & \text{otherwise} \end{cases}$$

Then the problem of finding the optimum subset of locations to build the facilities is exactly the set covering problem of the form (7.3.2) with the definitions of x_j and $\mathbf{F}_i$ as stated here.

Here is a numerical example in which a residential region is divided into 8 zones. The best location for a fire station in each zone has been determined already. From these locations and the population centers in each zone, we have estimates for the average number of minutes of fire truck driving time to respond to an emergency in zone j from a possible fire station located in zone i, given in the first table below. An estimate of more than 75 minutes indicates that it is not feasible to respond to an emergency within reasonable time using that route, so that cell is left blank. Because of traffic patterns etc., the estimate matrix is not symmetric.

	Average driving time							
	to $j = 1$	2	3	4	5	6	7	8
from $i = 1$	10		25		40			30
2		8	60	35		60	20	
3	30		5	15	30	60	20	
4	25		30	15	30	60	25	
5	40		60	35	10		32	23
6		50	40	70		20		25
7	60	20		20	35		14	24
8	30		25		25	30	25	9

x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	
1	0	0	1	0	0	0	0	≥ 1
0	1	0	0	0	0	1	0	≥ 1
1	0	1	0	0	0	0	1	≥ 1
0	0	1	1	0	0	1	0	≥ 1
0	0	0	0	1	0	0	1	≥ 1
0	0	0	0	0	1	0	0	≥ 1
0	1	1	1	0	0	1	1	≥ 1
0	0	0	0	1	1	1	1	≥ 1

$x_j = 0$ or 1 for all j.

It is not necessary to have a fire station in each zone, but each zone must be within an average 25 minute driving time reach of a fire station. We are required to formulate the problem of determining the zones in which fire stations should be located, so as to meet the constraint stated above with the smallest number of fire stations.

Define the decision variables: for $i = 1$ to 8, let x_i be the binary variable which takes the value 1 if a fire station is set up in the location in Zone i, 0 otherwise.

There will be a constraint in the model corresponding to each zone j, that requires x_i to be equal to 1 for at least one i such that the distance from location i to zone j is ≤ 25 minutes. So, for Zone $j = 1$

this constraint is $x_1 + x_4 \geq 1$, because locations 1, 4 are the only ones from which the driving time to Zone 1 is ≤ 25 minutes. Arguing the same way, we get the model for the problem given above. It is a set covering model.

Example 7.3.4: **Airline Crew Scheduling (or Crew Pairing) Problem:** This is a very important large scale application for

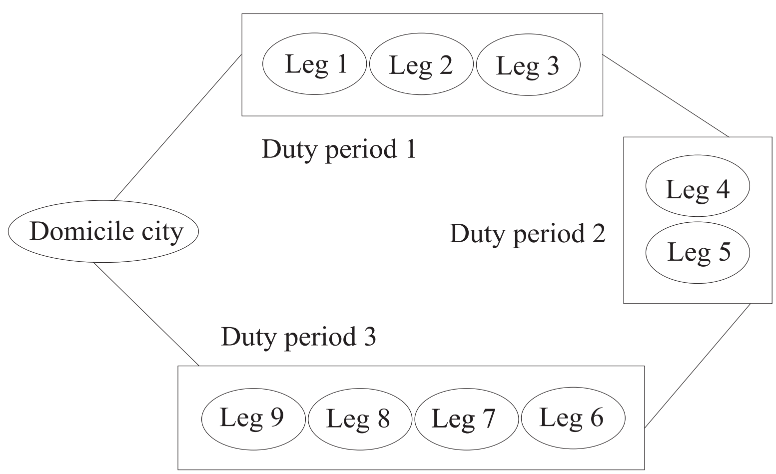

Figure 7.5: A pairing for a crew in airline operations.

the set covering model. The basic elements in this problem are **flight legs**. A flight leg is a flight between two cities, departing at one city at a specified time and landing next at the second city at a specified time, in an airline's timetable. A **duty period** for a crew is a continuous block of time during which the crew is on duty, consisting of a sequence of flight legs each one following the other in chronological order. A **pairing** for a crew is a sequence of duty periods that begins and ends at the same domicile. Federal aviation regulations, union rules, and company policies impose a complex set of restrictions in the formation of pairings. In particular, a duty period can contain no more than 7 flight legs, and cannot exceed 12 hours in duration; and a minimum rest period of 9.5 hours is required between consecutive duty periods in a pairing. And the crew can fly no more than 16 hours in any 48 hour interval. A pairing may include several days of work for a crew.

Even for a moderate sized airline the monthly crew scheduling problem may involve 500 flight legs. The problem of forming a minimum cost set of pairings which cover all the flight legs in a time table is a tantalizing combinatorial optimization problem.

One approach (a column generation approach) for handling this problem proceeds as follows. It generates a list consisting of a large number of good pairings using a pairing generator to enumerate candidate crew schedules, and computes the cost of each pairing. Suppose the list has n pairings. Each flight leg may appear in several pairings in the list. For $i = 1$ to m, let $\mathbf{F}_i$ = set of all pairings in the list that contain the ith flight leg. For $j = 1$ to n let c_j be the cost of the jth pairing in the list. Define for $j = 1$ to n

$$x_j = \begin{cases} 1, & \text{if the } j\text{th pairing in the list is implemented} \\ 0, & \text{otherwise} \end{cases}$$

Then the problem of selecting a minimum cost subset of pairings in the list to implement to cover all the flight legs is exactly the set covering problem of the form (7.3.2) with x_j and $\mathbf{F}_i$ as defined here.

The quality of solutions obtained improves when the list contains pairings of a variety of mixes including samples of as many types of combinations as possible. Crew pairing planners usually generate many thousands of pairings, and the resulting set covering model to select the subset of pairings in the list to implement becomes a very large scale problem that may take a long time to solve exactly. Often heuristic approaches are used to obtain good solutions for these models. $\bowtie$

The set covering model finds many other applications in such diverse areas as the design of switching circuits, assembly line balancing, etc.

The Set Packing Problem

A pure 0−1 integer program of the following form is known as a **set packing problem**.

$$\text{Maximize} \quad z(x) = cx$$

$$\text{subject to} \quad Ax \leq e \qquad (7.3.3)$$
$$x_j = 0 \text{ or } 1 \qquad \text{for all } j$$

where A is a 0−1 matrix of order $m \times n$, and e is the vector of all 1's in R^m.

Example 7.3.5: **Meetings scheduling problem:** Here is an example of an application of the set packing problem. Large organizations such as big hospitals etc. are run by teams of administrators. In the course of the workweek these administrators attend several meetings where decisions are taken, and administrative and policy problems are ironed out. This application is concerned with the timely scheduling of the necessary meetings. Suppose in a particular week there are n different meetings to be scheduled. For the sake of simplicity assume that each meeting lasts exactly one hour. Suppose we have T different time slots of one hour duration each, available during the week to hold the meetings (for example, if meetings can be held every morning Monday to Friday from 8 to 10 AM, we have $T = 10$ time slots available). Suppose there are k administrators in all, and we are given the following data: for $i = 1$ to k, $j = 1$ to n

$$a_{ij} = \begin{cases} 1, & \text{if the } i\text{th administrator has to attend the } j\text{th meeting} \\ 0, & \text{otherwise} \end{cases}$$

The 0−1 matrix (a_{ij}) is given. If two meetings require the attendance of the same administrator, they cannot both be scheduled in the same time slot, because that will create a conflict for that administrator. On the other hand if there is no common administrator that is required to attend two meetings, both of them can be scheduled in the same time slot. The problem is to schedule as many of the n meetings as possible in the T time slots available, subject to these conditions. For $j = 1$ to n, $t = 1$ to T, define

$$x_{jt} = \begin{cases} 1, & \text{if meeting } j \text{ is scheduled for time slot } t \\ 0, & \text{otherwise} \end{cases}$$

Then the problem of scheduling as many of the meetings as possible in the available time slots without creating any conflicts for any administrator is the following set packing model

$$
\begin{array}{lrcll}
\text{Maximize} & \sum_{j=1}^{n}\sum_{t=1}^{T} x_{jt} & & & \\
\text{subject to} & \sum_{j=1}^{n} a_{ij}x_{jt} & \leq & 1, & \text{for } i = 1 \text{ to } k,\ t = 1 \text{ to } T \\
& \sum_{t=1}^{T} x_{jt} & \leq & 1, & \text{for } j = 1 \text{ to } n \\
& x_{jt} & = & 0 & \text{or 1 for all } j, t
\end{array}
\qquad (7.3.4)
$$

The first set of constraints represents the fact that each administrator can attend at most one meeting in any time slot. The second set of constraints assures that each meeting is assigned at most one time slot.

Here is a numerical example from an undergraduate student project

Meeting	Administrators that must attend
M_1	1, 2, 3
M_2	1, 4, 5, 8
M_3	3, 5, 6
M_4	2, 4, 6, 13
M_5	8, 9
M_6	10, 11, 13
M_7	5, 7
M_8	9, 10, 12
M_9	7, 11, 12

I supervised in the 1970's for the UM (University of Michigan) Hospital. In those days the hospital was administered by a large team of administrators. The administrators would periodically have a large number of meetings to discuss various policy issues; and these meetings would be held over a whole week. Each meeting would last about half-a-day, and would require the attendance of a specified subset of administrators, even though some other interested administrators may attend voluntarily. Meetings that did not require the attendance of a common administrator can be held concurrently. Those with a common required administrator must be scheduled in different half-day periods.

We show the data for a set of 9 meetings involving 13 administrators numbered 1 to 13. The table given above provides the data on the set of administrators required to attend each meeting.

$A =$

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9
1	1	1	0	0	0	0	0	0	0
2	1	0	0	1	0	0	0	0	0
3	1	0	1	0	0	0	0	0	0
4	0	1	0	1	0	0	0	0	0
5	0	1	1	0	0	0	1	0	0
6	0	0	1	1	0	0	0	0	0
7	0	0	0	0	0	0	1	0	1
8	0	1	0	0	1	0	0	0	0
9	0	0	0	0	1	0	0	1	0
10	0	0	0	0	0	1	0	1	0
11	0	0	0	0	0	1	0	0	1
12	0	0	0	0	0	0	0	1	1
13	0	0	0	1	0	1	0	0	0

We consider the problem of finding the maximum number of these meetings that can be scheduled without conflicts over a time horizon consisting of three half-days.

For $j = 1$ to 9, $t = 1, 2, 3$, define the binary decision variable x_{jt} which takes the value 1 if M_j is held in t-th half-day time slot; 0 otherwise.

Also, let $A = (a_{ij})$ be the matrix of order 13×9 where $a_{ij} = 1$ if the i-th administrator is required to attend the j-th meeting, 0 otherwise, given above. Then the model to find a schedule that maximizes the number of meetings that can be held in time periods 1, 2, 3 is:

$$\text{Maximize} \quad \sum_{j=1}^{9}\sum_{t=1}^{3} x_{jt}$$

$$\text{subject to} \quad \sum_{j=1}^{9} a_{ij}x_{jt} \leq 1, \quad \text{for } i = 1 \text{ to } 13,\ t = 1 \text{ to } 3$$

$$\sum_{t=1}^{3} x_{jt} \leq 1, \quad \text{for } j = 1 \text{ to } 9$$
$$x_{jt} = 0 \text{ or } 1 \text{ for all } j, t$$

For example the first constraint above, $\sum_{j=1}^{9} a_{ij}x_{jt} \leq 1$ for all $t = 1$ to 3 implies that in each time slot, at most one meeting that requires the i-th administrator's attendance can be held.

In this example we illustrated one application of the set packing model to determine the maximum number of meetings that can be held witrhout conflicts in a limited time period. In the UM Hospital project they were actually interested in determining a schedule for all the meetings without conflicts using the smallest time period. This problem has a much simpler formulation directly as a combinatorial model which we used. This model is discussed in Section 7.9.

The Set Partitioning Problem

A set partitioning problem is a $0-1$ pure integer program of the following form

$$\begin{array}{rl} \text{Minimize} \quad z(x) = cx & \\ \text{subject to} \quad Ax = e & \\ x_j = 0 \text{ or } 1 & \text{for all } j \end{array} \tag{7.3.5}$$

where A is a $0-1$ matrix, and e is the column vector of all 1s of appropriate order. Notice the difference between the set covering problem (in which the constraints are of the form $Ax \geq e$), the set packing problem (in which the constraints are of the form $Ax \leq e$), and the set partitioning problem (in which the constraints are of the form $Ax = e$).

The set partitioning problem also finds many applications. One of them is the following. Consider a region consisting of many, say m, sales areas numbered 1 to m. These areas have to be arranged into groups to be called **sales districts** such that each district can be handled by one sales representative. The problem is to determine how to form the various sales areas into districts. One approach for handling this problem (a column generation approach) involves generating a list consisting of a large number of subsets of sales areas, each of which could form a good district (i.e., provides enough work for a sales representative and satisfies any other constraints that may be required). Let n be the number of such subsets generated, number them as subsets $1, \ldots, n$. For $i = 1$ to m, let $\mathbf{F}_i = \{j$: subset j includes the ith sales area$\}$. For $j = 1$ to n let c_j denote the cost of forming the subset j in the list into a sales district. The approach now selects subsets from the list to form into sales districts using a set partitioning model. Define

$$x_j = \begin{cases} 1, & \text{if the } j\text{th subset in the list is formed into a sales district} \\ 0, & \text{otherwise} \end{cases}$$

Since each sales area must be in a district, our problem leads to the following set partitioning model.

$$\begin{array}{lrcll} \text{Minimize} & \sum_{j=1}^{n} c_j x_j & & & \\ \text{subject to} & \sum_{j \in \mathbf{F}_i} x_j & = & 1, & \text{for each } i = 1 \text{ to } m \\ & x_j = 0 \quad \text{or } 1 & & & \text{for all } j \end{array}$$

In a similar manner, the set partitioning model has applications in political districting, and in various other problems in which a set has to be partitioned at minimum cost subject to various conditions.

Example 7.3.6: **A political districting example:**

Here is a highly simplified version of the political districting problem. A region (consisting of one or more cities) is divided into zones called wards for administrative and representative purposes. A precinct is an election (or political) district composed of a set of wards, which is a geographical area from which a representative will be elected for a political office. This example deals with the problem of forming precincts out of a region consisting of 14 wards numbered 1 to 14. The following list of 16 subsets of wards has been formed. Each of these subsets satisfies all the conditions required for a subset of wards to be a precinct.

$\{\{1,2,3,4\},\{5,6,7,8\},\{9,10,11\},\{12,13,14\},\{1,4,6,8\},\{2,3,9,10\},\{5,7,12\},\{11,13,14\},\{4,6,9,12\},\{2,5,7,14\},\{1,3,10,11\},\{8,13\},\{1,5,9,12\},\{2,8,11\},\{3,6,7,10\},\{9,13,14\}\}$.

Call subsets in the list S_1 to S_{16} in that order. Based on past voting records, the republican party estimates that the chance of a republican candidate winning in these subsets is the following vector: $p = (0.45, 0.33, 0.78, 0.56, 0.85, 0.28, 0.67, 0.91, 0.35, 0.45, 0.18, 0.47, 0.29, 0.39, 0.15, 0.21)^T$.

Clearly the expected number of republican candidates who win in this region, is the sum of the above probabilities over subsets in the list which are selected as precincts. In this example we will formulate the problem of determining which subsets in the list the republican party should champion for being selected as precincts. The constraints are that the precincts selected should form a partition of the set of wards, i.e., each ward should belong to one and only one precinct.

For $j = 1$ to 16, define the binary decision variable x_j which takes the value 1 if the subset S_j is selected as a precinct, 0 otherwise. Let $x = (x_j)$, a column vector. Then the expected number of republican candidates winning is $\sum_1^{16} p_j x_j = p^T x$, which is the objective function to be maximized.

The constraints in the model come from the requirement that each ward should lie in exactly one precinct. we verify that Ward 1 lies in the subsets S_1, S_5, S_{11}, S_{13}, so this leads to the constraint $x_1 + x_5 + x_{11} + x_{13} = 1$.

In the same way each of the wards leads to one constraint in the model for the problem given below.

$$
\begin{array}{rrcl}
 & \text{Maximize} \quad p^T x & & \\
\text{subject to} & x_1 + x_5 + x_{11} + x_{13} & = & 1 \\
 & x_1 + x_6 + x_{10} + x_{14} & = & 1 \\
 & x_1 + x_6 + x_{11} + x_{15} & = & 1 \\
 & x_1 + x_5 + x_9 & = & 1 \\
 & x_2 + x_7 + x_{10} + x_{13} & = & 1 \\
 & x_2 + x_5 + x_9 + x_{15} & = & 1
\end{array}
$$

$$
\begin{aligned}
x_2 + x_7 + x_{10} + x_{15} &= 1 \\
x_2 + x_5 + x_{12} + x_{14} &= 1 \\
x_3 + x_6 + x_9 + x_{13} + x_{16} &= 1 \\
x_3 + x_6 + x_{11} + x_{15} &= 1 \\
x_3 + x_8 + x_{11} + x_{14} &= 1 \\
x_4 + x_7 + x_9 + x_{13} &= 1 \\
x_4 + x_8 + x_{12} + x_{16} &= 1 \\
x_4 + x_8 + x_{10} + x_{16} &= 1 \\
\\
\text{all } x_j &\in \{0, 1\}
\end{aligned}
$$

They should champion all the subsets S_j corresponding to $x_j = 1$ in an optimum solution of this model, to become precincts.

Exercises

7.3.1: A Facilities Location Problem: A newly developing

	Average driving time								
	to $j = 1$	2	3	4	5	6	7	8	9
from $i = 1$	7	30	40			20		14	31
2	25	12		24	20		29	38	
3	35		15	35	18		20		12
4		22	35	5		10	32		
5		24	13		10	38		12	13
6	13			10	40	13			15
7		16	14				8	39	10
8	20	42		30	9		42	17	12
9	29		15		16	19	10	19	20

region is divided into 9 zones. In each zone there is a location reserved for setting up a fire station, if a decision is made to set up a fire station in that zone. From these locations, we have the following estimates for the average number of minutes of fire truck driving time to respond to an emergency in zone j from a possible fire station located in zone i. An estimate of more than 45 minutes indicates that it is not feasible to respond to an emergency within reasonable time using that route, so that cell is left blank. Because of traffic patterns etc., the estimate matrix is not symmetric.

It is not necessary to have a fire station in each zone, but each zone must be within an average 25 minute driving time reach of a fire station. Formulate the problem of determining the zones in which fire stations should be located, so as to meet the constraint stated above with the smallest number of fire stations.

7.3.2: A delivery problem: From the depot numbered 0, deliveries have to be made to customers at locations 1 to 10. To model this problem using column generation, the following list of good routes has been generated. Here routes are denoted in this way: 0-1-2-3-0, 40.6; this means that the vehicle following this route starts at the depot 0, stops at locations 1, 2, 3 making deliveries at these locations in this order, and then returns to depot 0; and that the total length of this route is 40.6 miles.

List = {0-1-2-3-0, 40.6; 0-3-4-7-0, 45.6; 0-2-1-5-6-0, 42.3; 0-7-6-1-0, 38.9; 0-5-9-4-0, 29.3; 0-8-6-4-2-0, 34.7; 0-3-9-4-0, 37.6; 0-4-2-5-0, 45.3; 0-3-9-8-0, 28.8; 0-5-7-9-0, 36.4; 0-1-5-6,0, 31.7}.

Comment on whether this is a good list of routes for modeling our problem by column generation. Explain your reasons for the same.

In addition to the above, include these additional routes: {0-9-5-7-0, 28.7; 0-8-10-2-0, 44.3; 0-1-10-3-0, 39.9; 0-10-6-2-0, 43.9}, and write the model for the problem of minimizing the total mileage of all the vehicles used.

If each of these routes is one days work for a driver, write the model for completing the deliveries using the smallest number of drivers.

7.3.3: A node covering problem: Write the model for the node covering problem in the network given in Figure 7.6.

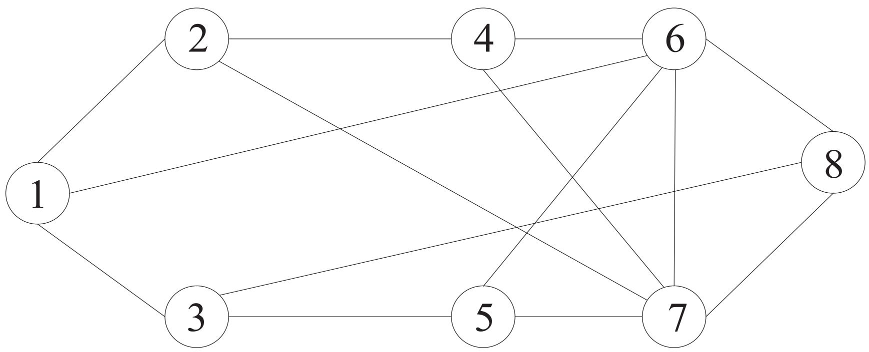

Figure 7.6:

7.3.4: A meeting scheduling problem: There are 8 top administrators numbered 1 to 8 in a company. In a particular week 7 different meetings are to be held for important planning discussions. Each meeting needs the attendance of some of these administrators as explained in the following table.

Meeting	Presence required of:
M_1	1, 4, 6
M_2	2, 5, 4, 7
M_3	3, 8, 1, 2
M_4	2, 3, 5
M_5	7, 8
M_6	3, 4, 5
M_7	1, 6, 8

These meetings will be held in a conference center that has many suitable rooms to hold meetings. Each meeting requires half-a-day. Meetings that do not require the attendance of a common administrator can be held concurrently.

Formulate the problem of finding the maximum number of these meetings that can be scheduled in one day.

Discuss a strategy to find the smallest total time period needed for holding all the meetings, using this set packing model more than once if necessary.

7.3.5: A political districting problem: This is a political districting problem similar to that in Example 7.3.6, and uses the same terminology. In this problem there are 12 wards numbered 1 to 12. The following list of subsets of wards have been formed, each satisfying all the conditions for being a precinct.

$\{\{1,2\}$, $\{3,4,5\}$, $\{6,7\}$, $\{8,9,10\}$, $\{11,12\}$, $\{1,3,6\}$, $\{2,4,5\}$, $\{7,8,9\}$, $\{10,11,12\}$, $\{1,6\}$, $\{2,8,10\}$, $\{3,5,11\}$, $\{4,7\}$, $\{9,12\}$, $\{1,8,9\}$, $\{2,12,5\}$, $\{3,7,11\}$, $\{4,6,10\}\}$.

Call these subsets S_1 to S_{18}. The democrats estimate that the probability of the democratic candidate winning in these subsets is the vector $p = (0.81, 0.43, 0.28, 0.59, 0.63, 0.74, 0.31, 0.39, 0.69, 0.58, 0.49, 0.57, 0.63, 0.30, 0.29, 0.46, 0.38, 0.57)^T$.

Formulate the problem of determining which subsets the democratic party should champion to be made into precincts, to maximize the expected number of democrastic candidates winning.

7.4 Plant Location Problems

Plant location problems are an important class of problems that can be modeled as MIPs. The simplest problems of this type have the following structure. There are n sites in a region that require a product. Over the planning horizon, the demand for the product in the area containing site i is estimated to be d_i units, $i = 1$ to n. The demand has to be met by manufacturing the product within the region. A decision has been taken to set up at most m plants for manufacturing the product. The set-up cost for building a plant at site i is $\$f_i$, and its production capacity will be at most k_i units over the planning horizon, $i = 1$ to n. $\$c_{ij}$ is the cost of transporting the product per unit from site i to site j.

In practice, m = the number of plants to be set up, will be much smaller than n = the number of sites where plants can be set up, and the product will be shipped from where it is manufactured to all other sites in the region. The problem is to determine an optimal subset of sites for locating the plants, and a shipping plan over the entire horizon so as to meet the demands at minimum total cost which includes the cost of building the plants and transportation costs. To determine the subset of sites for locating the plants is a combinatorial optimization problem. Once the optimum solution of this combinatorial problem is known, determining the amounts to be transported along the various routes is a simple transportation problem. For $i, j = 1$ to n, define

$$y_i = \begin{cases} 1, & \text{if a plant is located at site } i \\ 0, & \text{otherwise} \end{cases}$$

x_{ij} = units of product transported from site i to j over the planning horizon

The MIP model for the problem is

$$\begin{array}{rrcll}
\text{Minimize} & \sum_i f_i y_i & + & \sum_i \sum_j c_{ij} x_{ij} & \\
\text{subject to} & \sum_j x_{ij} - k_i y_i & \leq & 0 \quad \text{for all } i & (7.4.1) \\
& \sum_i x_{ij} & \geq & d_j \quad \text{for all } j & \\
& \sum_i y_i & \leq & m & \\
y_i = & 0 \text{ or } 1, \quad x_{ij} & \geq & 0 \quad \text{for all } i,\ j &
\end{array}$$

Other plant location problems may have more complicated constraints in them. They can be formulated as integer programs using similar ideas.

Illustrative Small Numerical Example

Suppose there are three sites, S_1, S_2, S_3 where production facilities can be set up. In the following table, money data is given in coded money units; production, demand, etc. are given in coded production units. Depots D_1 to D_4 will be set up to stock the product in four major markets, with each depot distributing the product in its region. Data is based on estimates for a 10-year lifecycle of the product. The company would like to set up at most two plants. We formulate the problem of meeting the demand over the lifecycle at minimum total cost. The decision variables are the x_{ij}, y_i for $i = 1$ to 3, $j = 1$ to 4. The MIP model for the problem is given following the data table:

Bldg.	c_{ij} to $j =$				f_i	p_i	k_i
Site i	D_1	D_2	D_3	D_4			
S_1	6	9	10	12	3000	60	10,000
S_2	15	8	6	4	5000	50	20,000
S_3	9	5	7	11	7000	55	15,000
d_j	9000	12000	7000	15000			

c_{ij} = transportation cost/unit from i to j, f_i = set up cost at i
p_i = production cost/unit at i, k_i = production capacity at i
d_j = estimated demand at depot D_j

$$\begin{array}{lrcl}
\text{Minimize} & 3000y_1 + 5000y_2 + 7000y_3 & + & \sum_{i=1}^{3} c_{ij}x_{ij} \\
& +60(\sum_{j=1}^{4} x_{1j}) + 50(\sum_{j=1}^{4} x_{2j}) & + & 55(\sum_{j=1}^{4} x_{3j}) \\
\text{subject to} & \sum_{j=1}^{4} x_{ij} - k_i y_i & \leq & 0, \quad \text{for } i = 1 \text{ to } 3 \\
& \sum_{i=1}^{3} x_{ij} & \geq & d_j \quad \text{for } j = 1 \text{ to } 4 \\
& \sum_{i=1}^{3} y_i & \leq & 2 \\
& y_i \text{ binary} & & x_{ij} \geq 0 \text{ for all } i, j
\end{array}$$

The Uncapacitated Plant Location Problem

In some applications there is freedom to select the production capacities of plants. Then the production capacity constraints do not apply, and the problem is known as the **uncapacitated plant location problem**. Here, if a plant is built at site i, there is no upper limit on how much can be shipped from this plant to any other sites. Also, if production cost/unit is the same at all the sites, the shipping cost itself is minimized if each site's demand d_j is completely satisfied from the plant at site i where i attains the minimum in $\min\{c_{rj}$: over r such that a plant is built at site$r\}$. As an example, suppose plants are built at sites 1 and 2. To meet site 3's demand, if $c_{13} = 10$ and $c_{23} = 20$ since each plant can produce as much as necessary, and at the same cost/unit, we would not ship any product to site 3 from the plant at site 2, since it is cheaper to ship from the plant at site 1 instead. So, in this case there exists an optimum shipping plan, in which each site receives all its demand from only one plant. Using this fact we can simplify the formulation of the problem in this case. For $i, j = 1$ to n, define new variables

$$z_{ij} \quad = \quad \text{fraction of demand at site } j \text{ shipped from a plant at site } i$$

So, these variables satisfy $\sum_i z_{ij} = 1$ for each $j = 1$ to n. We can think of the variable z_{ij} to be equal to x_{ij}/d_j in terms of the variable x_{ij} defined earlier. Also, since the production level at each plant depends on which sites it is required to supply in this case, the cost of setting it up may depend on that level. So, we assume that the cost of setting up a plant of capacity α at site i is $f_i + s_i\alpha$, where f_i is a fixed cost, and s_i is the variable cost of setting up production capacity/unit at site i. With y_i defined as before, here is the formulation of the problem.

$$\begin{array}{rrcl}
\text{Minimize} & \sum_{i=1}^{n} f_i y_i & + & \sum_{i=1}^{n} s_i (\sum_{j=1}^{n} d_j z_{ij}) + \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} d_j z_{ij} \\
\text{subject to} & \sum_{i=1}^{n} z_{ij} & = & 1, \quad j = \ 1 \text{ to } n \\
& \sum_{j=1}^{n} z_{ij} & \leq & n y_i, \quad i = \ 1 \text{ to } n \\
& z_{ij} \geq 0, y_i & = & 0 \text{ or } 1, \text{ for all } i, j
\end{array} \tag{7.4.2}$$

By the constraints in (7.4.2), $z_{ij} = 0$ for all j if no plant is set up at site i. Hence in the 2nd term in the objective function, the coefficient of s_i is guaranteed to be 0 if no plant is set up at site i.

Numerical Example

Consider the problem in the numerical example given above, but now suppose that $k_i = \infty$ for all $i = 1$ to 3, and $s = (s_i) = (100, 150, 120)$. To model this problem, the decision variables are the z_{ij} defined above for $i = 1$ to 3, $j = 1$ to 4. Once the plants are set up, assume that the production cost/unit is the same at all the sites. The model is (7.4.2) with j ranging over 1 to $n = 4$, and i ranging from 1 to 3.

Exercises

7.4.1: There are four sites S_1 to S_4, where a company can set up production facilities for a new product. The company will stock the product at depots D_1 to D_5 located in strategic locations. From these depots the product will be distributed to all the markets. All relevant data for the 15 year estimated lifetime of the product is given below. Formulate the problem at meeting the demand at minimum total cost if the company wants to set up at most three plants.

Bldg.	c_{ij} to $j =$					f_i	p_i	k_i
Site i	D_1	D_2	D_3	D_4	D_5			
S_1	2	6	4	9	7	6000	80	15000
S_2	8	3	5	6	2	9000	100	30000
S_3	9	7	6	4	8	10,000	90	40000
S_4	3	4	7	5	4	5000	60	50000
d_j	20000	35000	25000	17000	40000			

c_{ij} = transportation cost/unit from i to j, f_i = set up cost at i
p_i = production cost/unit at i, k_i = production capacity at i
d_j = estimated demand at depot D_j

7.4.2: Consider Exercise 7.4.1, but assume now that $k_i = \infty$ for all $i = 1$ to 4. Also assume that the cost of setting up a plant of capacity

α is $f_i + \alpha s_i$, where $s = (s_i) = (130, 150, 120, 125)$. Also, once the plants are set up, the production cost/unit at all the sites is the same. Formulate the problem with these changes.

7.5 Batch Size Problems

In addition to the usual linear equality-inequality constraints and nonnegativity restrictions in a linear program, suppose there are constraints of the following form: variable x_j in the model can be either 0, or if it is positive it must be $\geq$ some specified positive lower bound ℓ_j. Constraints of this type arise when the model includes variables that represent the amounts of some raw materials used, and the suppliers for these raw materials will only supply in amounts $\geq$ specified lower bounds.

There are two conditions on the decision variable x_j here, $x_j = 0$, or $x_j \geq \ell_j$, and the constraint requires that one of these two conditions must hold. We define a $0-1$ variable y_j to indicate these two possibilities for x_j, as given below.

$$y_j = \begin{cases} 0, & \text{if } x_j = 0 \\ 1, & \text{if } x_j \geq \ell_j \end{cases} \qquad (7.5.1)$$

To model this situation using the binary variables y_j correctly, it is necessary that x_j be bounded above in the problem; i.e., there must exist an upper bound such that x_j is $\leq$ it at all feasible solutions to the problem. Let α_j be such an upper bound for x_j at all feasible solutions to the problem. The constraint that:

x_j is either 0, or $\geq \ell_j$ is then equivalent to (7.5.2).

$$\begin{aligned} x_j - \ell_j y_j &\geq 0 \\ x_j - \alpha_j y_j &\leq 0 \\ y_j = 0 &\quad \text{or} \quad 1 \end{aligned} \qquad (7.5.2)$$

(7.5.2) represents through linear constraints the definition of the binary variable y_j associated with the two conditions on x_j as defined in (7.5.1).

Constraints like this can be introduced into the model for each such batch size restricted variable in the model. This transforms the model into an integer program.

As an example, suppose we have the constraints

$$\text{Either } x_1 = 0, \text{ or } x_1 \geq 10; \quad \text{and either } x_2 = 0, \text{ or } x_2 \geq 25 \tag{7.5.3}$$

in a linear programming model. Suppose 1000 is an upper bound for both x_1, x_2 among feasible solutions of this model. Then defining the binary variables y_1, y_2 corresponding to the two possibilities on x_1, x_2 respectively as in (7.5.1), we augment the following constraints to the LP model to make sure that (7.5.3) will hold.

$$\begin{array}{rcl} x_1 - 10y_1 & \geq & 0 \\ x_1 - 1000y_1 & \leq & 0 \\ x_2 - 25y_2 & \geq & 0 \\ x_2 - 1000y_2 & \leq & 0 \\ y_1, y_2 \quad \text{are both} & & 0 \text{ or } 1 \end{array}$$

Exercises

7.5.1: A problem has been modeled as the linear program

$$\begin{array}{rrcl} \text{Minimize} & z(x) = cx & & \\ \text{subject to} & Ax & = & b \\ & x & \geq & 0 \end{array}$$

The set of feasible solutions of this LP is known to be unbounded with every variable being unbounded above on it.

It has been realized that two constraints in the real problem have not been included in this LP model. They are: "x_1 should be either 0, or ≥ 10"; and "x_2 has to be either 0, or ≥ 50". Is it possible to model the whole problem as an MIP using the techniques discussed in this section? Why or why not?

Discuss how to solve this problem.

7.5.2: Formulate the following problem as an MIP.

$$\begin{array}{rl} \text{Minimize} \quad z = 7x_1 - 20x_2 - 35x_3 & \\ \text{subject to} \quad 2x_1 + 5x_2 - 7x_3 & \geq \ 100 \\ x_2 + 8x_3 & \geq \ 150 \\ 3x_1 + 8x_2 & \geq \ 200 \\ 8x_1 + 10x_3 & \geq \ 120 \\ 0 \leq x_j \leq 200 & \quad j = 1, 2, 3 \\ x_1 = 0 \text{ or } \geq 20 & \quad x_2 = 0 \text{ or } \geq 30 \end{array}$$

7.6 Other "Either, Or" Constraints

Let x be the column vector of decision variables in an LP, in which we have an additional constraint involving m conditions

$$\begin{array}{rcl} g_1(x) & \geq & 0 \\ & \vdots & \\ g_m(x) & \geq & 0 \end{array} \qquad (7.6.1)$$

where each of these conditions is a linear inequality. The additional constraint does not require that all the conditions in (7.6.1) must hold, but only specifies that at least k of the m conditions in (7.6.1) must hold. To model this requirement using linear constraints we define binary variables $y_1, \ldots, y_m$ with the following definitions.

$$y_i = \begin{cases} 0, & \text{if the condition } g_i(x) \geq 0 \text{ holds} \\ 1, & \text{otherwise} \end{cases} \qquad (7.6.2)$$

To model this situation correctly using these binary variables, it is necessary that each of these functions $g_i(x)$ be bounded below on the set of feasible solutions of the original LP model. Let L_i be a positive number such that $-L_i$ is a lower bound for $g_i(x)$ on the set of feasible solutions of the original LP model. Then the following system of constraints, augmented to the LP model, will guarantee that at least k of the conditions in (7.6.1) will hold.

$$\begin{aligned} g_1(x) + L_1 y_1 &\geq 0 \\ &\vdots \\ g_m(x) + L_m y_m &\geq 0 \\ y_1 + \ldots + y_m &\leq m - k \\ y_i = 0 \text{ or } 1 & \quad \text{for all } i \end{aligned} \tag{7.6.3}$$

In the same way, any restriction of the type that at least (or exactly, or at most) k conditions must hold in a given system of linear conditions, can be modeled using a system of linear constraints of the form (7.6.3) involving binary variables.

As an example, consider the system of linear constraints $0 \leq x_1 \leq 10$, $0 \leq x_2 \leq 10$, on the two variables x_1, x_2 in the two dimensional Cartesian plane. In addition, suppose we impose the constraint that "either $x_1 \leq 5$, or $x_2 \leq 5$" must hold. This constraint states that at least one of the following two conditions must hold.

$$\begin{aligned} g_1(x) = 5 - x_1 &\geq 0 \\ g_2(x) = 5 - x_2 &\geq 0 \end{aligned} \tag{7.6.4}$$

With this constraint, the set of feasible solutions of the combined system is the nonconvex dotted region in Figure 7.7.

A lower bound for both $g_1(x), g_2(x)$ in (7.6.4) in the original cube is -15. So, the constraint that at least one of the two constraints in (7.6.4) must hold is equivalent to the following system

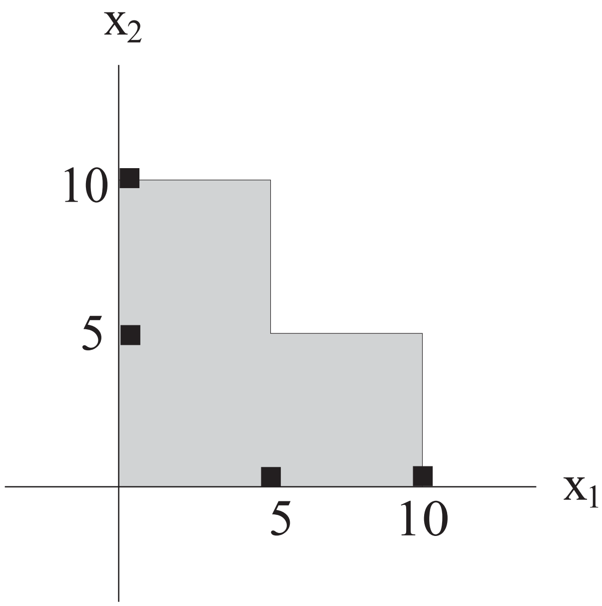

Figure 7.7:

$$
\begin{array}{lllll}
5 - x_1 & & +15y_1 & & \geq 0 \\
 & 5 - x_2 & & +15y_2 & \geq 0 \\
 & & y_1 & +y_2 & =1 \\
 & y_1, y_2 = 0 \text{ or } 1 & & &
\end{array}
\qquad (7.6.5)
$$

When the constraints in (7.6.5) are augmented to the constraints $0 \leq x_1 \leq 10$, $0 \leq x_2 \leq 10$ of the original cube, we get a system that represents the dotted region in the x_1, x_2-plane in Figure 7.7, using the binary variables y_1, y_2. Using similar arguments, sets that are not necessarily convex, but can be represented as the union of a finite number of convex polyhedra, can be represented as the set of feasible solutions of systems of linear constraints involving some binary variables.

Example 7.6.1: Consider the following integer programming model.

$$
\text{Minimize} \quad 6x_1 - 7x_2 + 8x_3
$$

$$\begin{array}{rcl} \text{subject to} \quad x_1 - x_2 + 2x_3 & \geq & 10 \\ 2x_1 + 3x_2 - x_3 & \geq & 15 \\ -8x_1 + 14x_2 + x_3 & \geq & 20 \\ 4x_1 + 2x_2 + 2x_3 & \geq & 20 \\ x_1, x_2 \geq 0, -5 \leq x_3 \leq 5 & & \text{all } x_j \text{ integer} \end{array}$$

Suppose it is required to impose an additional constraint that $x_3 \neq 0$ in this model. We will now show how this can be done.

Since x_3 is an integer bounded variable between -5 and $+5$, requiring that $x_3 \neq 0$ is equivalent to requiring that exactly one of the two constraints: $x_3 \leq -1$ or $x_3 \geq +1$ must hold. Define decision variables y_1, y_2 with the definitions:

$$y_1 = \begin{cases} 0, & \text{if } -x_3 + 1 \geq 0 \text{ holds} \\ 1, & \text{otherwise} \end{cases}$$

$$y_2 = \begin{cases} 0, & \text{if } x_3 + 1 \geq 0 \text{ holds} \\ 1, & \text{otherwise} \end{cases}$$

Then the combined model is

$$\begin{array}{rcl} \text{Minimize} \quad 6x_1 - 7x_2 + 8x_3 & & \\ \text{subject to} \quad x_1 - x_2 + 2x_3 & \geq & 10 \\ 2x_1 + 3x_2 - x_3 & \geq & 15 \\ -8x_1 + 14x_2 + x_3 & \geq & 20 \\ 4x_1 + 2x_2 + 2x_3 & \geq & 20 \\ -x_3 + 1 + 6y_1 & \geq & 0 \\ x_3 + 1 + 6y_2 & \geq & 0 \\ y_1 + y_2 & = & 1 \\ x_1, x_2 \geq 0, -5 \leq x_3 \leq 5 & & \text{all } x_j \text{ integer, } y_1, y_2 \text{ binary.} \end{array}$$

7.7 Indicator Variables

Indicator variables form a modeling tool for situations similar to those discussed in Sections 7.5, 7.6. They are binary variables that can be used in integer programming models to develop a system of linear constraints that will force the indicator variable to assume the value of 1 iff some property on the value of an affine function $f(x)$ (i.e., a linear function + a constant) holds. They are useful in modeling situations that call for enforcing some logical conditions.

Table 7.7.1

Property	Constraints that make definition of y true
y is a binary variable in all the following	
1. $f(x) \geq \alpha$	$f(x) - (U - \alpha + 1)y \leq \alpha - 1$ $f(x) - (\alpha - L)y \geq L$
2. $f(x) \leq \alpha$	$f(x) + (\alpha + 1 - L)y \geq \alpha + 1$ $f(x) + (U - L)y \leq \alpha + U - L$
3. $f(x) = \alpha$ where $L < \alpha < U$	$f(x) - (U - \alpha + 1)y_1 \leq \alpha - 1$ $f(x) - (\alpha - L)y_1 \geq L$ $f(x) + (\alpha + 1 - L)y_2 \geq \alpha + 1$ $f(x) + (U - L)y_2 \leq \alpha + U - L$ $y = y_1 + y_2 - 1$ y_1, y_2 binary
4.$f(x) = L$	$f(x) + y \geq L + 1$ $f(x) + (U - L)y \leq U$
5. $f(x) = U$	$f(x) - y \leq U - 1$ $f(x) - (U - L)y \geq L$
6. $f(x) \neq \alpha$	$f(x) - (U - \alpha + 1)y_1 \leq \alpha - 1$ $f(x) - (\alpha - L)y_1 \geq L$ $f(x) + (\alpha + 1 - L)y_2 \geq \alpha + 1$ $f(x) + (U - L)y_2 \leq \alpha + U - L$ $y = 2 - y_1 - y_2$ y_1, y_2 binary

We will denote the indicator variables by symbols y, y_1, or y_2. $f(x)$

is the affine function in variables $x = (x_1, \ldots, x_n)^T$ whose value we are concerned with.

An important thing to remember is that indicator variables are useful modeling tools in constructing integer programming models, so here we are assuming that $x_1, \ldots, x_n$ are either binary or integer variables in the problem and that the function $f(x)$ takes only integer values at all integer vectors x.

The techniques discussed in this section only work if $f(x)$ is bounded below and above over the feasible region in the x-space, we will denote these lower, uper bounds by L, U respectively (both L, U are integer).

We will now discuss various properties on the values of $f(x)$ and how to define indictor variables for them. The main indicator variable is the binary variable denoted by y, it must have the property that $y = 1$ iff the property holds, $y = 0$ otherwise. Here α is integer. Table 7.7.1 lists the properties, and the corresponding constraints

In understanding Property 3, remember that when $L < \alpha < U$, if $f(x) = \alpha$ does not hold, then either $f(x) \leq \alpha - 1$ or $f(x) \geq \alpha + 1$ must hold. Same way when $\alpha = L$ ($\alpha = U$), if $f(x) = \alpha$ does not hold, then $f(x) \geq L + 1$ ($f(x) \leq U - 1$) must hold.

We will now illustrate the use of indicator variables in modeling, with some examples taken from Chlond & Toase [2003]. The problems modeled are puzzle problems taken from [Smullyan 1978].

Example 7.7.1: A reporter is visiting a forest in which every human inhabitant is either an A or B. An A-person always tells the truth, a B-person always lies. In addition, some people also call themselves Cs, each C-person is of course either an A or B. The reporter meets 3 people P_1, P_2, P_3 exactly one of whom is a C. They make the following statements:

P_1: I am a C.

P_2: I am a C

P_3: At most one of us is an A.

We will now formulate the problem of classifying these 3 people as an integer feasibility problem. Define binary decision variables x_i

which takes value 1 if P_i is an A, 0 if P_i is a B; and y_i which takes the value 1 if P_i is a C, 0 otherwise; for $i = 1$ to 3.

Since only one person is a C-type, we have the constraint $y_1 + y_2 + y_3 = 1$.

If P_1 is an A-type, then $x_1 = 1$, and since A-type always tells the truth, from P_1's statement we have $y_1 = 1$. Thus $x_1 = 1$ implies $y_1 = 1$. Also, if P_1 is a B-type, then $x_1 = 0$; and since P_1's statement "I am a C" must be a lie in this case, we must have $y_1 = 0$. Thus $x_1 = 0$ implies $y_1 = 0$. From this analysis, P_1's statement implies the constraint $x_1 = y_1$. Exactly the same analysis on P_2's statement yields the constraint $x_2 = y_2$.

P_3's statement is true iff $x_1 + x_2 + x_3 \leq 1$, so this inequality implies that P_3's statement is true, which implies that P_3 must be an A-type, i.e., $x_3 = 1$. So, we conclude that $f(x) = x_1 + x_2 + x_3 \leq \alpha = 1$ implies $x_3 = 1$; i.e., x_3 is like an indicator variable for this condition. The minimum and maximum values for $f(x) = x_1 + x_2 + x_3$ are $L = 0$ and $U = 3$, since each x_i is a binary variable. So, by item 2 in Table 7.7.1, we have the constraints: $x_1 + x_2 + 3x_3 \geq 2$ and $x_1 + x_2 + 4x_3 \leq 4$.

Similarly, P_3's statement is a lie iff $x_1 + x_2 + x_3 \geq 2$, and in this case P_3 must be a B-type, i.e., $x_3 = 0$. Thus $x_1 + x_2 + x_3 \geq 2$ implies that $z_3 = 1 - x_3 = 1$, i.e., z_3 is like an indicator variable for this condition. So, by item 1 in the above table, we have the constraints: $x_1 + x_2 + 3x_3 \leq 3$, $x_1 + x_2 + 3x_3 \geq 2$.

Putting all the constraints together, we get the system:

$$
\begin{array}{cccccccc}
y_1 & +y_2 & +y_3 & & & & = & 1 \\
y_1 & & & -x_1 & & & = & 0 \\
 & y_2 & & & -x_2 & & = & 0 \\
 & & & x_1 & +x_2 & +3x_3 & \geq & 2 \\
 & & & x_1 & +x_2 & +4x_3 & \leq & 4 \\
 & & & x_1 & +x_2 & +3x_3 & \leq & 3
\end{array}
$$

All variables are binary.

To solve this system by total enumeration, we first try $x_1 = 1$, and verify that this leads to inconsistency. So does $x_2 = 1$. Trying $x_3 = 1$, it leads to the solution $x = (0,0,1)^T, y = (0,0,1)^T$, the unique solution of the system; i.e., P_1, P_2 are B-type, and P_3 is a C-type and also an A-type.

Example 7.7.2: The setting of this example is exactly the same as that in Example 7.7.1, but this time the reporter is given the information that among P_1, P_2, P_3 there is exactly one C-type and that this person is also A-type; and he gets only two statements:

P_1: At least one of the three of us is a B.

P_2: P_3 is an A

The problem is to find who is the C-type. To model this problem, we define the same decision variables as in Example 7.7.1.

From the information given there is only one C-type person, this leads to the constraint $y_1 + y_2 + y_3 = 1$. The fact that the C-type person is an A, along with the setting of this problem can be modeled into the constraint $x_i \geq y_i$ for $i = 1, 2, 3$.

$$\begin{array}{cccccccc}
y_1 & +y_2 & +y_3 & & & & = & 1 \\
 & & & & -x_2 & +x_3 & = & 0 \\
 & & & 4x_1 & +x_2 & +x_3 & \geq & 3 \\
 & & & 4x_1 & +x_2 & +x_3 & \leq & 5 \\
y_1 & & & -x_1 & & & \leq & 0 \\
 & y_2 & & & -x_2 & & \leq & 0 \\
 & & y_3 & & & -x_3 & \leq & 0
\end{array}$$

All variables are binary.

We verify that P_1's statement cannot be a lie, it must be true, so P_1 is an A-type; i.e., this statement, which says that $x_1+x_2+x_3 \leq 2$ implies that $x_1 = 1$. So, x_1 is like an indicator variable for this condition; which leads to the constraints: $4x_1 + x_2 + x_3 \geq 3$, $4x_1 + x_2 + x_3 \leq 5$.

If P_2's statement ($x_3 = 1$) is true, P_2 must be an A; i.e., $x_3 = 1$ implies $x_2 = 1$. Similarly if P_2's statement is false ($x_3 = 0$), P_2 must be a B, or x_2 must be 0. So, this statement leads to the constraint $x_3 = x_2$.

Putting all the constraints together we get the system given above, which is an integer feasibility model for this problem.

We leave it to the reader to solve this system by total enumeration.

Exercises

7.7.1 Lady or Tigers-2: (Adopted from R. Smullyan, 1982). In trying to win the hand of his beloved, a man becomes a prisoner, and is faced with a decision where he must open the door of one of nine rooms. Each room may be either empty, or hiding a tiger, or his beloved. Each door has a sign bearing a statement. The statements on the nine doors are:

Door 1: The lady is in an odd-numbered room.

Door 2: This room is empty.

Door 3: Either the sign on door 5 is right, or that on sign 7 is wrong.

Door 4: The sign on door 1 is wrong.

Door 5: Either the sign on door 2, or that on door 4 is right.

Door 6: The sign on door 3 is wrong.

Door 7: The lady is not in room 1.

Door 8: This room contains a tiger and room 9 is empty.

Door 9: This room contains a tiger, and the sign on door 6 is wrong.

The prisoner is given several clues: He is informed that the lady is in one of the rooms. The sign on the door hiding the lady is true, the signs on all doors hiding tigers are false, and the signs on the doors of empty rooms can be either true or false.

If the prisoner opens the door to find the lady, he can marry her; if he opens the door to find a tiger he will be eaten alive; and if he opens an empty room, he will be banished alone to a distant foreign land.

(i): Help the prisoner by formulating the problem of finding all the doors which may be hiding his beloved, subject to all the clues.

(ii): It turns out that the system does not have a unique solution. The prisoner asks for a decent clue, whether room 8 is empty or not. If he was told that room 8 was empty, it would have been impossible for him to have found the lady.

He was told that room 8 was not empty. With this additional clue, the prisoner was able to uniquely identify the right, wrong status of all the signs, and identify the unique room containing the lady. Find it.

7.7.2: (Adopted from R. Smullyan, 1982). Three robbers, let us call them A, B, and C stole a horse, a mule, and a camel; each one stealing one animal. They were finally caught, but it was not known which thief stole what. At the trial they made the following statements.

A: The horse was stolen by B.

C: Not so, B stole the mule.

B: Those are both lies, I didn't steal either.

The clues are: the one who stole the camel was lying, and the one who stole the horse was telling the truth. Required to find out who stole which animal. Formulate as a $0-1$ integer feasibility system.

7.7.3: (Adopted from R. Smullyan, 1997). On the 1009th night the King tells his wife "tonight I am in a mood for some logic puzzles". "Very well" said she, and continues "This is one from a curious town in Persia where every inhabitant is either an M or an A. All Ms always tell the truth, they never lie; and all As never tell the truth, they always lie. A reporter came across a group of 10 inhabitants, call them I_1 to I_{10}, and asks them how many of them are Ms and how many As. The statements made by them are:

I_i: Exactly i of us are As;   for $i = 1$ to 9.

I_{10}: All 10 of us are As.

It is required to find out for each $i = 1$ to 10, whether I_i is an M or an A. Formulate this as a $0-1$ integer feasbility problem. Show that system has a unique solution, and find that solution.

7.8 Discrete Valued Variables

Consider an LP model with the additional requirement that some variables can only lie in specified discrete sets. For example, in the problem of designing a water distribution system, one of the variables is the diameter of the pipe in inches; this variable must lie in the set {6, 8, 10, 12, 16, 20, 24, 30, 36} because the pipe is available only in these diameters.

In general, if x_1 is a decision variable that is restricted to assume values in the discrete set $\{\alpha_1, \ldots, \alpha_k\}$, this constraint is equivalent to

$$\begin{aligned} x_1 - (\alpha_1 y_1 + \ldots + \alpha_k y_k) &= 0 \\ y_1 + \ldots + y_k &= 1 \\ y_j = \ \ 0 \text{ or } 1 \text{ for each } j & \end{aligned}$$

Such constraints can be augmented to the other linear equality and inequality constraints in the model for each discrete valued variable. This transforms the problem into an integer program.

7.8.1: A bank van had several bags of coins, each containing either 16, 17, 23, 24, 39, or 40 coins. While the van was parked on the street, thieves stole some bags. A total of 100 coins were lost. It is required to find how many bags were stolen. Formulate this as a discrete variable feasibility model, and also transform this into a 0−1 model.

7.8.2: A merchant has bags of emeralds (nine to a bag), and rubies (four to a bag). He has a total of 59 jewels. Required to find how many of his jewels are rubies. Formulate as a 0−1 integer feasibility system.

7.9 The Graph Coloring Problem

A **graph** G is a network $(\mathcal{N}, \mathcal{A})$ where $\mathcal{N}$ is a set of nodes, and $\mathcal{A}$ is a set of edges, each edge joining a distinct pair of nodes. When the picture of a graph is drawn on paper for illustrative purposes, it is possible that some edges drawn intersect at a point on paper which is not a node in the network.

When $\mathcal{N}$ has n nodes, we will usually number them and denote them by their numbers 1,..., n and refer to them by their numbers. The edge joining nodes i, and j will be denoted by the symbol $(i; j)$ (or (i, j) in some books). Nodes i, and j are said to be **adjacent** in the graph, if there is an edge joining them.

Computer programs usually store the graph G by storing the set $\mathcal{N} = \{1, \ldots, n\}$ and the set $\mathcal{A}$ as a set of pairs of nodes $(i; j)$. Graph theory is the branch of mathematics dealing with the mathematical study of graphs and their properties.

A graph G is said to be a **planar graph** if it is possible to represent the nodes in it as points on a piece of paper, and draw all the edges in such a way that every pair of edges without a common node never intersect in the picture. A nonplanar graph is one that is not planar.

An optimization problem in graphs that has many applications and has been investigated extensively is the **graph coloring** problem: given a graph $G = (\mathcal{N}, \mathcal{A})$, find a coloring of its nodes with each node colored by a single color, satisfying the constraint that for each edge $(i; j) \in \mathcal{A}$, the colors used for coloring i and j are different; using the smallest possible number of colors. It is a **combinatorial optimization problem**.

As an example consider the graph with 5 nodes on the left side of Figure 7.8.

Suppose we use color 1 to color node 1. Since nodes 2, 5, 4 are all adjacent to node 1, none of them can be colored with color 1; but node 3 which is not adjacent to node 1 can be colored with color 1. Since node 5 is adjacent to node 3, we cannot also use color 2 to color node 5; so suppose we color node 5 with color 3. We can color node 4 with color 2. The colors used are shown in Figure 7.8, on the right side. This coloring satisfies all the constraints, and clearly it uses the smallest number of colors, 3.

History of the Graph Coloring Problem

Interest in the graph coloring problem was originally ignited by a

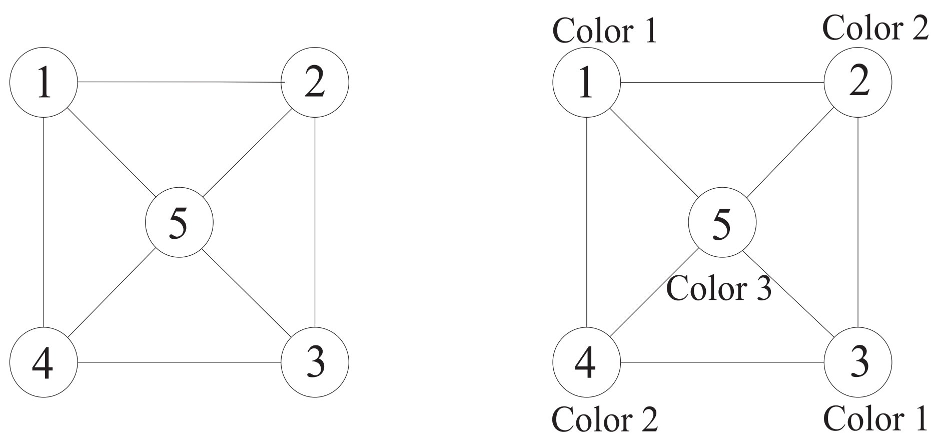

Figure 7.8: Graph to be colored is on the left, the coloring obtained is shown on the right.

highly publicized mathematics problem known as the **four color problem**. Dating back to 1852, credit for making the four color problem popular belongs to a British mapmaker, Francis Guthrie.

The basic rule for coloring a map is that no two regions sharing a common boundary line of positive length can be colored with the same color, to avoid ambiguity. If two regions meet at only a single point, and do not have a common boundary line, it is okay for them to be colored with the same color. If you look at an atlas, you can verify that this is how all familiar maps are colored.

In the trade, mapmakers have known for a long time that if you plan well enough, you will never need more than four colors to color any map. No mapmaker has ever stumbled upon a map that required the use of five colors. This puzzled Francis Guthrie, so in 1852 he dashed off a letter to his brother Frederick Guthrie who is an academic, enquiring whether he can explain the reasons for the same. Frederick showed the letter to his advisor, the mathematics professor DeMorgan. This brought the question to the attention of the European mathematics community at that time, who perceived it as an interesting but surprisingly difficult problem to solve.

Represent each region in the map by a node in a graph. Join a pair or nodes by an edge if the corresponding regions have a common boundary line of positive length. The resulting graph G will be a planar graph. In terms of graph coloring, the problem posed by Francis Guthrie is to prove that one needs no more than four colors to color any planar graph. Over the years it has become known as the **four color conjecture**. Here is a map and the corresponding graph that needs actually four colors.

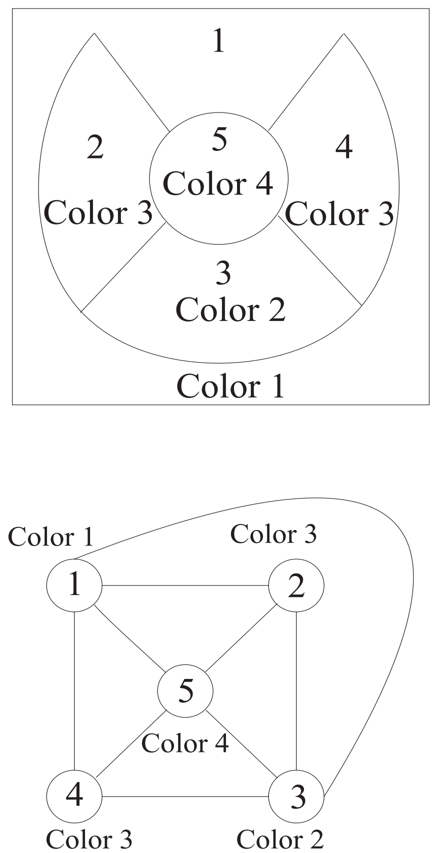

Figure 7.9: A map with five regions numbered 1 to 5. Its planar graph representation is shown. A feasible coloring with four colors is shown.

Shortly after the problem became known, a simple and fairly short proof has been constructed to show that actually five colrs are sufficient to color any planar graph, this has become famous as the "five color theorem", but the goal of showing that four will be sufficient turned out to be elusive. Until the 1970s, several claimed proofs of the four color conjecture have been published, but their incorrectness was recognized after a few years. But these failed proofs did have some value, ideas in them were used by other mathematicians to make various forms of progress on the four color problem.

Finally in 1976 Kenneth Appel and Wolfgang Haken proved the four color conjecture using a computer program that is thousands of lines long and took over 1200 hours to run. Since then the four color conjecture has been referred to as the "four color theorem". A simpler, human-verifiable, but still quite long and complicated proof was developed in 1997 by N. Robertson, D. P. Sanders, P. D. Seymour, and R. Thomas, see the website: http://www.math.gatech.edu/˜ thomas/FC/fourcolor.html for details.

An Integer Programming Formulation of the Graph Coloring Problem

The graph coloring problem can be modeled as an integer program. Let $G = (\mathcal{N}, \mathcal{A})$ be the graph with n nodes that we want to color. Clearly no more than n colors are needed to color G, our aim is to find a coloring using the smallest number of colors. For $i = 1$ to n, define the decision variable

$$x_i = \text{number of the color used to color node } i$$

Then x_i is an integer variable taking values between 1 to n. So, for any pair of nodes i, j, $x_i - x_j$ varies between $-n$ to $+n$; and we require it to be different from 0 if $(i; j)$ is an edge of G. So, here is the integer programming formulation for the coloring of G (here y_{ij1}, y_{ij2} are indicator variables to require that either $x_i - x_j \leq -1$, or $x_i - x_j \geq +1$ for each edge $(i; j)$ in G):

$$\begin{array}{lrcll} & \text{Minimize} \quad \theta & & & \\ \text{subject to} & x_i - x_j - 1 + 2ny_{ij1} & \geq & 0 & \text{for each } \ (i;j) \in \mathcal{A} \\ & -x_i + x_j - 1 + 2ny_{ij2} & \geq & 0 & \text{for each } \ (i;j) \in \mathcal{A} \end{array}$$

$$\begin{aligned} y_{ij1} + y_{ij2} &= 1 \quad \text{for each} \quad (i;j) \in \mathcal{A} \\ 1 \leq x_i \leq n & \quad \text{for all } i \\ \theta \geq x_i & \quad \text{for all } i \\ x_i & \quad \text{integer for all } i, \\ y_{ij1}, y_{ij2} & \quad \text{binary for all} \quad (i;j) \in \mathcal{A} \end{aligned}$$

From the constraints it is clear that $\theta \geq$maximum$\{x_1, \ldots, x_n\}$; and since θ is being minimized, in the optimum solution θ will be equal to maximum$\{x_1, \ldots, x_n\}$, and consequently equal to the number of colors used. So the optimum solution of this model will give a coloring of G using the smallest number of colors.

But no one really uses an integer programming model for solving graph coloring problems, because even with the best algorithms available today, these models take very long to reach optimality. Several efficient and fast heuristic methods have been developed for graph coloring that yield results very close to the optimum. Practitioners invariably use these heuristic methods. We discuss some of these heuristic methods in Chapter 9.

An Application of the Graph Coloring Problem

The graph coloring problem appears in many applications. One such is as a direct model for the meeting scheduling problem discussed in Section 7.3. There we modeled the problem of determining the maximum number of meetings that can be scheduled without conflicts in a given time period. Now we show that the problem of determining the smallest number of time slots needed for scheduling all the meetings without any conflicts can be modeled and solved as a graph coloring problem.

Represent each meeting by a node in a graph $G = (\mathcal{N}, \mathcal{A})$. $\mathcal{N}$ is the set of meeting nodes. Join the two nodes i, and j by an edge $(i;j)$ iff meetings i and j both require the attendance of a common administrator. $\mathcal{A}$ is the set of edges obtained this way.

Thus if two meetings do not require a common administrator to attend them, there won't be an edge connecting their nodes in G.

The time slot into which a meeting is scheduled can play the role of the color for coloring its node. Clearly the scheduling conflict constraint is exactly the coloring constraint for coloring G. So under this representation, the meeting scheduling problem is exactly the coloring problem for G.

Find the coloring of nodes of G using the smallest number of colors. Its optimum value is the smallest number of time slots required for the meeting scheduling problem. An optimum schedule is to schedule all meetings corresponding to nodes of the same color in one time slot.

For our project at the UM Hospital, we used a graph coloring heuristic to solve the problem. It produced very satisfactory results.

Example 7.9.1:

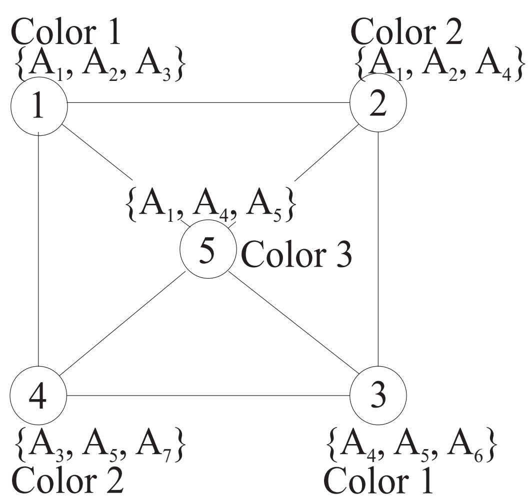

Figure 7.10: A graph coloring formulation of the meeting scheduling problem. Each node represents a meeting, we show the set of administrators who should attend it by its side. The color gives the time slot number for scheduling the meeting.

Consider a meeting scheduling problem for 7 administrators numbered A_1 to A_7, involving 5 different meetings numbered M_1 to M_5; with the required sets of administrators to attend each meeting as shown in the following table.

Meeting	Administrators required to attend
M_1	A_1, A_2, A_3
M_2	A_1, A_2, A_4
M_3	A_4, A_5, A_6
M_4	A_3, A_5, A_7
M_5	A_1, A_4, A_5

The graph representation of this problem is shown in Figure 7.10. Node i represents meeting M_i for $i = 1$ to 5, and by the side of each meeting node, the set of administrators required to attend it is shown. Nodes 1, 2 are joined by an edge in the graph because the sets of administrators required to attend these meetings has A_1 (and also A_2) in common. Nodes 1, 3 do not have an edge joining them because the sets of administrators required to attend them are disjoint.

An optimal coloring for the graph is shown in the figure, it requires 3 colors. So, 3 time slots are needed to schedule the meetings without conflicts. Letting color k represent time slot k, here is an optimum schedule.

Time slot	Meetings scheduled
1	M_1, M_3
2	M_2, M_4
3	M_5

Verify that any two meetings scheduled in the same time slot do not require a common administrator to attend them, so they can take place simultaneously without creating any conflicts.

Exercises

7.9.1: Model the meeting scheduling problem discussed in Example 7.3.5 as a graph coloring problem.

7.10 The Traveling Salesman Problem (TSP)

A salesperson has to visit cities $2, \ldots, n$, and his/her trip begins at, and must end in, city 1. c_{ij} = the cost of traveling from city i to city j, is given for all $i \neq j = 1$ to n, and $c = (c_{ij})$ of order $n \times n$ is known as the **cost matrix** for the problem. Beginning in city 1, the trip must visit each of the cities $2, \ldots, n$ once and only once in some order, and must return to city 1 at the end. The cost matrix is the input data for the problem. The problem is to determine an optimal order for traveling the cities so that the total cost is minimized.

This is a classic combinatorial optimization problem that has been the object of very intense research since the late 1950s.

If the salesperson travels to the cities in the order i to $i+1$, $i = 1$ to $n-1$, and then from city n to city 1, this route can be represented by the order "1, 2, $\ldots, n; 1$". Such an order is known as a **tour** or a **hamiltonian**.

From the initial city 1 the salesperson can go to any of the other $n-1$ cities. So, there are $n-1$ different possibilities for selecting the first city to travel from the initial city 1. From that city the salesperson can travel to any of the remaining $n-2$ cities, etc. Thus the total number of possible tours in a n city traveling salesman problem is $(n-1)(n-2)\ldots 1 = (n-1)!$. This number grows explosively as n increases.

Let $\mathcal{N} = \{1, \ldots, n\}$ be the set of cities under consideration. Let $\mathcal{N}_1$ be a proper subset of $\mathcal{N}$. A tour covering the cities in $\mathcal{N}_1$ only, without touching any of the cities in $\mathcal{N} \backslash \mathcal{N}_1$ is known as a **subtour** covering or spanning the subset of cities $\mathcal{N}_1$. See Figure 7.11.

We will use the symbol τ to denote tours. Given a tour τ, define binary variables x_{ij} by

$$x_{ij} = \begin{cases} 1, & \text{if the salesperson goes from city } i \text{ to city } j \text{ in tour } \tau \\ 0, & \text{otherwise} \end{cases}$$

Then the $n \times n$ matrix $x = (x_{ij})$ is obviously an assignment, i.e., a feasible solution of an assignment problem (3.8.1), (3.8.2) of Section 3.8; it is the assignment corresponding to the tour τ. Hence every tour corresponds to an assignment. For example, the assignment corresponding to the tour 1, 5, 2, 6, 3, 4; 1 in Figure 7.11 follows the figure.

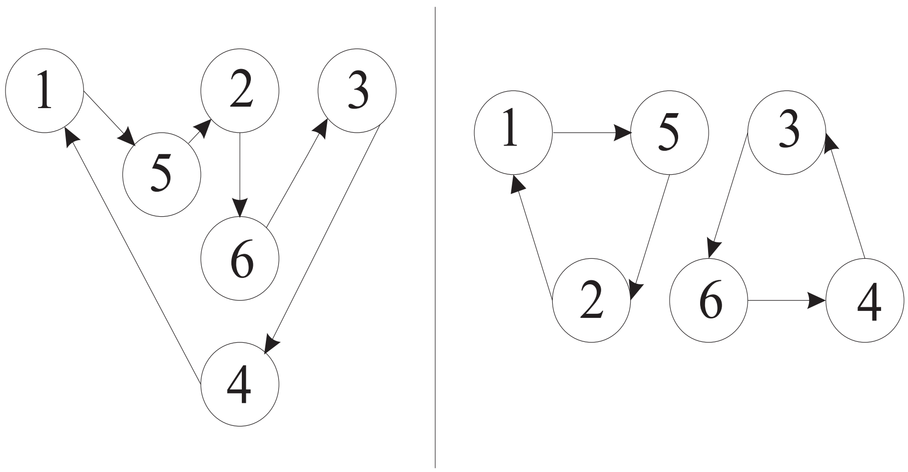

Figure 7.11: Each node represents a city. On the left is the tour 1, 5, 2, 6, 3, 4; 1. On the right we have two subtours 1, 5, 2; 1 and 3, 6, 4; 3.

to	j=1	2	3	4	5	6
from i = 1					1	
2						1
3				1		
4	1					
5		1				
6			1			

where the blank entries in the matrix are all zeros. In the notation of Chapter 5, this is the assignment {(1,5), (5,2), (2,6), (6,3), (3,4), (4,1)}. An assignment is called a **tour assignment** iff it corresponds to a tour.

As an example, the assignment {(1,5), (5,2), (2,1), (3,6), (6,4), (4,3)} is not a tour assignment since it represents the two subtours on the right hand side of Figure 7.11.

So, the **traveling salesman problem** (usually abbreviated as **TSP**) of order n with the cost matrix $c = (c_{ij})$ is

$$
\begin{array}{rcll}
\text{Minimize } z_c(x) & = & \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij} & \\
\text{Subject to } \sum_{j=1}^{n} x_{ij} & = & 1 \text{ for } i = 1 \text{ to } n & \qquad (7.10.1) \\
\sum_{i=1}^{n} x_{ij} & = & 1 \text{ for } j = 1 \text{ to } n & \\
x_{ij} & = & 0 \text{ or } 1 \text{ for all } i, j & \\
\text{and } x = (x_{ij}) & & \text{is a tour assignment} &
\end{array}
$$

Since the salesperson is always going from a city to a different city, the variables x_{ii} will always be 0 in every tour assignment. To make sure that all the variables x_{ii} will be 0, we define the cost coefficients c_{ii} to be equal to a very large positive number for all $i = 1$ to n in (7.10.1)

There are $n!$ assignments of order n. Of these, only $(n-1)!$ are tour assignments. The last constraint that $x = (x_{ij})$ must be a tour assignment makes this a hard problem to solve.

Notice that the formulation given in (7.10.1) for the TSP is not an integer programming model, since the last constraint in it is not a linear constraint. There are several ways for modeling the TSP as an integer program, but the model (7.10.1) with the nonlinear constraint that x be a tour assignment turns out to be the most useful to develop approaches to solve the TSP. So, we will not discuss those integer programming models for the TSP in this book.

7.11 Exercises

7.1: A Word Puzzle: Following is a list of 47 words each having three letters (these may not be words in the English language)

ADV	AFT	BET	BKS	CCW	CIR	DER	DIP	EAT	EGO
FAR	FIN	GHQ	GOO	HAT	HOI	HUG	ION	IVE	JCS
JOE	KEN	LKK	LIP	LYE	MOL	MTG	NES	NTH	OIL
OSF	PIP	PRF	QMG	QUE	ROE	RUG	STG	SIP	TUE
TVA	UTE	VIP	WHO	XIN	YES	ZIP			

Each letter corresponds to a unique numerical value, these letter values are: A = 1, B = 2, ..., Z = 26, in the usual order.

It is required to select a subset of 8 words from the list given above to satisfy the following: let s_t = sum of the letter values of the tth letter in the selected words, $t = 1, 2, 3$. Then s_1 must be less than both s_2 and s_3. The selection should maximize $s_1 + s_2 + s_3$ subject to these constraints.

(i) Formulate this problem and solve it using an integer programming software package.

(ii) If there are ties for the optimum solution of the above problem, find a solution that maximizes s_1 among those tied.

(iii) If there are still ties, it is required to find the solution that maximizes s_2 among those that tie for (ii). Give an argument to show that the solution found in (ii) meets this requirement.

([G. Weber, 1990]).

7.2: It is required to assign distinct values 1 through 9 to the letters A, E, F, H, O, P, R, S, T, to satisfy the conditions and achieve the objective mentioned below. Two groups of 6 words each are given below.

Group 1	Group 2
AREA	ERST
FORT	FOOT
HOPE	HEAT
SPAR	PAST
THAT	PROF
TREE	STOP

Let s_1, s_2 refer to the totals for groups 1, 2 using the letter values assigned.

(i) The letter value assignment should minimize $s_1 - s_2$. Formulate this problem and find an optimum solution for it. Use the decision variables

$$x_{ij} = \begin{cases} 1 & \text{if } i\text{th letter is given value } j \\ 0 & \text{otherwise} \end{cases}$$

(ii) It is required to find a letter-value assignment that maximizes s_1 subject to the constraint that $s_1 - s_2 = 0$. Formulate this problem and find an optimum solution for it using an integer programming software package.

([G. Weber, 1990]).

7.3: Round table conference of European Foreign Ministers: The foreign ministers of European countries (east and west) numbered 1 to 10 are planning a round table conference. In Figure 7.12 each minister is represented by a node, and a line joins two nodes if the corresponding ministers speak a common language. If there is no line joining two nodes, it means that the corresponding ministers cannot speak with each other without an interpreter.

An ideal seating arrangement around the table is one in which every pair of ministers occupying adjacent seats both know a common language. If an ideal one does not exist, an optimal arrangement should minimize the number of pairs of adjacent ministers who cannot speak with each other. Formulate the problem of finding an ideal or optimal seating arrangement as a traveling salesman problem, and try to get a good solution for it by trial and error.

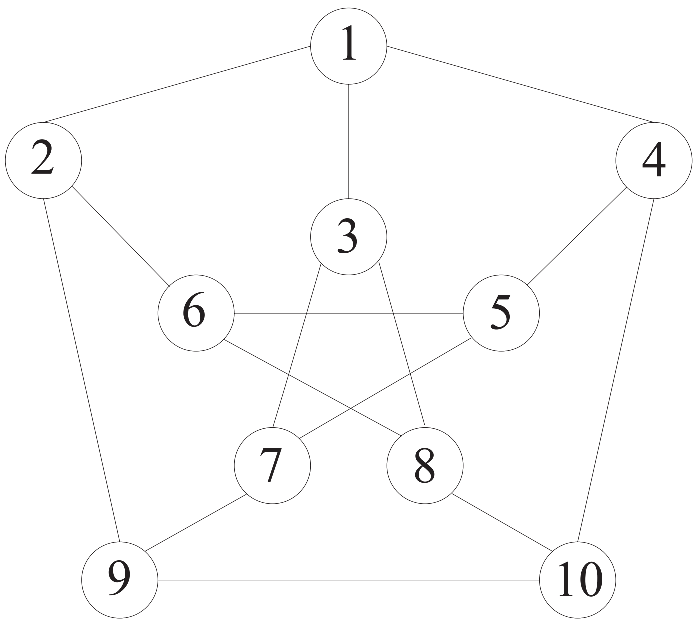

Figure 7.12:

7.4: Ms Skorean's parties in Washington D.C.: Ms. Skorean is famous for her parties in the Washington, D.C. area. In each four-year administration she throws k parties, sets m tables at each party and seats n people at each table. She makes a list of mn influential people at the start of the four year period and invites them to every party. Her tables are all distinct from each other (teak, oak, cherry etc.) and $k \leq m$. It is rumored that her parties are famous because of her clever seating arrangements. She makes sure that each of her guests sits at each table at most once. And she believes very strongly that the atmosphere remains lively if each guest meets different people at his/her table at each party. So, every time any pair of guests find themselves at the same table after the first time, she awards herself an imaginary penalty of c units, where $c > 0$. Given k, m, n (all positive integers ≥ 2 and $k \leq m$). Formulate the problem of determining the seating arrangement of her guests at the tables at the various parties, subject to the constraint that no guest sits at a table more than once and the overall penalty is as small as possible, as an integer program.

If $k = m$, what conditions should m, n satisfy to guarantee that a zero penalty seating arrangement exists?

When $k = m = n = 2$, show that there is no zero penalty seating arrangement using your formulation above. (Vishwas Bawle).

7.5: Subset-Sum Problem: We are given n positive integers $w_1, \ldots, w_n$; and another positive integer, w, called the **goal**. It is required to find a subset of $\{w_1, \ldots, w_n\}$ such that the sum of the elements in the subset is closest to the goal, without exceeding it. Formulate this as a special case of the knapsack problem. When the set of integers is {80, 66, 23, 17, 19, 9, 21, 32}, and the goal is 142, write this formulation, and see if you can solve it.

7.6: Bin Packing Problem n objects are given, with the ith object having the positive integer weight w_i kg., $i = 1$ to n. The objects need to be packed in bins, all of which are identical, and can hold any subset of objects as long as their total weight is $\leq w$ kg., which is the positive integral weight capacity of each bin. Assume that $w \geq w_i$ for each $i = 1$ to n. Objects cannot be split, and each must be packed whole into a bin. Formulate the problem of determining the minimum number of bins required for this packing, subject to the bin's weight holding capacity, as an integer program. Write this formulation when there are 8 objects with weights 80, 66, 23, 17, 19, 9, 21, 32 kg, respectively, and the bins weight holding capacity is 93 kg. See if you can solve it.

7.7: A company can form 6 different teams using 10 experts it has available. In the following table if there is an entry of 1 in the first line in the row of team i and the column of expert j, then expert j has to be on team i if it is formed (in this case, the amount in \$ that he/she is to be paid for being a member of this team, r_{ij}, is given just below this

1); he/she need not be included on this team otherwise. In addition to the team membership fee r_{ij}, if expert j is included in one or more teams that are formed, he/she has to be paid a *retainer fee* of $\$c_j$ given in the last row of the table. If team i is formed, the company derives a gross profit of $\$d_i$ given in the last column of the table. Also, because of other work commitments, company rules stipulate that no expert can work on more than two teams. It is required to determine which teams should be formed to maximize company's total net profit (gross profit from the teams formed, minus the retainer and team membership fees paid to the experts). Formulate this as an integer program. (T. Ramesh).

Team	Expert										d_i
	1	2	3	4	5	6	7	8	9	10	
1	1			1	1			1			10,000
	200			200	300			200			
2		1	1				1			1	15,000
		200	400				200			300	
3	1					1		1	1		6,000
	300					200		250	150		
4		1	1		1					1	8,000
		200	400		250					200	
5	1					1	1		1		12,000
	200					200	150		150		
6		1		1				1		1	9,000
		200		200				150		200	
c_j	800	500	600	700	800	600	400	500	400	500	

c_j = Retainer fees, d_i = gross profit

7.8: Balancing with available weights: There is an object whose weight is w kg. There are n types of stones, each stone of type i weighs exactly a_i kg., for $i = 1$ to n; and an unlimited number of copies of each are available. $w; a_1, \ldots, a_n$ are given positive integers. The object is placed in the right pan of a balance. It is required to place stones in the right and/or left pans of the balance so that it becomes perfectly balanced. It is required to do this using the smallest possible number of stones. Formulate this as an integer program. Write this formulation for the instance in which $n = 5, w = 3437$, and a_1 to a_5 are 1, 5, 15, 25, 57, 117, respectively.

7.9:Assembly line balancing: An assembly line is being designed for manufacturing a discrete part. There are 7 operations numbered 1 to 7 to be performed on each part. Each operation can be started on a part any time after all its immediate predecessor operations given in the following table are completed, but not before. The table also gives the time it takes an operator to carry out each operation, in seconds. The cycle time of the assembly line will be 20 seconds (i.e., each operator will have up to 20 seconds to work on a part before it has to be put back on the line). An operator on the assembly line can be assigned to carry out any subset of operations, as long as the work can be completed within the cycle time and the assignments do not violate the precedence constraints among the operations. It is required to determine the assignment of operations to operators on this line, so as to minimize the number of operators needed. Formulate this as an integer program.

Operation	Immediate predecessors	Time in secs.
1		7
2		9
3	1,2	6
4	1	4
5	3	8
6	4	7
7	5,6	5

7.10: 5000 acceptable units of a discrete product need to be manufactured in one day. There are 4 machines which can make this product, but the production rates, costs, and percentage defectives produced vary from machine to machine. Data is given below. Formulate a production plan to meet the demand at minimum cost.

Machine	Setup cost	Prod. cost/unit after mc. setup	Max. daily production	Expected defective %
1	400	4	2000	10
2	1000	6	4000	5
3	600	2	1000	15
4	300	5	3000	8

7.11: There are 5 locations where oil wells need to be drilled. There are two platforms from each of which any of these wells can be drilled. If a platform is to be used for drilling one or more of these wells, it needs to be prepared, the cost of which is given below. Once a platform is prepared, the drilling cost of each well depends on the drilling angle and other considerations. All the costs are in coded money units. Formulate the problem of determining which platform to use for drilling each well to complete all drilling at minimum total cost.

Platform	Preparation	Cost of drilling to location				
	cost	1	2	3	4	5
1	15	10	8	30	15	19
2	20	14	25	25	15	16

7.12: A company has 5 projects under consideration for carrying

Project	Required expenditure in			Expected annual yield
	Year 1	Year 2	Year 3	after construction
1	10	5	15	3
2	5	5	11	2
3	15	20	25	10
4	20	10	5	7
5	10	8	6	5
Available funds	50	40	50	

out over the next 3 years. If selected, each project requires a certain level of investment in each year over the 3 year period, and will result in an expected yield annually after this 3 year construction period. In the data given below, expenditures and income are in units of \$10,000. Formulate the problem of selecting projects to carry out, to maximize total return.

7.13: There are 5 objects which can be loaded on a vessel. The weight w_i (in tons), volume v_i (in ft^3), and value r_i (in units of \$1000) per unit of object i is given below for $i = 1$ to 5. Only 4 copies of object 1 and 5 copies of object 2 are available; but the other objects 3, 4, 5 are available in unlimited number of copies. For each object, the number of copies loaded has to be a nonnegative integer. The vessel can take a maximum cargo load of $w = 112$ tons, and has a maximum space of $v = 109$ ft^3. Formulate the problem of maximizing the value of cargo loaded subject to all the constraints.

Object i	w_i	v_i	r_i
1	5	1	4
2	8	8	7
3	3	6	6
4	2	5	5
5	7	4	4

7.14: The 800-telephone service of an airline operates round the clock. Data on

Period	Time of day	Min. operators needed
1	3 AM to 7 AM	26
2	7 AM to 11 AM	52
3	11 AM to 3 PM	86
4	3PM to 7 PM	120
5	7 PM to 11 PM	75
6	11 PM to 3 AM	35

the estimated number of operators needed during the various periods in a day, for attending to most of the calls in a satisfactory manner, is given above. Assume that each operator works for a consecutive period of 8 hours, but they can start work at the beginning of any of the 6 periods. Let x_t denote the number of operators starting work at the beginning of period t, $t = 1$ to 6. Formulate the problem of finding the optimum values for x_t, to meet the requirements in all the periods by employing the least number of operators.

7.15: The public works division in a region has the responsibility to subcontract work to private contractors. The work is of several types, and is carried out by teams, each of which is capable of doing all types of work.

The region is divided into 16 districts, and estimates of the amount of work to be done in each district are available.

There are 28 contractors of which the first 10 are experienced contractors. Data on the following quantities is provided.

i	=	1 to 10 are indices representing experienced contractors; $i =$ 11 to 28 are indices representing other contractors.
j	=	1 to 16 are indices representing the various districts.
a_i	=	number of teams contractor i can provide.
b_j	=	number of teams required by district j.
e_j	=	2 or 3, is the specified minimum number of contractors allotted to district j, this is to prevent overdependence on any one contractor.
c_{ij}	=	expected yearly cost of a team from contractor i allotted to district j.

At least one experienced contractor must be appointed in each district, a precaution in case some difficult work arises. Enough teams must be allotted to meet the estimated demand in each district, and no contractor can be asked to provide more teams than it has available. Formulate the problem of determining the number of teams from each contractor to allot to each district, so as to satisfy the above constraints at minimum cost.

Also, develop a heuristic method to find a reasonably good solution to this problem. ([M. Cheshire, K. I. M. McKinnon, and H. P. Williams, August 1984]).

7.16: Equitable Distribution of Assets There are n assets with the value of the ith asset being \$$a_i$ for $i = 1$ to n. $A = \sum_{i=1}^{n} a_i$ is the total value of all the assets. It is required to allot these assets to two beneficiaries in an equitable manner. Assets are indivisible, i.e., each asset has to be given completely to one beneficiary or the other, but cannot be split. Let A_1, A_2 be the total value of assets allotted to beneficiary 1, 2 respectively. It is required to distribute the assets in such a way that the difference between A_1 and A_2 is as small as possible. Formulate the problem of finding such a distribution as an integer program. Give this formulation for the numerical example with data $n = 10$, and $(a_i) = (14, 76, 46, 54, 22, 5, 68, 68, 94, 39)$. Do the same for the problem with data $n = 10$, and $(a_i) = (8, 12, 117, 148, 2, 85, 15, 92, 152, 130)$. Solve both these problems using an available integer programming package. ([H. M. Weingartner and B. Gavish, May 1993]

7.17: There are 10 customers for a product and 9 potential locations where facilities for manufacturing it can be established. In the

Cust. i	c_{ij} for $j =$									d_i
	1	2	3	4	5	6	7	8	9	
1	15	16	27	28	25	27	27	14	15	28
2	20	13	15	24	13	16	15	15	20	44
3	12	17	25	16	22	15	20	24	26	26
4	25	27	16	23	21	25	26	24	26	31
5	22	10	9	19	18	23	10	22	23	39
6	20	16	24	19	26	24	19	17	20	30
7	17	17	16	25	19	26	14	12	24	43
8	18	20	23	22	28	18	19	17	15	37
9	23	17	16	24	12	25	17	19	22	39
10	14	14	16	20	25	12	23	23	19	30
k_j	46	55	74	68	38	67	52	49	48	
f_j	727	547	674	501	605	482	382	442	606	

above table d_i = expected demand of customer i for the product (in units) over the lifetime of the facilities, k_j = expected production capacity of a facility if established in location j, c_{ij} = cost of transporting the product (per unit) to customer i from a facility established at location j, and f_j = cost of establishing a manufacturing facility at location j.

It is required to determine in which locations manufacturing facilities should be established, and the shipping pattern from the facilities to the customers, so as to meet the demands at minimum total cost which is = the cost of establishing the facilities + the cost of meeting the demand at the customers from the established facilities. Formulate this as an MIP. How does the formulation change if it is required to establish no more than 5 manufacturing facilities? ([K. Darby-Dowman and H. S. Lewis, November 1988]).

7.18: There are n assets with the value of the ith asset being $\$a_i$, $i = 1$ to n. These assets have to be distributed to two beneficiaries in such a way that the first beneficiary gets a fraction f of the total value of all the assets as closely as possible. The assets are indivisible, i.e., each asset has to be given to one beneficiary or the other and cannot be split. Formulate this as an integer program. Obtain an optimum solution for the problem with data $n = 10$, $f = 0.7$, and $(a_i) = (8, 12, 117, 148, 2, 85, 15, 92, 152, 130)$.)[H. M. Weingartner and B. Gavish, May 1993]).

7.19: There are three sites 1, 2, 3 for possible location of manufacturing facilities. There are two products P_1, P_2, and two types of manufacturing facilities (L = large with 4000 units/month production capacity of either product, and S = small with 2000 units/month production capacity of either product) that can be opened at any site.

Site	Monthly set-up cost to make	
	P_1	P_2
1	\$1815	2255
2	1975	2015
3	2215	2575

The fixed monthly set-up cost of operating an L-facility (S-facility) at any site is \$4000 (\$2000). Once the fixed monthly set-up cost of operating a facility at a site is paid, there is an additional monthly set-up cost for equipping that facility to manufacture a product, these are given in the above table.

Data on the unit transportation costs for shipping products is given below.

	Shipping cost/unit from site i to j					
	P_1			P_2		
	$j = 1$	2	3	1	2	3
$i = 1$	1.92	28.8	38.4	6.4	9.6	12.8
2	9.6	6.4	12.8	28.4	25.6	51.2
3	12.8	12.8	6.4	12.8	12.8	6.4

The monthly demand for each product at sites 1, 2, 3, of 500, 1000, 800 units respectively, has to be met. Formulate the problem of determining at which sites facilities have to be set up, of what types, and the shipping patterns, so as to meet the demand at minimum cost (= sum of monthly set-up costs and transportation costs).

7.20: A university library is considering 15 journals as candidates for weeding out of their collection to yield annual subscription savings to meet a proposed budget cut.

The citation counts CO_i in the subject, and CR_i in related area, of a journal J_i refer to the average number of times per year that articles appearing in J_i are referenced in the appropriate scientific literature. The faculty rating R_i of J_i is an average score between 1 and 5 given by the faculty of the university as an indication of the importance of journal J_i, in which the higher the score, the more important the journal is considered to be. The usage data u_i of J_i is the average number of times issues of journal J_i have been removed from the shelf for either borrowing or reading inside the library per quarter, obtained from data collected by the library. The availability rating a_i of journal J_i is an evaluation by the librarian on how easily available this journal is from other libraries; the smaller this rating the easier it is to obtain this journal from other sources. All this data is given below.

Journal	Subscription ($/year)	CO_i	CR_i	R_i	u_i	a_i
J_1	300	200	110	5	70	2
J_2	220	50	120	5	70	1
J_3	400	400	200	2	90	2
J_4	700	60	80	4	90	2
J_5	350	70	100	3	60	2
J_6	260	160	210	5	90	3
J_7	250	351	152	2	105	2
J_8	360	130	111	1	85	2
J_9	250	85	95	6	70	2
J_{10}	210	70	65	5	40	3
J_{11}	260	215	98	3	65	2
J_{12}	320	45	35	4	45	1
J_{13}	200	66	43	5	50	2
J_{14}	520	130	120	4	70	3
J_{15}	200	312	110	1	90	4

The following constraints have to be met.

(a) **Citation count in subject:** Keep the average of the citation count in subject, per canceled journal, to ≤ 800.

(b) **Citation count in related area:** Keep the average of the citation count in related area, per canceled journal, to ≤ 500.

(c) **Faculty ratings:** Keep the average of the faculty rating, per canceled journal, to ≤ 3.

(d) **Journal usage:** Keep the total usage rate per quarter, of all the journals canceled from this list, at 500 or less.

(e) **Availability from other sources:** Keep the average of the availability rating, per canceled journal, to ≤ 2.

Formulate the problem of determining which journals to cancel, to maximize the total subscription cost of journals canceled, subject to the constraints given. ([M. J. Schniederjans and R. Santhanam, 1989]).

7.21: There are 5 projects being considered for approval. The following table presents the data on AR = expected annual return, FI = investment needed in first year, WC = working capital expenses, and SE = expected safety and accident expenses, on each project in some money units.

Project	AR	FI	WC	SE
1	49.3	150	105	1.09
2	39.5	120	83	1.64
3	52.6	90	92	0.95
4	35.7	20	47	0.37
5	38.2	80	54	0.44
Constraint on total	≥ 100	≤ 250	≤ 300	≤ 3.8

Formulate the problem of determining which projects to approve to maximize the expected annual return from the approved projects, subject to the constraints mentioned above.

7.22: There are 7 projects which are being considered for approval.

Project	Other projects that must be approved if this is	Profit or cost of this project
1	2	\$10 m. profit
2		\$8 m. cost
3	1, 5	\$2 m. profit
4	2, 6	\$ 4 m. profit
5		\$5 m. cost
6		\$ 3 m. profit
7	3	\$ 2 m. profit

Some projects can be approved only if other specified projects are also approved, as explained in the above table. Each project results in a profit or cost as indicated in the table. Formulate the problem of determining which projects to approve, so as to maximize the total net profit.

([H. P. Williams, Feb. 1982]).

7.23: There are 5 project proposals. If any of these proposals is accepted, the amount of investment money granted for its implementation must be 1, 2, 3, or 4 units (the money unit in this problem is \$100,000). Rejecting a proposal is equivalent to accepting it with an investment grant of 0 units for its implementation. Data on the managing costs (MC), the expected annual returns (EAR), and the % interest expenses on investment (I) on each project are tabulated below.

Proj.	MC and EAR if amount granted is								I
	1		2		3		4		
	MC	EAR	MC	EAR	MC	EAR	MC	EAR	
1	0.02	0.045	0.04	0.087	0.06	0.117	0.08	0.147	1
2	0.03	0.055	0.054	0.106	0.078	0.146	0.1	0.166	2
3	0.07	0.08	0.13	0.14	0.18	0.18	0.22	0.19	3
4	0.05	0.07	0.1	0.123	0.14	0.163	0.18	0.193	4
5	0.04	0.065	0.075	0.121	0.11	0.171	0.14	0.191	5

The total interest expenses cannot exceed 0.17, and the total managing costs have to be ≤ 0.2 in money units. Formulate the problem of determining which projects to accept, and how much investment money to grant to each of the accepted projects, so as to maximize the total expected annual return from the accepted projects, subject to these constraints, as an integer program with the smallest possible number of constraints. Find an optimum solution to the problem.

7.24: An investment firm is faced with a pool of 20 capital investment projects, and has as its constraints a limited capital budget of 600, 450, and 80 for periods 1, 2, 3, respectively; and target cash flow requirements of 240, 300, 320, 350, 130, 125, and 120 for periods 4 to 10, respectively. Formulate the problem of selecting the projects to invest in to maximize the present value of total return over the 10 year planning horizon with 8% as the interest rate for money per period, subject to these constraints.

Project	Net cash flow in period									
	1	2	3	4	5	6	7	8	9	10
1	−50	−50	15	25	15	15	30	20	10	0
2	−80	−35	−35	40	70	70	70	0	0	0
3	−70	−60	−60	45	45	50	50	50	50	50
4	−35	−45	25	25	25	30	30	0	0	0
5	−50	−30	−30	0	0	0	40	20	20	20
6	−5	−45	−10	30	15	15	15	20	20	0
7	−100	−30	20	35	35	40	20	10	10	10
8	−50	10	−40	20	20	0	0	0	0	0
9	−20	−30	−40	−15	20	25	30	35	40	45
10	−55	0	5	35	15	10	0	0	0	0
11	−40	−10	25	10	10	15	15	10	5	5
12	−30	−50	−20	25	25	25	20	0	0	0
13	0	−20	−25	10	20	30	30	10	0	0
14	−60	−50	−20	−25	40	50	60	0	0	0
15	−10	−25	−20	20	25	35	20	0	0	0
16	−100	−20	15	20	20	30	0	0	0	0
17	−30	5	−10	15	15	15	15	15	15	15
18	−55	−10	10	15	20	20	20	0	0	0
19	−55	5	10	30	30	0	0	0	0	0
20	−75	−50	−30	20	30	40	40	20	20	20

7.25: Metal Ingot Production A steel company has to fill orders for 4 types of ingots. For $i = 1$ to 4, r_i is the number of ingots of type i to be delivered, this and other data is given below.

Ingot type	1	2	3	4
Weight (tons)	7	11	15	23
r_i	53	84	117	243

They smelt the metal in vessels of fixed size of 100 tons, and then cast it into ingots. A vessel of liquid metal which is ready to be poured is called a *heat*, it is cast into as many full ingots as possible that can be made with it, and any leftover metal is poured out and has to be remelted - an expensive operation. That's why leftover metal at the end of pouring a heat is called *wastage* (measured in tons) and the company tries to minimize the total wastage generated.

The company prepares several combinations of ingots of various types that can be poured from a heat. For example, here are two combinations: Combination 1 - 4 ingots of type 4, and 1 ingot of type 1; Combination 2 - 5 ingots of type 3, 1 ingot of type 2, and 2 ingots of type 1. If Combination 1 is poured, the wastage is $100 - 4\times23 - 1\times7 = 1$ ton. If Combination 2 is poured, the wastage is $100 - 5\times15 - 1\times11 - 2\times7 = 0$.

Generate at least 10 different good combinations, i.e., those in which the wastage is reasonably small.

Construct a model to determine how many heats should be poured for each of the combinations generated by you so that the number of ingots of type i produced is $\geq r_i$ for all i, while minimizing 3(wastage) + (weight of ingots of all types left over after the order is filled). Find an optimum solution for this model using one of the available integer programming software packages.

Discuss how the company's Industrial Engineer should organize the weekly pouring schedule if there are many types of ingots to consider (as many as 50 different types), and the number of ingots of each type to be produced that week becomes known at the beginning of the week. ([R. W. Ashford and R. C. Daniel, May 1992]).

7.26:A firm is faced with a pool of 12 investment projects which are examined over a 5-year horizon, and have as constraints a limited capital budget in periods 1 and 2, as well as a target cash inflow for periods 3, 4, 5. The following table presents the data.

It is required to determine the projects to invest in, to maximize the total return while meeting all the constraints. Formulate this problem.

Project	Net cash flow in period				
	1	2	3	4	5
1	−45	−15	40	60	0
2	−90	25	35	70	0
3	−50	0	30	50	0
4	−60	−10	10	60	40
5	−100	−10	25	60	80
6	−40	−35	30	50	30
7	−60	5	10	30	50
8	−80	−15	35	40	60
9	−75	0	−10	50	75
10	−30	−20	−5	40	40
11	−35	−10	−5	30	40
12	−54	−20	−15	50	70
Available budget	500	80			
Target cash inflow			90	290	245

7.27: A company is making plans to manufacture two products P_1, P_2. There are 6 operations O_1 to O_6, some are required by both products, others are required by only one product, as explained below. There are three types of machines, M_1, M_2, M_3 each of which can perform some of the operations on one or both of the products as explained below.

Product data

Product	Operations to be performed	Annual Prod. target (units)
P_1	O_1, O_2, O_6, O_5	48,000
P_2	O_2, O_3, O_5, O_4	38,000

Processing time (PT) data (minutes/unit)

Operation	P_1			P_2		
	PT if carried out on					
	M_1	M_2	M_3	M_1	M_2	M_3
O_1	5.2	6.2	7.3			
O_2	3.0	3.0		2.5	2.9	
O_3				4.0		5.2
O_4				2.0	2.0	2.4
O_5	2.0		2.2	5.9	7.1	8.1
O_6	7.0	8.2	9.0			

Blank indicates either that operation not needed for product, or that mc. can't perform operation on product

Each machine will be available to work 1900 hours/year. Machine types M_1, M_2, M_3 cost \$96,000, \$82,000, \$70,000 respectively per copy. The company can buy a nonnegative number of copies of each machine type. The problem is to determine how many copies of each machine type to buy, and how much of each product-operation combination to allot to each machine purchased, in order to carry out the yearly workload with minimum investment. Formulate as an MIP. (Y. Bozer).

7.28: Segregated Storage Problem There are m different products to be stored, with $a_i > 0$ being the quantity of product i to be stored in some units. There are n different storage compartments, with $b_j > 0$ being the capacity in units of storage compartment j.

However, in each compartment, at most one product may be stored. A typical problem of this type is the 'silo problem' in which different varieties of grain are to be stored separately in the various compartments of the silo. Other examples are: different types of crude oil in storage tanks, customer orders on trucks with no mixing of different orders on any truck, etc.

We assume that there exists external storage space, available at premium cost, which is capable of storing any and all products. Call this external storage the $(n+1)$st compartment. For $i = 1$ to m, $j = 1$ to $n+1$, let c_{ij} denote the unit cost of storing product i in compartment j.

It is required to store the available quantities of the products in the compartments at minimum cost, subject to the storage capacities of compartments 1 to n, and the constraint that each of the compartments 1 to n can hold at most one product. Formulate this problem. Give this formulation for the numerical example with the following data in which the fourth compartment represents external storage ([A. W. Neebe, Sept. 1987]).

	c_{ij} for $j =$				
Compartment $j =$	1	2	3	4	a_i
Product $i = 1$	20	14	19	24	1
2	15	13	20	22	8
3	18	18	15	22	7
Capacity b_j	3	7	4	16	

7.29: The Symmetric Assignment Problem There are 6 students in a projects course. It is required to form them into groups of at most 2 students each (so a single student can constitute a group by himself/herself). Here is the cost data. It is required to find a minimum cost grouping. Formulate this as an integer program.

Cost of forming students i, j into a group, for $j \geq i$

$j =$	1	2	3	4	5	6
$i = 1$	16	10	8	58	198	70
2		10	6	72	50	32
3			15	26	198	24
4				15	14	18
5					13	6
6						10

7.30: The Asymmetric Assignment Problem Let $C = (c_{ij})$ be an $n \times n$ cost matrix for an assignment problem, with $c_{ii} = \infty$ for all i (i.e., all cells (i, i) are forbidden cells). Here we want a minimum cost assignment satisfying the additional constraints "if $x_{ij} = 1$, then $x_{ji} = 0$, for all $i \neq j$." These conditions are called **asymmetry**

constraints since they force the feasible assignment to be asymmetric. Because of this the assignment problem with these constraints is known as the **asymmetric assignment problem**. Give a formulation of this problem.

Additional exercises for this chapter are available in Chapter 13 at the end.

7.12 References

Here **JORS** = *Journal of the Operatioal Research Society*, **COR** = *Computers and Operations Research*

R. W. ASHFORD, and R. C. DANIEL, May 1992, "Some Lessons in Solving Practical Integer Programs", *JORS*, 43, no. 5, 425-433.

Y. L. CHEN, and Y. H. CHIN, 1989, "Scheduling Unit-time Jobs On Processors With Different Capabilities", *COR*, 16, no. 5, 409-417.

M. CHESHIRE, K. I. M. McKINNON, and H. P. WILLIAMS, Aug. 1984, "The Efficient Allocation of Private Contractors to Public Works", *JORS*, 35, no. 8, 705-709.

M. J. CHLOND, and C. M. TOASE, 2003, "IP Modeling and the Logical Puzzles of Raymond Smullyan", *INFORMS Transactions on Education, 3, no. 3, 1-12, at the website: http://ite.pubs.informs.org/Vol3No3/ChlondToase/*

K. DARBY-DOWMAN, and H. S. LEWIS, Nov. 1988, "Lagrangian Relaxation and the Single-source Capacitated Facility-location Problem", JORS*, 39, no. 11, 1035-1040.*

J. DARZENTAS, and P. BAIRAKTARIS, 1986, "On the Set Partitioning Type Formulation for the Discrete Location Problem", COR*, 13, no. 6, 671-679.*

R. F. DECKRO, and S. RANGACHARI, 1990, "A Goal Approach to Assembly Line Balancing", COR*, 17, no. 5, 509-521.*

S. DUFFUAA, Nov. 1991, "A Mathematical Optimization Model for Chemical Production at Saudi Arabian Fertilizer Company", Applied Mathematical Modeling*, 15 (652-656).*

S. DUFFUAA, and N. H. AL-SAGGAF, 1992, "An Integrated Optimization Model for Three Reactors Settings", Tech. report, SE, KFUPM, Dhahran 31261, Saudi Arabia.

A. J. KEOWN; B. W. TAYLOR,III; and J. M. PINKERTON; 1981, "Multiple Objective Capital Budgeting Within the University", COR*, 8, 59-70.*

J. MAES, and L. V. WASSENHOVE, Nov. 1988, "Multi-item Single-level Capacitated Dynamic Lot-sizing Heuristics: A General Review", JORS*, 39, no. 11, 991-1004.*

J. B. MAZZOLA, and A. W. NEEBE, 1993, "An Algorithm for the Bottleneck

Generalized Assignment Problem", COR*, 20, no. 4, 355-362.*
B. MOORES, 1986, "Dynamic Programming in Transformer Design", JORS*, 37, no. 10, 967-969.*
K. G. MURTY, P. DJANG, W. BUTLER, and R. R. LAFERRIERE, 1995, "The Army Training Mix Model", JORS*, 46, 294-303.*
A. W. NEEBE, Sept. 1987, "An Improved Multiplier Adjustment Procedure for the Segregated Storage Problem", JORS*, 38, no. 9, 815-825.*
A. W. NEEBE, and M. R. RAO, Nov. 1983, "An Algorithm for the Fixed-charge Assigning Users to Sources Problem", JORS*, 34, no. 11, 1107-1113.*
H. PIRKUL, and D. A. SCHILLING, Feb. 1991, "The Maximal Covering Location Problem With Capacities on Total Workload", Management Science*, 37, no. 2, 233-248.*
P. R. RAO, Oct. 1990, "A Dynamic Programming Approach to Determine Optimal Manpower Recruitment Policies", JORS*, 41, no. 10, 983-988.*
G. R. REEVES, and E. P. HICKMAN, Sept.-Oct. 1992, "Assigning MBA Students to Field Study Project Teams: A Multicriteria Approach", Interfaces*, 22, no. 5, 52-58.*
M. J. SCHNIEDERJANS, and G. C. KIM, 1987, "A Goal Programming Model to Optimize Departmental Preference in Course Assignments", COR*, 14, no. 2, 87-96.*
M. J. SCHNIEDERJANS, and R. SANTHANAM, 1989, "A 0-1 Goal Programming Approach for the Journal Selection and Cancellation Problem", COR*, 16, no. 6, 557-565.*
R. M. SMULLYAN, 1978, What Is the Name of this Book?*, Prentice Hall, Englewood Cliffs, NJ.*
R. M. SMULLYAN, 1982, The Lady or the Tiger*, Alfred A. Knopf, NY.*
R. M. SMULLYAN, 1997, The Riddle of Scheherazade and Other Amazing Puzzles, Ancient and Modern*, Alfred A. Knopf, NY.*
R. SRIDHARAN, July 1991, "A Lagrangian Heuristic for the Capacitated Plant Location Problem", JORS*, 42, no. 7, 579-585.*
P. TRYFOS, Oct. 1986, "An Integer Programming Approach to the Apparel Sizing Problem", JORS*, 37, no. 10, 1001-1006.*
G. WEBER, March-April 1990, "Puzzle Contests in MS/OR Education", Interfaces*, 20, no. 2, 72-76.*
H. M. WEINGARTNER, and B. GAVISH, May 1993, "How to Settle an Estate", Management Science*, 39, no. 5, 588-601.*
H. P. WILLIAMS, Feb. 1982, "Models With Network Duals", JORS*, 33, no. 2, 161-169.*
J. M. WILSON, 1988, "Programming in Transformer design", JORS*, 39, no. 1, 109-110.*

Index

For each index entry we provide the section number where it is defined or discussed first, and number of some sections where it appears prominently.