فصل ۱۳- بهینه‌سازی عدد صحیح (گسسته)

ورود به مطلب– مطالب این فصل عینا از کتاب Engineering Optization اثر Rao فصل دهم اقتباس شده است.

10

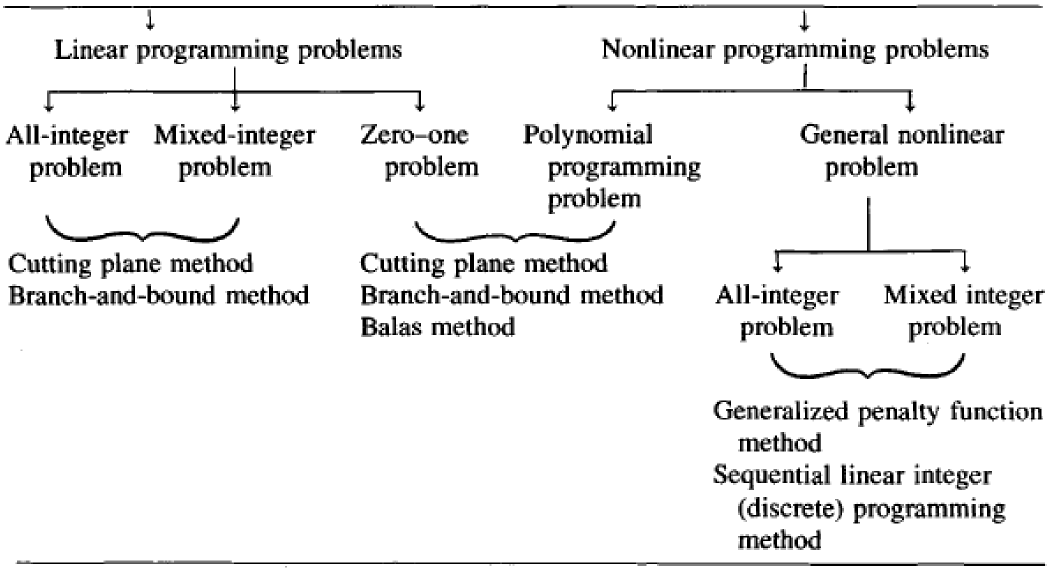
INTEGER PROGRAMMING

10.1 INTRODUCTION

In all the optimization techniques considered so far, the design variables are assumed to be continuous, which can take any real value. In many situations it is entirely appropriate and possible to have fractional solutions. For example, it is possible to use a plate of thickness 2.60 mm in the construction of a boiler shell, 3.34 hours of labor time in a project, and 1.78 lb of nitrate to produce a fertilizer. Also, in many engineering systems, certain design variables can only have discrete values. For example, pipes carrying water in a heat exchanger may be available only in diameter increments of $\frac{1}{8}$ in. However, there are practical problems in which the fractional values of the design variables are neither practical nor physically meaningful. For example, it is not possible to use 1.6 boilers in a thermal power station, 1.9 workers in a project, and 2.76 lathes in a machine shop. If an integer solution is desired, it is possible to use any of the techniques described in previous chapters and round off the optimum values of the design variables to the nearest integer values. However, in many cases, it is very difficult to round off the solution without violating any of the constraints. Frequently, the rounding of certain variables requires substantial changes in the values of some other variables to satisfy all the constraints. Further, the round-off solution may give a value of the objective function that is very far from the original optimum value. All these difficulties can be avoided if the optimization problem is posed and solved as an integer programming problem.

When all the variables are constrained to take only integer values in an optimization problem, it is called an (all)-integer programming problem. When the variables are restricted to take only discrete values, the problem is called

TABLE 10.1 Integer Programming Methods



Linear programming problems

All-integer problem

Mixed-integer problem

Zero–one problem

Cutting plane method
Branch-and-bound method

Cutting plane method
Branch-and-bound method
Balas method

Nonlinear programming problems

Polynomial programming problem

General nonlinear problem

All-integer problem

Mixed integer problem

Generalized penalty function method
Sequential linear integer (discrete) programming method

a *discrete programming problem*. When some variables only are restricted to take integer (discrete) values, the optimization problem is called a *mixed-integer (discrete) programming problem*. When all the design variables of an optimization problem are allowed to take on values of either zero or 1, the problem is called a *zero-one programming problem*. Among the several techniques available for solving the all-integer and mixed-integer linear programming problems, the cutting plane algorithm of Gomory [10.7] and the branch-and-bound algorithm of Land and Doig [10.8] have been quite popular. Although the zero–one linear programming problems can be solved by the general cutting plane or the branch-and-bound algorithms, Balas [10.9] developed an efficient enumerative algorithm for solving those problems. Very little work has been done in the field of integer nonlinear programming. The generalized penalty function method and the sequential linear integer (discrete) programming method can be used to solve all integer and mixed-integer nonlinear programming problems. The various solution techniques of solving integer programming problems are summarized in Table 10.1. All these techniques are discussed briefly in this chapter.

INTEGER LINEAR PROGRAMMING

10.2 GRAPHICAL REPRESENTATION

Consider the following integer programming problem:

$$\text{Maximize } f(\mathbf{X}) = 3x_1 + 4x_2$$

subject to

$$
\begin{aligned}
3x_1 - x_2 &\le 12 \\
3x_1 + 11x_2 &\le 66 \\
x_1 &\ge 0 \qquad\qquad (10.1)\\
x_2 &\ge 0 \\
x_1 \text{ and } x_2 &\text{ are integers}
\end{aligned}
$$

The graphical solution of this problem, by ignoring the integer requirements, is shown in Fig. 10.1. It can be seen that the solution is $x_1 = 5\frac{1}{2}$, $x_2 = 4\frac{1}{2}$ with a value of $f = 34\frac{1}{2}$. Since this is a noninteger solution, we truncate the fractional parts and obtain the new solution as $x_1 = 5$, $x_2 = 4$, and $f = 31$. By comparing this solution with all other integer feasible solutions (shown by dots in Fig. 10.1), we find that this solution is optimum for the integer LP problem stated in Eqs. (10.1).

It is to be noted that truncation of the fractional part of a LP problem will not always give the solution of the corresponding integer LP problem. This can be illustrated by changing the constraint $3x_1 + 11x_2 \le 66$ to $7x_1 + 11x_2 \le 88$ in Eqs. (10.1). With this altered constraint, the feasible region and the

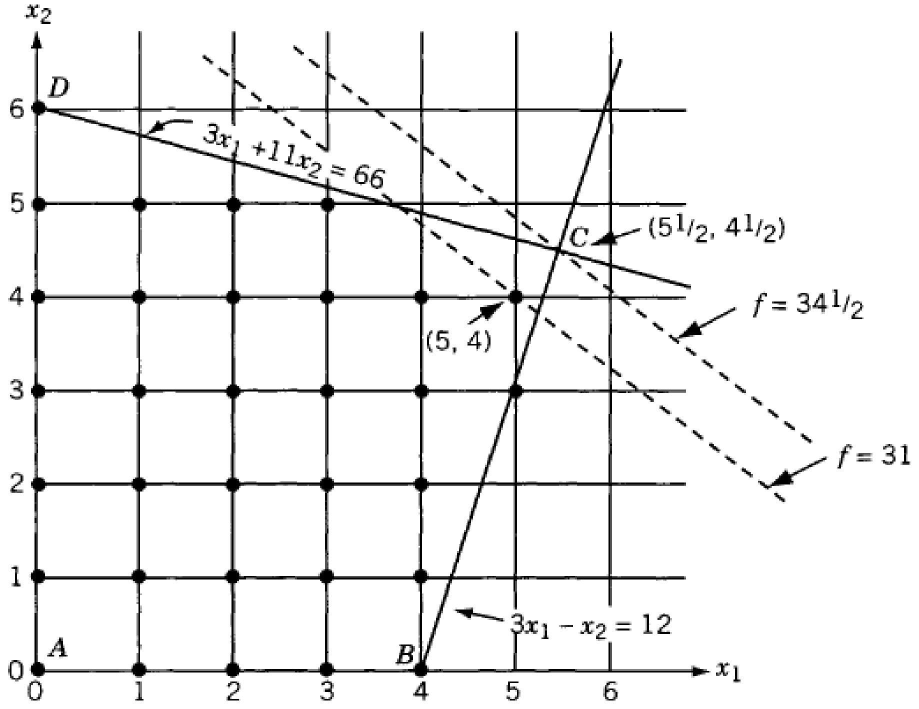


Figure 10.1 Graphical solution of the problem stated in Eqs. (10.1).

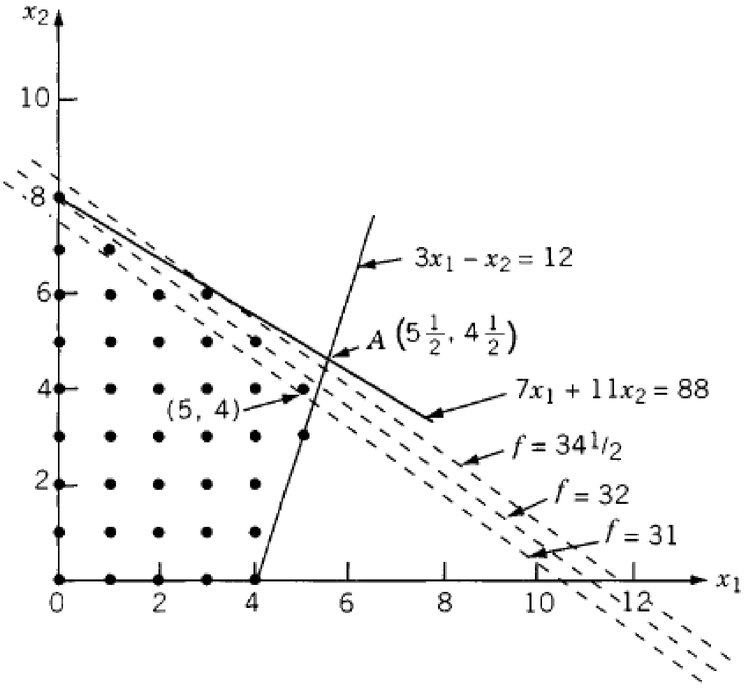


Figure 10.2 Graphical solution with modified constraint.

solution of the LP problem, without considering the integer requirement, are shown in Fig. 10.2. The optimum solution of this problem is identical with that of the preceding problem: namely, $x_1 = 5\frac{1}{2}$, $x_2 = 4\frac{1}{2}$, and $f = 34\frac{1}{2}$. The truncation of the fractional part of this solution gives $x_1 = 5$, $x_2 = 4$, and $f = 31$. Although this truncated solution happened to be optimum to the corresponding integer problem in the earlier case, it is not so in the present case. In this case the optimum solution of the integer programming problem is given by $x_1^* = 0$, $x_2^* = 8$, and $f^* = 32$.

10.3 GOMORY'S CUTTING PLANE METHOD

10.3.1 Concept of a Cutting Plane

Gomory's method is based on the idea of generating a cutting plane. To illustrate the concept of a cutting plane, we again consider the problem stated in Eqs. (10.1). The feasible region of the problem is denoted by *ABCD* in Fig. 10.1. The optimal solution of the problem, without considering the integer requirement, is given by point *C*. This point corresponds to $x_1 = 5\frac{1}{2}$, $x_2 = 4\frac{1}{2}$, and $f = 34\frac{1}{2}$, which is not optimal to the integer programming problem since the values of x_1 and x_2 are not integers. The feasible integer solutions of the problem are denoted by dots in Fig. 10.1. These points are called the *integer lattice points*.

In Fig. 10.3, the original feasible region is reduced to a new feasible region *ABEFGD* by including the additional (arbitrarily selected) constraints. The idea behind adding these additional constraints is to reduce the original feasible convex region *ABCD* to a new feasible convex region (such as *ABEFGD*) such that an extreme point of the new feasible region becomes an integer optimal solution to the integer programming problem. There are two main considerations to be taken while selecting the additional constraints: (1) the new feasible region should also be a convex set, and (2) the part of the original feasible region that is sliced off because of the additional constraints should not include any feasible integer solutions of the original problem.

In Fig. 10.3, the inclusion of the two arbitrarily selected additional constraints *PQ* and *P′Q′* gives the extreme point $F(x_1 = 5, x_2 = 4, f = 31)$ as the optimal solution of the integer programming problem stated in Eqs. (10.1). Gomory's method is one in which the additional constraints are developed in a systematic manner.

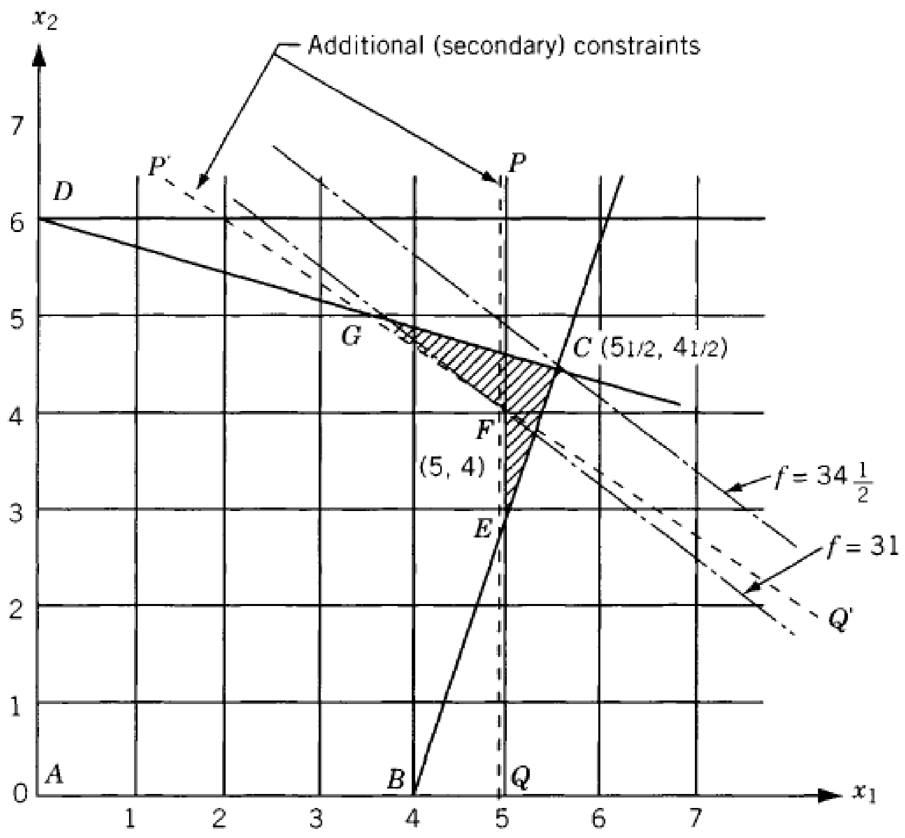


Figure 10.3 Effect of additional constraints.

10.3.2 Gomory's Method for All-Integer Programming Problems

In this method the given problem [Eqs. (10.1)] is first solved as an ordinary LP problem by neglecting the integer requirement. If the optimum values of the variables of the problem happen to be integers, there is nothing more to be done since the integer solution is already obtained. On the other hand, if one or more of the basic variables have fractional values, some additional constraints, known as *Gomory constraints*, which will force the solution toward an all-integer point will have to be introduced. To see how the Gomory constraints are generated, let the tableau corresponding to the optimum (noninteger) solution of the ordinary LP problem be as shown in Table 10.2. Here it is assumed that there are a total of $m + n$ variables (n original variables plus m slack variables). At the optimal solution, the basic variables are represented as x_i ($i = 1,2,\ldots,m$) and the nonbasic variables as y_j ($j = 1,2,\ldots,n$) for convenience.

Gomory's Constraint. From Table 10.2, choose the basic variable with the largest fractional value. Let this basic variable be x_i. When there is a tie in the fractional values of the basic variables, any of them can be taken as x_i. This variable can be expressed, from the ith equation of Table 10.2, as

$$x_i = \bar{b}_i - \sum_{j=1}^{n} \bar{a}_{ij} y_j \tag{10.2}$$

where $\bar{b}_i$ is a noninteger. Let us write

$$\bar{b}_i = \hat{b}_i + \beta_i \tag{10.3}$$

$$\bar{a}_{ij} = \hat{a}_{ij} + \alpha_{ij} \tag{10.4}$$

where $\hat{b}_i$ and $\hat{a}_{ij}$ denote the integers obtained by truncating the fractional parts from $\bar{b}_i$ and $\bar{a}_{ij}$, respectively. Thus β_i will be a strictly positive fraction ($0 < \beta_i < 1$) and α_{ij} will be a nonnegative fraction ($0 \le \alpha_{ij} < 1$). With the help

TABLE 10.2 Optimum Noninteger Solution of Ordinary LP Problem

Basic	Coefficient Corresponding to:								Objective	
Variables	x_1	$x_2 \cdots$	$x_i \cdots$	x_m	y_1	$y_2 \cdots$	$y_j \cdots$	y_n	Function	Constants
x_1	1	0	0	0	$\bar{a}_{11}$	$\bar{a}_{12}$	$\bar{a}_{1j}$	$\bar{a}_{1n}$	0	$\bar{b}_1$
x_2	0	1	0	0	$\bar{a}_{21}$	$\bar{a}_{22}$	$\bar{a}_{2j}$	$\bar{a}_{2n}$	0	$\bar{b}_2$
$\vdots$										
x_i	0	0	1	0	$\bar{a}_{i1}$	$\bar{a}_{i2}$	$\bar{a}_{ij}$	$\bar{a}_{in}$	0	$\bar{b}_i$
$\vdots$										
x_m	0	0	0	1	$\bar{a}_{m1}$	$\bar{a}_{m2}$	$\bar{a}_{mj}$	$\bar{a}_{mn}$	0	$\bar{b}_m$
f	0	$0 \cdots 0 \cdots 0$			$\bar{c}_1$	$\bar{c}_2$	$\bar{c}_j$	$\bar{c}_n$	1	$\bar{f}$

of Eqs. (10.3) and (10.4), Eq. (10.2) can be rewritten as

$$\beta_i - \sum_{j=1}^{n} \alpha_{ij} y_j = x_i - \hat{b}_i + \sum_{j=1}^{n} \hat{a}_{ij} y_j \tag{10.5}$$

Since all the variables x_i and y_j must be integers at an optimal integer solution, the right-hand side of Eq. (10.5) must be an integer. Thus we obtain

$$\beta_i - \sum_{j=1}^{n} \alpha_{ij} y_j = \text{integer} \tag{10.6}$$

Notice that α_{ij} are nonnegative fractions and y_j are nonnegative integers. Hence the quantity $\Sigma_{j=1}^{n} \alpha_{ij} y_j$ will always be a nonnegative number. Since β_i is a strictly positive fraction, we have

$$\left(\beta_i - \sum_{j=1}^{n} \alpha_{ij} y_j\right) \leq \beta_i < 1 \tag{10.7}$$

As the quantity $(\beta_i - \Sigma_{j=1}^{n} \alpha_{ij} y_j)$ has to be an integer [from Eq. (10.6)], it can be either a zero or a negative integer. Hence we obtain the desired constraint as

$$+\beta_i - \sum_{j=1}^{n} \alpha_{ij} y_j \leq 0 \tag{10.8}$$

By adding a nonnegative slack variable s_i, the Gomory constraint equation becomes

$$s_i - \sum_{j=1}^{n} \alpha_{ij} y_j = -\beta_i \tag{10.9}$$

where s_i must also be an integer by definition.

Computational Procedure. Once the Gomory constraint is derived, the coefficients of this constraint are inserted in a new row of the final tableau of the ordinary LP problem (i.e., Table 10.2). Since all $y_j = 0$ in Table 10.2, the Gomory constraint equation (10.9), becomes

$$s_i = -\beta_i = \text{negative}$$

which is infeasible. This means that the original optimal solution is not satisfying this new constraint. To obtain a new optimal solution that satisfies the new constraint, Eq. (10.9), the dual simplex method discussed in Chapter 4

TABLE 10.3 Optimal Solution with Gomory Constraint

Basic	Coefficient Corresponding to:										
Variables	x_1	$x_2 \cdots$	$x_i \cdots$	x_m	y_1	$y_2 \cdots$	$y_j \cdots$	y_n	f	s_i	Constants
x_1	1	0	0	0	$\bar{a}_{11}$	$\bar{a}_{12}$	$\bar{a}_{1j}$	$\bar{a}_{1n}$	0	0	$\bar{b}_1$
x_2	0	1	0	0	$\bar{a}_{21}$	$\bar{a}_{22}$	$\bar{a}_{2j}$	$\bar{a}_{2n}$	0	0	$\bar{b}_2$
$\vdots$											
x_i	0	0	1	0	$\bar{a}_{i1}$	$\bar{a}_{i2}$	$\bar{a}_{ij}$	$\bar{a}_{in}$	0	0	$\bar{b}_i$
$\vdots$											
x_m	0	0	0	1	$\bar{a}_{m1}$	$\bar{a}_{m2}$	$\bar{a}_{mj}$	$\bar{a}_{mn}$	0	0	$\bar{b}_m$
f	0	0	0	0	$\bar{c}_1$	$\bar{c}_2$	$\bar{c}_j$	$\bar{c}_n$	1	0	$\bar{f}$
s_i	0	0	0	0	$-\alpha_{i1}$	$-\alpha_{i2}$	$-\alpha_{ij}$	$-\alpha_{in}$	0	1	$-\beta_i$

can be used. The new tableau, after adding the Gomory constraint, is as shown in Table 10.3.

After finding the new optimum solution by applying the dual simplex method, test whether the new solution is all-integer or not. If the new optimum solution is all-integer, the process ends. On the other hand, if any of the basic variables in the new solution take on fractional values, a new Gomory constraint is derived from the new simplex tableau and the dual simplex method is applied again. This procedure is continued until either an optimal integer solution is obtained or the dual simplex method indicates that the problem has no feasible integer solution.

Remarks:

1. If there is no feasible integer solution to the given (primal) problem, this can be detected by noting an unbounded condition for the dual problem.
2. The application of the dual simplex method to remove the infeasibility of Eq. (10.9) is equivalent to cutting off the original feasible solution towards the optimal integer solution.
3. This method has a serious drawback. This is associated with the round-off errors that arise during numerical computations. Due to these round-off errors, we may ultimately get a wrong optimal integer solution. This can be rectified by storing the numbers as fractions instead of as decimal quantities. However, the magnitudes of the numerators and denominators of the fractional numbers, after some calculations, may exceed the capacity of the computer. This difficulty can be avoided by using the all-integer integer programming algorithm developed by Gomory [10.10].
4. For obtaining the optimal solution of an ordinary LP problem, we start from a basic feasible solution (at the start of phase II) and find a sequence of improved basic feasible solutions until the optimum basic feasible solution is found. During this process, if the computations have to be

terminated at any stage (for some reason), the current basic feasible solution can be taken as an approximation to the optimum solution. However, this cannot be done if we apply Gomory's method for solving an integer programming problem. This is due to the fact that the problem remains infeasible in the sense that no integer solution can be obtained until the whole problem is solved. Thus we will not be having any good integer solution that can be taken as an approximate optimum solution in case the computations have to be terminated in the middle of the process.

5. From the description given above, the number of Gomory constraints to be generated might appear to be very large, especially if the solution converges slowly. If the number of constraints really becomes very large, the size of the problem also grows without bound since one (slack) variable and one constraint are added with the addition of each Gomory constraint. However, it can be observed that the total number of constraints in the modified tableau will not exceed the number of variables in the original problem, namely, $n + m$. The original problem has m equality constraints in $n + m$ variables and we observe that there are n nonbasic variables. When a Gomory constraint is added, the number of constraints and the number of variables will each be increased by one, but the number of nonbasic variables will remain n. Hence at most n slack variables of Gomory constraints can be nonbasic at any time, and any additional Gomory constraint must be redundant. In other words, at most n Gomory constraints can be binding at a time. If at all a $(n + 1)$th constraint is there (with its slack variable as a basic and positive variable), it must be implied by the remaining constraints. Hence we drop any Gomory constraint once its slack variable becomes basic in a feasible solution.

Example 10.1

$$\text{Minimize } f = -3x_1 - 4x_2$$

subject to

$$3x_1 - x_2 + x_3 = 12$$

$$3x_1 + 11x_2 + x_4 = 66$$

$$x_i \geq 0, \qquad i = 1 \text{ to } 4$$

$$\text{all } x_i \text{ are integers.}$$

This problem can be seen to be same as the one stated in Eqs. (10.1) with the addition of slack variables x_3 and x_4.

SOLUTION

Step 1: Solve the LP problem by neglecting the integer requirement of the variables x_i, $i = 1$ to 4, using the regular simplex method as shown below.

Basic Variables	Coefficients of Variables						
	x_1	x_2	x_3	x_4	$-f$	$\bar{b}_i$	$\bar{b}_i/\bar{a}_{is}$ for $\bar{a}_{is} > 0$
x_3	3	-1	1	0	0	12	
x_4	3	$\boxed{11}$ Pivot element	0	1	0	66	6 ←
$-f$	-3	-4	0	0	1	0	
		↑ Most negative $\bar{c}_j$					

Result of pivoting:

x_3	$\boxed{\frac{36}{11}}$ Pivot element	0	1	$\frac{1}{11}$	0	18	$\frac{11}{2}$ ← Smaller one
x_2	$\frac{3}{11}$	1	0	$\frac{1}{11}$	0	6	22
$-f$	$-\frac{21}{11}$	0	0	$\frac{4}{11}$	1	24	
	↑ Most negative $\bar{c}_j$						

Result of pivoting:

x_1	1	0	$\frac{11}{36}$	$\frac{1}{36}$	0	$-\frac{11}{2}$
x_2	0	1	$-\frac{1}{12}$	$-\frac{1}{12}$	0	$\frac{9}{2}$
$-f$	0	0	$\frac{7}{12}$	$\frac{5}{12}$	1	$\frac{69}{2}$

Since all the cost coefficients are nonnegative, the last tableau gives the optimum solution as

$$x_1 = \tfrac{11}{2}, \quad x_2 = \tfrac{9}{2}, \quad x_3 = 0, \quad x_4 = 0, \quad f_{\min} = -\tfrac{69}{2}$$

which can be seen to be identical to the graphical solution obtained in Section 10.2.

Step 2: Generate a Gomory constraint. Since the solution above is noninteger, a Gomory constraint has to be added to the last tableau. Since there is a tie between x_1 and x_2, let us select x_1 as the basic variable having the largest fractional value. From the row corresponding to x_1 in the last tableau, we can write

$$x_1 = \tfrac{11}{2} - \tfrac{11}{36}\, y_1 - \tfrac{1}{36}\, y_2 \tag{E$_1$}$$

where y_1 and y_2 are used in place of x_3 and x_4 to denote the nonbasic variables. By comparing Eq. (E$_1$) with Eq. (10.2), we find that

$$i = 1,\quad \bar{b}_1 = \tfrac{11}{2},\quad \hat{b}_1 = 5,\quad \beta_1 = \tfrac{1}{2},\quad \bar{a}_{11} = \tfrac{11}{36},$$
$$\hat{a}_{11} = 0,\quad \alpha_{11} = \tfrac{11}{36},\quad \bar{a}_{12} = \tfrac{1}{36},\quad \hat{a}_{12} = 0,\quad \text{and}\quad \alpha_{12} = \tfrac{1}{36}$$

From Eq. (10.9), the Gomory constraint can be expressed as

$$s_1 - \alpha_{11} y_1 - \alpha_{12} y_2 = -\beta_1 \tag{E$_2$}$$

where s_1 is a new nonnegative (integer) slack variable. Equation (E$_2$) can be written as

$$s_1 - \tfrac{11}{36}\, y_1 - \tfrac{1}{36}\, y_2 = -\tfrac{1}{2} \tag{E$_3$}$$

By introducing this constraint, Eq. (E$_3$), into the previous optimum tableau, we obtain the new tableau shown below.

Basic Variables	Coefficients of variables							$\bar{b}_i/\bar{a}_{is}$ for $\bar{a}_{is} > 0$
	x_1	x_2	y_1	y_2	$-f$	s_1	$\bar{b}_i$	
x_1	1	0	$\frac{11}{36}$	$\frac{1}{36}$	0	0	$\frac{11}{2}$	
x_2	0	1	$-\frac{1}{12}$	$\frac{1}{12}$	0	0	$\frac{9}{2}$	
$-f$	0	0	$\frac{7}{12}$	$\frac{5}{12}$	1	0	$\frac{69}{2}$	
s_1	0	0	$-\frac{11}{36}$	$-\frac{1}{36}$	0	1	$-\frac{1}{2}$	

Step 3: Apply the dual simplex method to find a new optimum solution: For this, we select the pivotal row r such that $\bar{b}_r = \min(\bar{b}_i < 0) = -\frac{1}{2}$ corresponding to s_1 in this case. The first column s is selected such that

$$\frac{\bar{c}_s}{-\bar{a}_{rs}} = \min_{\bar{a}_{rj} < 0}\left(\frac{\bar{c}_j}{-\bar{a}_{rj}}\right)$$

Here

$$\frac{\bar{c}_j}{-\bar{a}_{rj}} = \frac{7}{12} \times \frac{36}{11} = \frac{21}{11} \quad \text{for column } y_1$$
$$= \frac{5}{12} \times \frac{36}{1} = 15 \quad \text{for column } y_2.$$

Since $\frac{21}{11}$ is minimum out of $\frac{21}{11}$ and 15, the pivot element will be $(-\frac{11}{36})$. The result of pivot operation is given in the following tableau.

Basic Variables	Coefficients of Variables							$\bar{b}_i/\bar{a}_{is}$ for $\bar{a}_{is} > 0$
	x_1	x_2	y_1	y_2	$-f$	s_1	$\bar{b}_i$	
x_1	1	0	0	0	0	1	5	
x_2	0	1	0	$\frac{1}{11}$	0	$-\frac{3}{11}$	$\frac{51}{11}$	
$-f$	0	0	0	$\frac{4}{11}$	1	$\frac{21}{11}$	$\frac{369}{11}$	
y_1	0	0	1	$\frac{1}{11}$	0	$-\frac{36}{11}$	$\frac{18}{11}$	

The solution given by the present tableau is $x_1 = 5$, $x_2 = 4\frac{7}{11}$, $y_1 = 1\frac{7}{11}$, and $f = -33\frac{6}{11}$, in which some variables are still nonintegers.

Step 4: Generate a new Gomory constraint. To generate the new Gomory constraint, we arbitrarily select x_2 as the variable having the largest fractional value (since there is a tie between x_2 and y_1). The row corresponding to x_2 gives

$$x_2 = \frac{51}{11} - \frac{1}{11}y_2 + \frac{3}{11}s_1$$

From this equation, the Gomory constraint [Eq. (10.9)] can be written as

$$s_2 - \frac{1}{11}y_2 + \frac{3}{11}s_1 = -\frac{7}{11}$$

When this constraint is added to the previous tableau, we obtain the following tableau:

Basic Variables	Coefficients of Variables							
	x_1	x_2	y_1	y_2	$-f$	s_1	s_2	$\bar{b}_i$
x_1	1	0	0	0	0	1	0	5
x_2	0	1	0	$\frac{1}{11}$	0	$-\frac{3}{11}$	0	$\frac{51}{11}$
y_1	0	0	1	$\frac{1}{11}$	0	$-\frac{36}{11}$	0	$\frac{18}{11}$
$-f$	0	0	0	$\frac{4}{11}$	1	$\frac{21}{11}$	0	$\frac{369}{11}$
s_2	0	0	0	$-\frac{1}{11}$	0	$\frac{3}{11}$	1	$-\frac{7}{11}$

Step 5: Apply the dual simplex method to find a new optimum solution: To carry the pivot operation, the pivot row is selected to correspond to the most negative value of $\bar{b}_i$. This is the s_2 row in this case.

Since only $\bar{a}_{rj}$ corresponding to column y_2 is negative, the pivot element will be $(-1/11)$ in the s_2 row. The pivot operation on this element leads to the following tableau:

Basic	Coefficients of Variables							
Variables	x_1	x_2	y_1	y_2	$-f$	s_1	s_2	$\bar{b}_i$
x_1	1	0	0	0	0	1	0	5
x_2	0	1	0	0	0	0	1	4
y_1	0	0	1	0	0	-3	1	1
$-f$	0	0	0	0	1	3	4	31
y_2	0	0	0	1	0	-3	-11	7

The solution given by this tableau is $x_1 = 5$, $x_2 = 4$, $y_1 = 1$, $y_2 = 7$, and $f = -31$, which can be seen to satisfy the integer requirement. Hence this is the desired solution.

10.3.3 Gomory's Method for Mixed-Integer Programming Problems

The method discussed in Section 10.3.2 is applicable to solve all integer programming problems where both the decision and slack variables are restricted to integer values in the optimal solution. In the mixed-integer programming problems, only a subset of the decision and slack variables are restricted to integer values. The procedure for solving mixed-integer programming problems is similar to that of all-integer programming problems in many respects.

Solution Procedure. As in the case of an all-integer programming problem, the first step involved in the solution of a mixed-integer programming problem is to obtain an optimal solution of the ordinary LP problem without considering the integer restrictions. If the values of the basic variables, which were restricted to integer values, happen to be integers in this optimal solution, there is nothing more to be done. Otherwise, a Gomory constraint is formulated by taking the integer-restricted basic variable, which has the largest fractional value in the optimal solution of the ordinary LP problem.

Let x_i be the basic variable which has the largest fractional value in the optimal solution (as shown in Table 10.2), although it is restricted to take on only integer values. If the nonbasic variables are denoted as y_j, $j = 1,2,\ldots,n$, the basic variable x_i can be expressed as (from Table 10.2)

$$x_i = \bar{b}_i - \sum_{j=1}^{n} \bar{a}_{ij} y_j \qquad (10.2)$$

We can write

$$\bar{b}_i = \hat{b}_i + \beta_i \qquad (10.3)$$

where $\hat{b}_i$ is the integer obtained by truncating the fractional part of $\overline{b}_i$ and β_i is the fractional part of $\overline{b}_i$. By defining

$$\overline{a}_{ij} = a_{ij}^+ + a_{ij}^- \tag{10.10}$$

where

$$a_{ij}^+ = \begin{cases} \overline{a}_{ij} & \text{if } \ \overline{a}_{ij} \geq 0 \\ 0 & \text{if } \ \overline{a}_{ij} < 0 \end{cases} \tag{10.11}$$

$$a_{ij}^- = \begin{cases} 0 & \text{if } \ \overline{a}_{ij} \geq 0 \\ \overline{a}_{ij} & \text{if } \overline{a}_{ij} < 0 \end{cases} \tag{10.12}$$

Eq. (10.2) can be rewritten as

$$\sum_{j=1}^{n} (a_{ij}^+ + a_{ij}^-)y_j = \beta_i + (\hat{b}_i - x_i) \tag{10.13}$$

Here, by assumption, x_i is restricted to integer values while $\overline{b}_i$ is not an integer. Since $0 < \beta_i < 1$ and $\hat{b}_i$ is an integer, we can have the value of $\beta_i + (\hat{b}_i - x_i)$ either ≥ 0 or < 0. First, we consider the case where

$$\beta_i + (\hat{b}_i - x_i) \geq 0 \tag{10.14}$$

In this case, in order for x_i to be an integer, we must have

$$\beta_i + (\hat{b}_i - x_i) = \beta_i \quad \text{or} \quad \beta_i + 1 \quad \text{or} \quad \beta_i + 2, \ldots \tag{10.15}$$

Thus Eq. (10.13) gives

$$\sum_{j=1}^{n} (a_{ij}^+ + a_{ij}^-)y_j \geq \beta_i \tag{10.16}$$

Since $\overline{a}_{ij}$ are nonpositive and y_j are nonnegative by definition, we have

$$\sum_{j=1}^{n} a_{ij}^+ y_j \geq \sum_{j=1}^{n} (a_{ij}^+ - a_{ij}^-)y_j \tag{10.17}$$

and hence

$$\sum_{j=1}^{n} a_{ij}^+ y_j \geq \beta_i \tag{10.18}$$

Next, we consider the case where

$$\beta_i + (\hat{b}_i - x_i) < 0 \tag{10.19}$$

For x_i to be an integer, we must have (since $0 < \beta_i < 1$)

$$\beta_i + (\hat{b}_i - x_i) = -1 + \beta_i \quad \text{or} \quad -2 + \beta_i \quad \text{or} \quad -3 + \beta_i, \ldots \tag{10.20}$$

Thus Eq. (10.13) yields

$$\sum_{j=1}^{n} (a_{ij}^{+} + a_{ij}^{-}) y_j \leq \beta_i - 1 \tag{10.21}$$

Since

$$\sum_{j=1}^{n} a_{ij}^{-} y_j \leq \sum_{j=1}^{n} (a_{ij}^{+} + a_{ij}^{-}) y_j$$

we obtain

$$\sum_{j=1}^{n} a_{ij}^{-} y_j \leq \beta_i - 1 \tag{10.22}$$

Upon dividing this inequality by the negative quantity $(\beta_i - 1)$, we obtain

$$\frac{1}{\beta_i - 1} \sum_{j=1}^{n} a_{ij}^{-} y_j \geq 1 \tag{10.23}$$

Multiplying both sides of this inequality by $\beta_i > 0$, we can write the inequality (10.23) as

$$\frac{\beta_i}{\beta_i - 1} \sum_{j=1}^{n} a_{ij}^{-} y_j \geq \beta_i \tag{10.24}$$

Since one of the inequalities in (10.18) and (10.24) must be satisfied, the following inequality must hold true:

$$\sum_{j=1}^{n} a_{ij}^{+} y_j + \frac{\beta_i}{\beta_i - 1} \sum_{j=1}^{n} (a_{ij}^{-}) y_j \geq \beta_i \tag{10.25}$$

By introducing a slack variable s_i, we obtain the desired Gomory constraint as

$$s_i = \sum_{j=1}^{n} a_{ij}^{+} y_j + \frac{\beta_i}{\beta_i - 1} \sum_{j=1}^{n} \bar{a}_{ij} y_j - \beta_i \tag{10.26}$$

This constraint must be satisfied before the variable x_i becomes an integer. The slack variable s_i is not required to be an integer. At the optimal solution of the ordinary LP problem (given by Table 10.2), all $y_j = 0$ and hence Eq. (10.26) becomes

$$s_i = -\beta_i = \text{negative}$$

which can be seen to be infeasible. Hence the constraint Eq. (10.26) is added at the end of Table 10.2, and the dual simplex method applied. This procedure is repeated the required number of times until the optimal mixed integer solution is found.

Discussion. In the derivation of the Gomory constraint, Eq. (10.26), we have not made use of the fact that some of the variables (y_j) might be integer variables. We notice that any integer value can be added to or subtracted from the coefficient of $\bar{a}_{ik}$ ($= a_{ik}^{+} + a_{ik}^{-}$) of an integer variable y_k provided that we subtract or add, respectively, the same value to x_i in Eq. (10.13), that is,

$$\sum_{\substack{j=1 \\ j \neq k}}^{n} \bar{a}_{ij} y_j + (\bar{a}_{ik} \pm \delta) y_k = \beta_i + \hat{b}_i - (x_i \mp \delta) \qquad (10.27)$$

From Eq. (10.27), the same logic as was used in the derivation of Eqs. (10.18) and (10.24) can be used to obtain the same final equation, Eq. (10.26). Of course, the coefficients of integer variables y_k will be altered by integer amounts in Eq. (10.26). It has been established that to cut the feasible region as much as possible (through the Gomory constraint), we have to make the coefficients of integer variables y_k as small as possible. We can see that the smallest positive coefficient we can have for y_j in Eq. (10.13) is

$$\alpha_{ij} = \bar{a}_{ij} - \hat{a}_{ij}$$

and the largest negative coefficient as

$$1 - \alpha_{ij} = 1 - \bar{a}_{ij} + \hat{a}_{ij}$$

where $\hat{a}_{ij}$ is the integer obtained by truncating the fractional part of $\bar{a}_{ij}$ and α_{ij} is the fractional part. Thus we have a choice of two expressions, $(\bar{a}_{ij} - \hat{a}_{ij})$ and $(1 - \bar{a}_{ij} + \hat{a}_{ij})$, for the coefficients of y_j in Eq. (10.26). We choose the smaller one out of the two to make the Gomory constraint, Eq. (10.26), cut deeper into the original feasible space. Thus Eq. (10.26) can be rewritten as

$$s_i = \underbrace{\sum_j a_{ij}^+ y_j + \frac{\beta_i}{\beta_i - 1} \sum_j (+ a_{ij}^-) y_j}_{\text{for noninterger variables } y_j} \quad + \underbrace{\sum_j (\bar{a}_{ij} - \hat{a}_{ij}) y_j}_{\substack{\text{for integer variables } y_j \\ \text{and for } \bar{a}_{ij} - \hat{a}_{ij} \le \beta_i}}$$

$$+ \underbrace{\frac{\beta_i}{\beta_i - 1} \sum_j (1 - \bar{a}_{ij} + \hat{a}_{ij}) y_j}_{\substack{\text{for integer variables } y_j \\ \text{and for } \bar{a}_{ij} - \hat{a}_{ij} > \beta_i}} - \beta_i$$

where the slack variable s_i is not restricted to be an integer.

Example 10.2 Solve the problem of Example 10.1 with x_2 only restricted to take integer values.

SOLUTION

Step 1: Solve the LP problem by simplex method by neglecting the integer requirement. This gives the following optimal tableau:

Basic Variables	Coefficients of Variables					
	x_1	x_2	y_1	y_2	$-f$	$\bar{b}_i$
x_1	1	0	$\frac{11}{36}$	$\frac{1}{36}$	0	$\frac{11}{2}$
x_2	0	1	$-\frac{1}{12}$	$\frac{1}{12}$	0	$\frac{9}{2}$
$-f$	0	0	$\frac{7}{12}$	$\frac{5}{12}$	1	$\frac{69}{2}$

The noninteger solution given by this tableau is

$$x_1 = 5\tfrac{1}{2}, \quad x_2 = 4\tfrac{1}{2}, \quad y_1 = y_2 = 0 \quad \text{and} \quad f_{\min} = -34\tfrac{1}{2}$$

Step 2: Formulate a Gomory constraint. Since x_2 is the only variable that is restricted to take integer values, we construct the Gomory constraint for x_2. From the tableau of step 1, we obtain

$$x_2 = \bar{b}_2 - \bar{a}_{21} y_1 - \bar{a}_{22} y_2$$

where

$$\bar{b}_2 = \tfrac{9}{2}, \quad \bar{a}_{21} = -\tfrac{1}{12}, \quad \text{and} \quad \bar{a}_{22} = \tfrac{1}{12}$$

According to Eq. (10.3), we write $\bar{b}_2$ as $\bar{b}_2 = \hat{b}_2 + \beta_2$ where $\hat{b}_2 = 4$ and β_2

$= \frac{1}{2}$. Similarly, we write from Eq. (10.10)

$$\bar{a}_{21} = a_{21}^{+} + a_{21}^{-}$$

$$\bar{a}_{22} = a_{22}^{+} + a_{22}^{-}$$

where

$$a_{21}^{+} = 0, \qquad a_{21}^{-} = -\tfrac{1}{12} \text{ (since } \bar{a}_{21} \text{ is negative)}$$

$$a_{22}^{+} = \tfrac{1}{12},\ a_{22}^{-} = 0 \text{ (since } \bar{a}_{22} \text{ is nonnegative)}$$

The Gomory constraint can be expressed as [from Eq. (10.26)]:

$$s_2 - \sum_{j=1}^{2} a_{2j}^{+} y_j + \frac{\beta_2}{\beta_2 - 1} \sum_{j=1}^{2} a_{2j}^{-} y_j = -\beta_2$$

where s_2 is a slack variable which is not required to take integer values. By substituting the values of a_{ij}^{+}, a_{ij}^{-}, and β_i, this constraint can be written as

$$s_2 + \tfrac{1}{12} y_1 - \tfrac{1}{12} y_2 = -\tfrac{1}{2}$$

When this constraint is added to the tableau above, we obtain the following:

Basic Variables	Coefficients of Variables						
	x_1	x_2	y_1	y_2	$-f$	s_2	$\bar{b}_i$
x_1	1	0	$\frac{11}{36}$	$\frac{1}{36}$	0	0	$\frac{11}{2}$
x_2	0	1	$-\frac{1}{12}$	$\frac{1}{12}$	0	0	$\frac{9}{2}$
$-f$	0	0	$\frac{7}{12}$	$\frac{5}{12}$	1	0	$\frac{69}{2}$
s_2	0	0	$\frac{1}{12}$	$-\frac{1}{12}$	0	1	$-\frac{1}{2}$

Step 3: Apply dual simplex method to find a new optimum solution. Since $-\frac{1}{2}$ is the only negative $\bar{b}_i$ term, the pivot operation has to be done in s_2 row. Further, $\bar{a}_{ij}$ corresponding to y_2 column is the only negative coefficient in s_2 row and hence pivoting has to be done on this element, $-\frac{1}{12}$. The result of pivot operation is shown in the following tableau.

Basic Variables	Coefficients of Variables						
	x_1	x_2	y_1	y_2	$-f$	s_2	$\bar{b}_i$
x_1	1	0	$\frac{1}{3}$	0	0	$\frac{1}{3}$	$\frac{16}{3}$
x_2	0	1	0	0	0	1	4
$-f$	0	0	1	0	1	5	32
y_2	0	0	−1	1	0	−12	6

This tableau gives the desired integer solution as

$$x_1 = 5\tfrac{1}{2}, \quad x_2 = 4, \quad y_2 = 6, \quad y_1 = 0, \quad s_2 = 0, \quad \text{and} \quad f_{\min} = -32$$

10.4 BALAS' ALGORITHM FOR ZERO–ONE PROGRAMMING PROBLEMS

When all the variables of a LP problem are constrained to take values of 0 or 1 only, we have a zero–one (or binary) LP problem. A study of the various techniques available for solving zero–one programming problems is important because of the following reasons.

1. As we shall see later in this chapter (Section 10.5), a certain class of integer nonlinear programming problems can be converted into equivalent zero–one LP problems,
2. A wide variety of industrial, management, and engineering problems can be formulated as zero–one problems. For example, in structural control, the problem of selecting optimal locations of actuators (or dampers) can be formulated as a zero–one problem. In this case, if a variable is zero or 1, it indicates the absence or presence of the actuator, respectively, at a particular location [10.31].

The zero–one LP problems can be solved by using any of the general integer LP techniques like Gomory's cutting plane method and Land and Doig's branch-and-bound method by introducing the additional constraint that all the variables must be less than or equal to 1. This additional constraint will restrict each of the variables to take a value of either zero (0) or one (1). Since the cutting plane and the branch-and-bound algorithms were developed primarily to solve a general integer LP problem, they do not take advantage of the special features of zero–one LP problems. Thus several methods have been proposed to solve zero–one LP problems more efficiently. In this section we present an algorithm developed by Balas (in 1965) for solving LP problems with binary variables only [10.9].

If there are n binary variables in a problem, an explicit enumeration process will involve testing 2^n possible solutions against the stated constraints and the objective function. In Balas method, all the 2^n possible solutions are enumerated, explicitly or implicitly. The efficiency of the method arises out of the clever strategy it adopts in selecting only a few solutions for explicit enumeration.

The method starts by setting all the n variables equal to zero and consists of a systematic procedure of successively assigning to certain variables the value 1, in such a way that after trying a (small) part of all the 2^n possible combinations, one obtains either an optimal solution or evidence of the fact that no feasible solution exists. The only operations required in the computation are additions and subtractions, and hence the round-off errors will not be there. For this reason the method is some times referred to as *additive algorithm*.

Standard Form of the Problem. To describe the algorithm, consider the following form of the LP problem with zero–one variables:

$$\text{Find } \mathbf{X} = \begin{Bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{Bmatrix} \text{ such that } f(\mathbf{X}) = \mathbf{C}^T\mathbf{X} \rightarrow \text{minimum}$$

subject to (10.28)

$$\mathbf{AX} + \mathbf{Y} = \mathbf{B}$$
$$x_i = 0 \quad \text{or} \quad 1$$
$$\mathbf{Y} \geq \mathbf{0}$$

where

$$\mathbf{C} = \begin{Bmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \end{Bmatrix} \geq \mathbf{0}, \quad \mathbf{Y} = \begin{Bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{Bmatrix}, \quad \mathbf{B} = \begin{Bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{Bmatrix}$$

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

where **Y** is the vector of slack variables and c_i and a_{ij} need not be integers.

Initial Solution. An initial solution for the problem stated in Eqs. (10.28) can be taken as

$$f_0 = 0$$
$$x_i = 0, \qquad i = 1,2,\ldots,n \tag{10.29}$$
$$\mathbf{Y}^{(0)} = \mathbf{B}$$

If $\mathbf{B} \geq \mathbf{0}$, this solution will be feasible and optimal since $\mathbf{C} \geq \mathbf{0}$ in Eqs. (10.28). In this case there is nothing more to be done as the starting solution itself happens to be optimal. On the other hand, if some of the components b_j are negative, the solution given by Eqs. (10.29) will be optimal (since $\mathbf{C} \geq \mathbf{0}$) but infeasible. Thus the method starts with an optimal (actually better than optimal) and infeasible solution. The algorithm forces this solution toward feasibility while keeping it optimal all the time. This is the reason why Balas called his method the *pseudo dual simplex method*. The word *pseudo* has been used since the method is similar to the dual simplex method only as far as the starting solution is concerned and the subsequent procedure has no similarity at all with the dual simplex method. The details can be found in Ref. [10.9].

INTEGER NONLINEAR PROGRAMMING

10.5 INTEGER POLYNOMIAL PROGRAMMING

Watters [10.2] has developed a procedure for converting integer polynomial programming problems to zero–one LP problems. The resulting zero–one LP problem can be solved conveniently by the Balas method discussed in Section 10.4. Consider the optimization problem:

$$\text{Find } \mathbf{X} = \begin{Bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{Bmatrix} \text{ which minimizes } f(\mathbf{X})$$

subject to the constraints (10.30)

$$g_j(\mathbf{X}) \leq 0, \qquad j = 1,2,\ldots,m$$
$$x_i = \text{integer}, \qquad i = 1,2,\ldots,n$$

where f and g_j, $j = 1,2,\ldots,m$, are polynomials in the variables $x_1, x_2, \ldots, x_n$. A typical term in the polynomials can be represented as

$$c_k \prod_{l=1}^{n_k} (x_l)^{a_{kl}} \tag{10.31}$$

where c_k is a constant, a_{kl} a nonnegative constant exponent, and n_k the number of variables appearing in the kth term. We shall convert the integer polynomial programming problem stated in Eq. (10.30) into an equivalent zero–one LP problem in two stages. In the first stage we see how an integer variable, x_i, can be represented by an equivalent system of zero–one (binary) variables. We consider the conversion of a zero–one polynomial programming problem into a zero–one LP problem in the second stage.

10.5.1 Representation of an Integer Variable by an Equivalent System of Binary Variables

Let x_i be any integer variable whose upper bound is given by u_i so that

$$x_i \leq u_i < \infty \tag{10.32}$$

We assume that the value of the upper bound u_i can be determined from the constraints of the given problem.

We know that in the decimal number system, an integer p is represented as

$$p = p_0 + 10^1 p_1 + 10^2 p_2 + \dots, 0 \leq p_i \leq (10 - 1 = 9)$$
$$\text{for} \quad i = 0,1,2,\dots$$

and written as $p = \cdots p_2 p_1 p_0$ by neglecting the zeros to the left. For example, we write the number $p = 008076$ as 8076 to represent $p = 6 + (10^1)$ $7 + (10^2)(0) + (10^3)8 + (10^4)0 + (10^5)0$. In a similar manner, the integer p can also be represented in binary number system as

$$p = q_0 + 2^1 q_1 + 2^2 q_2 + 2^3 q_3 + \cdots$$

where $0 \leq q_i \leq (2 - 1 = 1)$ for $i = 0,1,2,\dots$.

In general, if $y_i^{(0)}, y_i^{(1)}, y_i^{(2)}, \dots$ denote binary numbers (which can take a value of 0 or 1), the variable x_i can be expressed as

$$x_i = \sum_{k=0}^{N_i} 2^k y_i^{(k)} \tag{10.33}$$

where N_i is the smallest integer such that

$$\frac{u_i + 1}{2} \leq 2^{N_i} \tag{10.34}$$

Thus the value of N_i can be selected for any integer variable x_i once its upper bound u_i is known. For example, for the number 97, we can take $u_i = 97$ and hence the relation

$$\frac{u_i + 1}{2} = \frac{98}{2} = 49 \leq 2^{N_i}$$

is satisfied for $N_i \geq 6$. Hence by taking $N_i = 6$, we can represent u_i as

$$97 = q_0 + 2^1 q_1 + 2^2 q_2 + 2^3 q_3 + 2^4 q_4 + 2^5 q_5 + 2^6 q_6$$

where $q_0 = 1$, $q_1 = q_2 = q_3 = q_4 = 0$, and $q_5 = q_6 = 1$. A systematic method of finding the values of $q_0, q_1, q_2, \ldots$ is given below.

Method of Finding $q_0, q_1, q_2, \ldots$. Let M be the given positive integer. To find its binary representation $q_n q_{n-1} \cdots q_1\, q_0$, we compute the following recursively:

$$\begin{aligned}
b_0 &= M \\
b_1 &= \frac{b_0 - q_0}{2} \\
b_2 &= \frac{b_1 - q_1}{2} \\
&\ \ \vdots \\
b_k &= \frac{b_{k-1} - q_{k-1}}{2}
\end{aligned} \tag{10.35}$$

where $q_k = 1$ if b_k is odd and $q_k = 0$ if b_k is even. The procedure terminates when $b_k = 0$.

Equation (10.33) guarantees that x_i can take any feasible integer value less than or equal to u_i. The use of Eq. (10.33) in the problem stated in Eq. (10.30) will convert the integer programming problem into a binary one automatically. The only difference is that the binary problem will have $N_1 + N_2 + \cdots + N_n$ zero–one variables instead of the n original integer variables.

10.5.2 Conversion of a Zero–One Polynomial Programming Problem into a Zero–One LP Problem

The conversion of a polynomial programming problem into a LP problem is based on the fact that

$$x_i^{a_{ki}} \equiv x_i \tag{10.36}$$

if x_i is a binary variable (0 or 1) and a_{ki} is a positive exponent. If $a_{ki} = 0$, then obviously the variable x_i will not be present in the kth term. The use of Eq. (10.36) permits us to write the kth term of the polynomial, Eq. (10.31), as

$$c_k \prod_{l=1}^{n_k} (x_l)^{a_{kl}} = c_k \prod_{l=1}^{n_k} x_l = c_k\,(x_1, x_2, \ldots, x_{n_k}) \tag{10.37}$$

Since each of the variables $x_1, x_2, \ldots$ can take a value of either 0 or 1, the product $(x_1 \quad x_2 \quad \cdots \quad x_{n_k})$ also will take a value of 0 or 1. Hence by defining a binary variable y_k as

$$y_k = x_1 x_2 \cdots x_{n_k} = \prod_{l=1}^{n_k} x_l \tag{10.38}$$

the kth term of the polynomial simply becomes $c_k y_k$. However, we need to add the following constraints to ensure that $y_k = 1$ when all $x_i = 1$ and zero otherwise:

$$y_k \geq \left(\sum_{i=1}^{n_k} x_i \right) - (n_k - 1) \tag{10.39}$$

$$y_k \leq \frac{1}{n_k} \left(\sum_{i=1}^{n_k} x_i \right) \tag{10.40}$$

It can be seen that if all $x_i = 1$, $\Sigma_{i=1}^{n_k} x_i = n_k$, and Eqs. (10.39) and (10.40) yield

$$y_k \geq 1 \tag{10.41}$$

$$y_k \leq 1 \tag{10.42}$$

which can be satisfied only if $y_k = 1$. If at least one $x_i = 0$, we have $\Sigma_{i=1}^{n_k} x_i < n_k$, and Eqs. (10.39) and (10.40) give

$$y_k \geq -(n_k - 1) \tag{10.43}$$

$$y_k < 1 \tag{10.44}$$

Since n_k is a positive integer, the only way to satisfy Eqs. (10.43) and (10.44) under all circumstances is to have $y_k = 0$.

This procedure of converting an integer polynomial programming problem into an equivalent zero–one LP problem can always be applied, at least in theory.

10.6 BRANCH-AND-BOUND METHOD

The branch-and-bound method is very effective in solving mixed-integer linear and nonlinear programming problems. The method was originally developed by Land and Doig [10.8] to solve integer linear programming problems and was later modified by Dakin [10.23]. Subsequently, the method has been extended to solve nonlinear mixed-integer programming problems. To see the basic solution procedure, consider the following nonlinear mixed-integer programming problem:

$$\text{Minimize } f(\mathbf{X}) \tag{10.45}$$

subject to

$$g_j(\mathbf{X}) \geq 0, \qquad j = 1,2,\ldots,m \tag{10.46}$$

$$h_k(\mathbf{X}) = 0, \qquad k = 1,2,\ldots,p \tag{10.47}$$

$$x_j = \text{integer}, \qquad j = 1,2,\ldots,n_0 \ (n_0 \leq n) \tag{10.48}$$

where $\mathbf{X} = \{x_1 \quad x_2 \quad \cdots \quad x_n\}^T$. Note that in the design vector **X**, the first n_0 variables are identified as the integer variables. If $n_0 = n$, the problem becomes an all-integer programming problem. A design vector **X** is called a *continuous feasible solution* if **X** satisfies constraints (10.46) and (10.47). A design vector **X** that satisfies all the constraints, Eqs. (10.46) to (10.48), is called an *integer feasible solution*.

The simplest method of solving an integer optimization problem involves enumerating all integer points, discarding infeasible ones, evaluating the objective function at all integer feasible points, and identifying the point that has the best objective function value. Although such an exhaustive search in the solution space is simple to implement, it will be computationally expensive even for moderate-size problems. The branch-and-bound method can be considered as a refined enumeration method in which most of the nonpromising integer points are discarded without testing them. Also note that the process of complete enumeration can be used only if the problem is an all-integer programming problem. For mixed-integer problems in which one or more variables may assume continuous values, the process of complete enumeration cannot be used.

In the branch-and-bound method, the integer problem is not directly solved. Rather, the method first solves a continuous problem obtained by relaxing the integer restrictions on the variables. If the solution of the continuous problem happens to be an integer solution, it represents the optimum solution of the integer problem. Otherwise, at least one of the integer variables, say x_i, must assume a nonintegral value. If x_i is not an integer, we can always find an integer $[x_i]$ such that

$$[x_i] < x_i < [x_i] + 1 \tag{10.49}$$

Then two subproblems are formulated, one with the additional upper bound constraint

$$x_i \leq [x_i] \tag{10.50}$$

and another with the lower bound constraint

$$x_i \geq [x_i] + 1 \tag{10.51}$$

The process of finding these subproblems is called *branching*.

The branching process eliminates some portion of the continuous space that is not feasible for the integer problem, while ensuring that none of the integer feasible solutions are eliminated. Each of these two subproblems are solved again as a continuous problem. It can be seen that the solution of a continuous problem forms a *node* and from each node two branches may originate.

The process of branching and solving a sequence of continuous problems discussed above is continued until an integer feasible solution is found for one of the two continuous problems. When such a feasible integer solution is found, the corresponding value of the objective function becomes an upper bound on the minimum value of the objective function. At this stage we can eliminate from further consideration all the continuous solutions (nodes) whose objective function values are larger than the upper bound. The nodes that are eliminated are said to have been *fathomed* because it is not possible to find a better integer solution from these nodes (solution spaces) than what we have now. The value of the upper bound on the objective function is updated whenever a better bound is obtained.

It can be seen that a node can be fathomed if any of the following conditions are true:

1. The continuous solution is an integer feasible solution.
2. The problem does not have a continuous feasible solution.
3. The optimal value of the continuous problem is larger than the current upper bound.

The algorithm continues to select a node for further branching until all the nodes have been fathomed. At that stage, the particular fathomed node that has the integer feasible solution with the lowest value of the objective function gives the optimum solution of the original nonlinear integer programming problem.

Example 10.3 Solve the following LP problem using the branch-and-bound method:

$$\text{Maximize } f = 3x_1 + 4x_2$$

subject to (E$_1$)

$$7x_1 + 11x_2 \leq 88, \qquad 3x_1 - x_2 \leq 12, \qquad x_1 \geq 0, \qquad x_2 \geq 0$$

$$x_i = \text{integer}, \qquad i = 1,2 \tag{E$_2$}$$

SOLUTION The various steps of the procedure are illustrated using graphical method.

Step 1: First the problem is solved as a continuous variable problem [without Eq. (E_2)] to obtain:

Problem (E_1): Fig. 10.2; ($x_1^* = 5.5$, $x_2^* = 4.5$, $f^* = 34.5$)

Step 2: The branching process, with integer bounds on x_1, yields the problems:

$$\text{Maximize } f = 3x_1 + 4x_2$$

subject to (E_3)

$$7x_1 + 11x_2 \leq 88, \quad 3x_1 - x_2 \leq 12, \quad x_1 \leq 5, \quad x_2 \geq 0$$

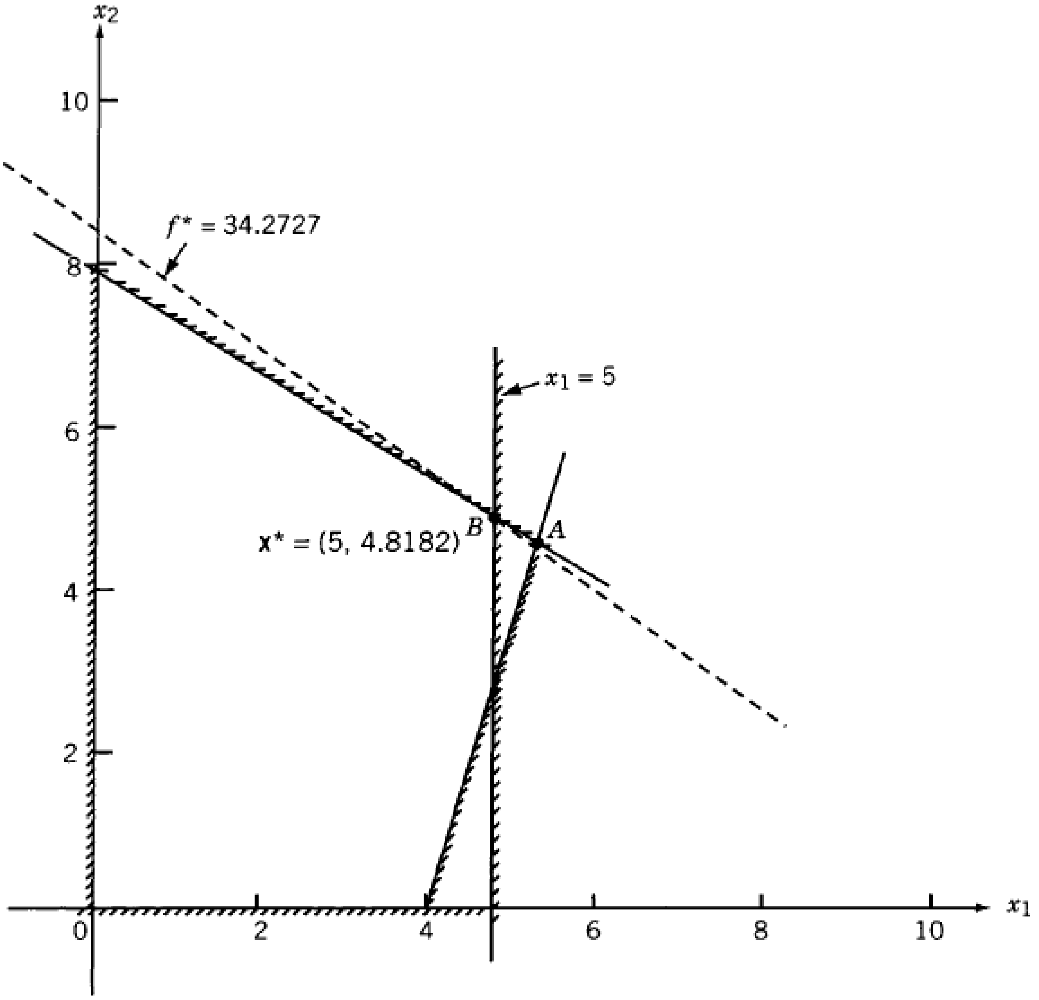


Figure 10.4 Graphical solution of problem (E_3).

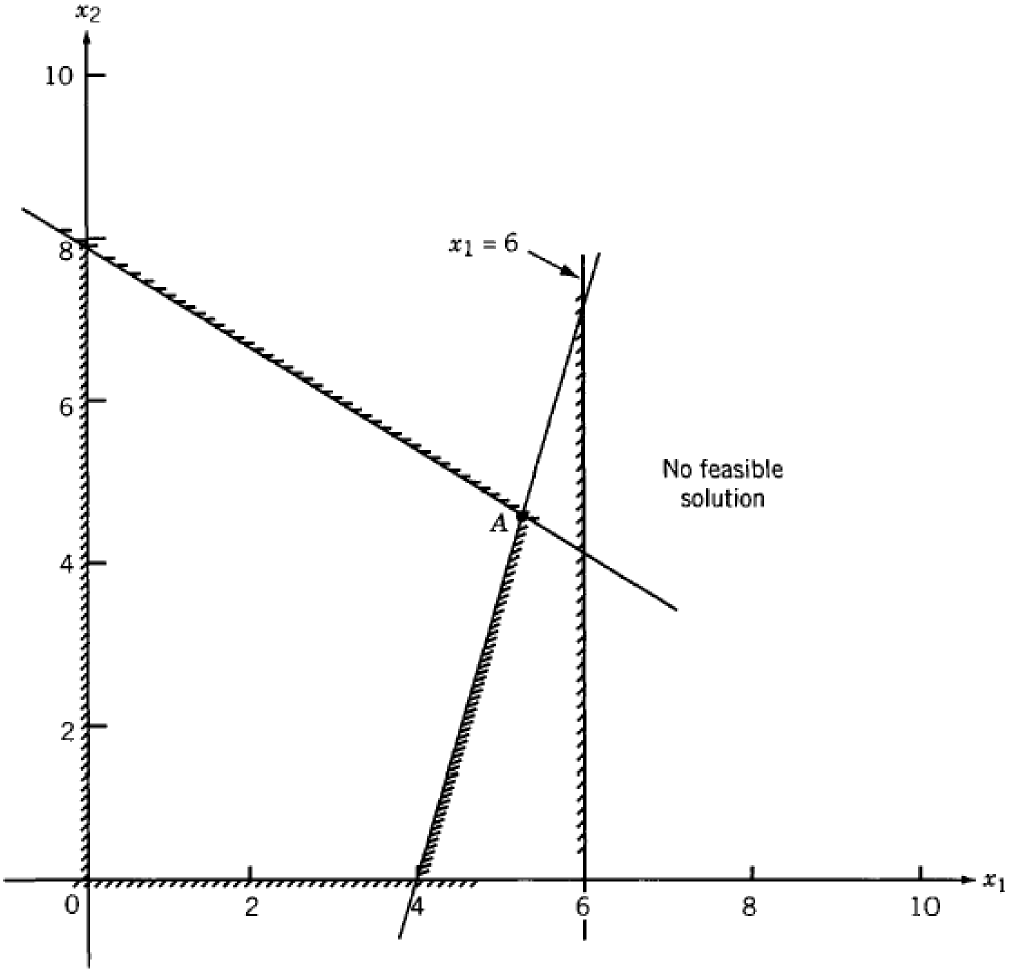


Figure 10.5 Graphical solution of problem (E_4).

and

$$\text{Maximize } f = 3x_1 + 4x_2$$

subject to $\qquad$ (E_4)

$$7x_1 + 11x_2 \leq 88, \quad 3x_1 - x_2 \leq 12, \quad x_1 \geq 6, \quad x_2 \geq 0$$

The solutions of problems (E_3) and (E_4) are given by:

Problem (E_3): Fig. 10.4; ($x_1^* = 5$, $x_2^* = 4.8182$, $f^* = 34.2727$)
Problem (E_4): Fig. 10.5; no feasible solution exists.

Step 3: The next branching process, with integer bounds on x_2, leads to the following problems:

$$\text{Maximize } f = 3x_1 + 4x_2$$

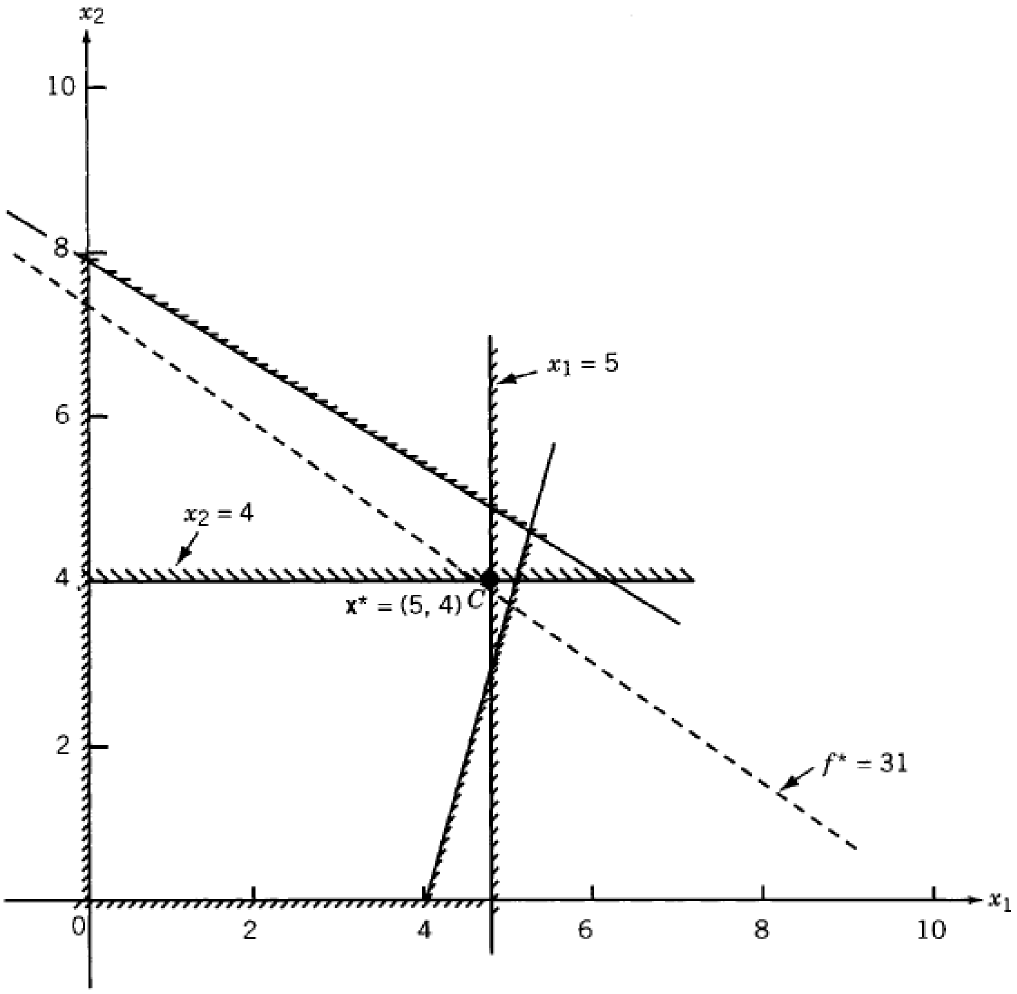


Figure 10.6 Graphical solution of problem (E_5).

subject to (E_5)

$$7x_1 + 11x_2 \leq 88, \quad 3x_1 - x_2 \leq 12, \quad x_1 \leq 5, \quad x_2 \leq 4$$

and

$$\text{Maximize } f = 3x_1 + 4x_2$$

subject to (E_6)

$$7x_1 + 11x_2 \leq 88, \quad 3x_1 - x_2 \leq 12, \quad x_1 \leq 5, \quad x_2 \geq 5$$

The solutions of problems (E_5) and (E_6) are given by:

Problem (E_5): Fig. 10.6; ($x_1^* = 5, x_2^* = 4, f^* = 31$)
Problem (E_6): Fig. 10.7; ($x_1^* = 0, x_2^* = 8, f^* = 32$)

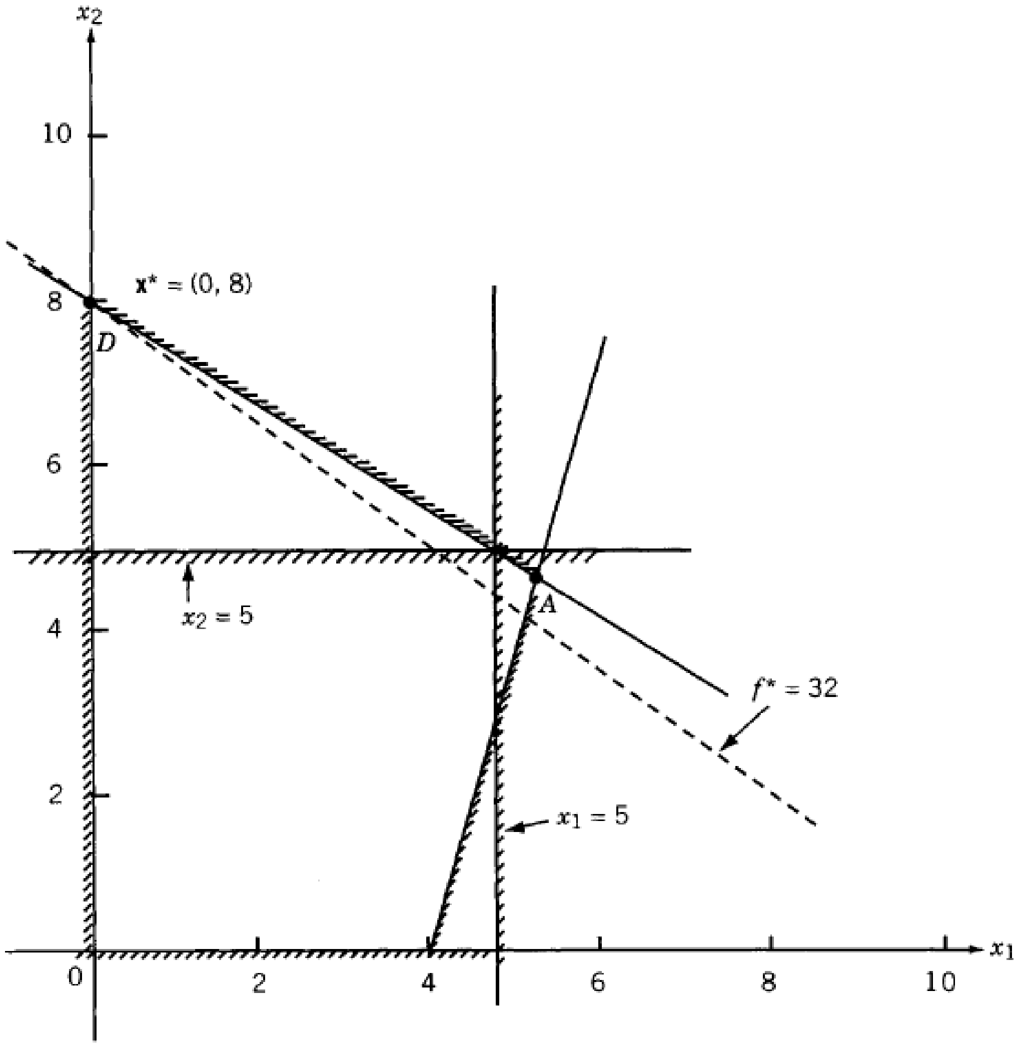


Figure 10.7 Graphical solution of problem (E_6).

Since both the variables assumed integer values, the optimum solution of the integer LP problem, Eqs. (E_1) and (E_2), is given by ($x_1^* = 0$, $x_2^* = 8$, $f^* = 32$).

Example 10.4 Find the solution of the welded beam problem of Section 7.22.3 by treating it as a mixed-integer nonlinear programming problem by requiring x_3 and x_4 to take integer values.

SOLUTION The solution of this problem using the branch-and-bound method was reported in Ref. [10.25]. The optimum solution of the continuous variable nonlinear programming problem is given by

$$\mathbf{X}^* = \{0.24, 6.22, 8.29, 0.24\}^T, \qquad f^* = 2.38$$

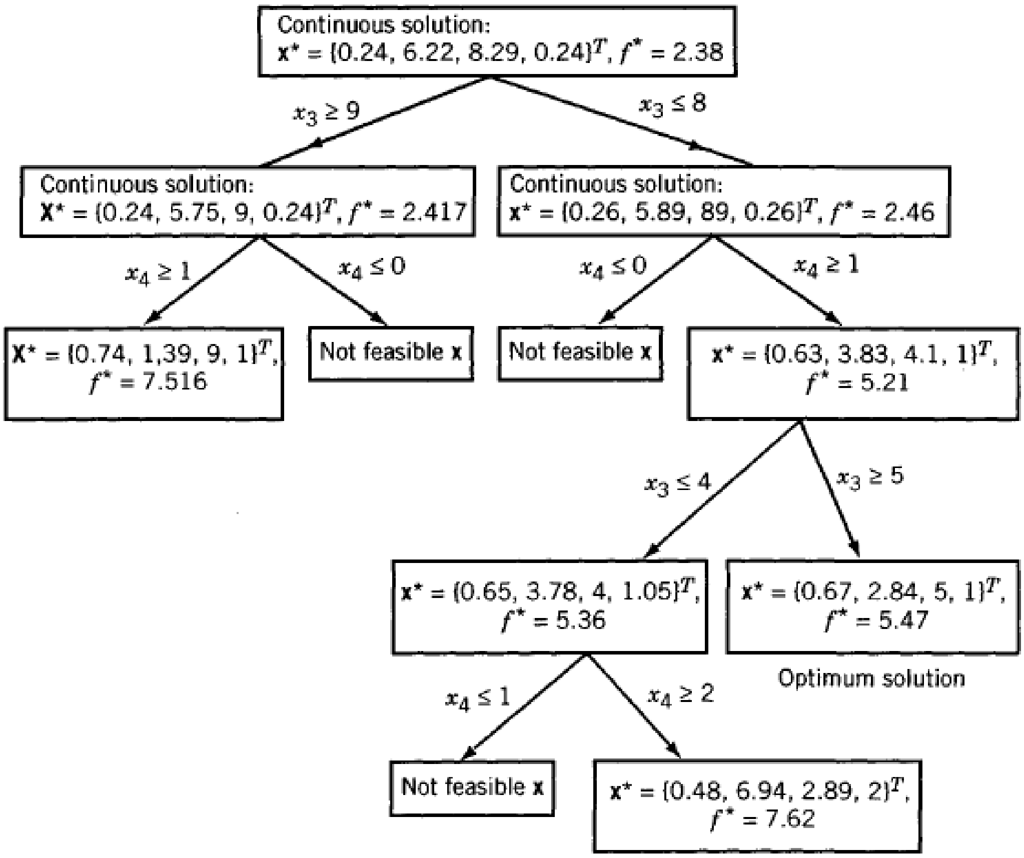

Continuous solution:
x* = {0.24, 6.22, 8.29, 0.24}^T, f* = 2.38
x3 ≥ 9
x3 ≤ 8
Continuous solution:
X* = {0.24, 5.75, 9, 0.24}^T, f* = 2.417
Continuous solution:
x* = {0.26, 5.89, 89, 0.26}^T, f* = 2.46
x4 ≥ 1
x4 ≤ 0
x4 ≤ 0
x4 ≥ 1
X* = {0.74, 1,39, 9, 1}^T, f* = 7.516
Not feasible x
Not feasible x
x* = {0.63, 3.83, 4.1, 1}^T, f* = 5.21
x3 ≤ 4
x3 ≥ 5
x* = {0.65, 3.78, 4, 1.05}^T, f* = 5.36
x* = {0.67, 2.84, 5, 1}^T, f* = 5.47
Optimum solution
x4 ≤ 1
x4 ≥ 2
Not feasible x
x* = {0.48, 6.94, 2.89, 2}^T, f* = 7.62

Figure 10.8 Solution of the welded beam problem using branch-and-bound method. [10.25] (Reprinted with permission from ASME).

Next, the branching problems, with integer bounds on x_3, are solved and the procedure is continued until the desired optimum solution is found. The results are shown in Fig. 10.8.

10.7 SEQUENTIAL LINEAR DISCRETE PROGRAMMING

Let the nonlinear programming problem with discrete variables be stated as follows:

$$\text{Minimize } f(\mathbf{X}) \tag{10.52}$$

subject to

$$g_j(\mathbf{X}) \le 0, \quad j = 1,2,\ldots,m \tag{10.53}$$

$$h_k(\mathbf{X}) = 0, \quad k = 1,2,\ldots,p \tag{10.54}$$

$$x_i \in \{d_{i1}, d_{i2}, \ldots, d_{iq}\}, \qquad i = 1,2,\ldots,n_0 \tag{10.55}$$

$$x_i^{(l)} \leq x_i \leq x_i^{(u)}, \qquad i = n_0 + 1, \quad n_0 + 2, \ldots, n \tag{10.56}$$

where the first n_0 design variables are assumed to be discrete, d_{ij} is the jth discrete value for the variable i, and $\mathbf{X} = \{x_1 \quad x_2 \quad \cdots \quad x_n\}^T$. It is possible to find the solution of this problem by solving a series of mixed-integer linear programming problems.

The nonlinear expressions in Eqs. (10.52) to (10.54) are linearized about a point $\mathbf{X}^0$ using a first-order Taylor's series expansion and the problem is stated as:

$$\text{Minimize } f(\mathbf{X}) \approx f(\mathbf{X}^0) + \nabla f(\mathbf{X}^0)\, \delta\mathbf{X} \tag{10.57}$$

subject to

$$g_j(\mathbf{X}) \approx g_j(\mathbf{X}^0) + \nabla g_j(\mathbf{X}^0)\, \delta\mathbf{X} \leq 0, \qquad j = 1,2,\ldots,m \tag{10.58}$$

$$h_k(\mathbf{X}) \approx h_k(\mathbf{X}^0) + \nabla h_k(\mathbf{X}^0)\, \delta\mathbf{X} = 0, \qquad k = 1,2,\ldots,p \tag{10.59}$$

$$x_i^0 + \delta x_i \in \{d_{i1}, d_{i2}, \ldots, d_{iq}\}, \qquad i = 1,2,\ldots,n_0 \tag{10.60}$$

$$x_i^{(l)} \leq x_i^0 + \delta x_i \leq x_i^{(u)}, \qquad i = n_0 + 1, n_0 + 2, \ldots, n \tag{10.61}$$

$$\delta\mathbf{X} = \mathbf{X} - \mathbf{X}^0 \tag{10.62}$$

The problem stated in Eqs. (10.57) to (10.62) cannot be solved using mixed-integer linear programming techniques since some of the design variables are discrete and noninteger. The discrete variables are redefined as [10.26]

$$x_i = y_{i1}d_{i1} + y_{i2}d_{i2} + \cdots + y_{iq}d_{iq} = \sum_{j=1}^{q} y_{ij}d_{ij}, \qquad i = 1,2,\ldots,n_0 \tag{10.63}$$

with

$$y_{i1} + y_{i2} + \cdots + y_{iq} = \sum_{j=1}^{q} y_{ij} = 1 \tag{10.64}$$

$$y_{ij} = 0 \text{ or } 1, \qquad i = 1,2,\ldots,n_0, \quad j = 1,2,\ldots,q \tag{10.65}$$

Using Eqs. (10.63) to (10.65) in Eqs. (10.57) to (10.62), we obtain

$$\text{Minimize } f(\mathbf{X}) \approx f(\mathbf{X}^0) + \sum_{i=1}^{n_0} \frac{\partial f}{\partial x_i} \left(\sum_{j=1}^{q} y_{ij}d_{ij} - x_i^0 \right) + \sum_{i=n_0+1}^{n} \frac{\partial f}{\partial x_i} (x_i - x_i^0) \tag{10.66}$$

subject to

$$g_j(\mathbf{X}) \approx g_j(\mathbf{X}^0) + \sum_{i=1}^{n_0} \frac{\partial g_j}{\partial x_i} \left(\sum_{l=1}^{n_0} y_{il} d_{il} - x_i^0 \right) + \sum_{i=n_0+1}^{n} \frac{\partial g_j}{\partial x_i} (x_i - x_i^0) \le 0,$$

$$j = 1,2,\ldots,m \tag{10.67}$$

$$h_k(\mathbf{X}) \approx h_k(\mathbf{X}^0) + \sum_{i=1}^{n_0} \frac{\partial h_k}{\partial x_i} \left(\sum_{l=1}^{n_0} y_{il} d_{il} - x_i^0 \right) + \sum_{i=n_0+1}^{n} \frac{\partial h_k}{\partial x_i} (x_i - x_i^0) = 0,$$

$$k = 1,2,\ldots,p \tag{10.68}$$

$$\sum_{j=1}^{q} y_{ij} = 1, \qquad i = 1,2,\ldots,n_0 \tag{10.69}$$

$$y_{ij} = 0 \text{ or } 1, \qquad i = 1,2,\ldots,n_0, \quad j = 1,2,\ldots,q \tag{10.70}$$

$$x_i^{(l)} \le x_i^0 + \delta x_i \le x_i^{(u)}, \qquad i = n_0 + 1, n_0 + 2, \ldots, n \tag{10.71}$$

The problem stated in Eqs. (10.66) to (10.71) can now be solved as a mixed-integer LP problem by treating both y_{ij} ($i = 1,2,\ldots,n_0, j = 1,2,\ldots,q$) and x_i ($i = n_0 + 1, n_0 + 2, \ldots, n$) as unknowns.

In practical implementation, the initial linearization point $\mathbf{X}^0$ is to be selected carefully. In many cases the solution of the discrete problem is expected to lie in the vicinity of the continuous optimum. Hence the original problem can be solved as a continuous nonlinear programming problem (by ignoring the discrete nature of the variables) using any of the standard nonlinear programming techniques. If the resulting continuous optimum solution happens to be a feasible discrete solution, it can be used as $\mathbf{X}^0$. Otherwise, the values of x_i from the continuous optimum solution are rounded (in a direction away from constraint violation) to obtain an initial feasible discrete solution $\mathbf{X}^0$. Once the first linearized discrete problem is solved, the subsequent linearizations can be made using the result of the previous optimization problem.

Example 10.5 **[*10.26*]**

$$\text{Minimize } f(\mathbf{X}) = 2x_1^2 + 3x_2^2$$

subject to

$$g(\mathbf{X}) = \frac{1}{x_1} + \frac{1}{x_2} - 4 \le 0$$

$$x_1 \in \{0.3,0.7,0.8,1.2,1.5,1.8\}$$

$$x_2 \in \{0.4,0.8,1.1,1.4,1.6\}$$

SOLUTION In this example, the set of discrete values of each variable is truncated by allowing only three values—its current value, the adjacent higher value, and the adjacent lower value—for simplifying the computations. Using $\mathbf{X}^0 = \begin{Bmatrix} 1.2 \\ 1.1 \end{Bmatrix}$, we have

$$f(\mathbf{X}^0) = 6.51, \qquad g(\mathbf{X}^0) = -2.26$$

$$\nabla f(\mathbf{X}^0) = \begin{Bmatrix} 4x_1 \\ 6x_2 \end{Bmatrix}_{\mathbf{X}^0} = \begin{Bmatrix} 4.8 \\ 6.6 \end{Bmatrix}, \qquad \nabla g(\mathbf{X}^0) = \begin{Bmatrix} -\dfrac{1}{x_1^2} \\ -\dfrac{1}{x_2^2} \end{Bmatrix}_{\mathbf{X}^0} = \begin{Bmatrix} -0.69 \\ -0.83 \end{Bmatrix}$$

Now

$$\begin{aligned}
x_1 &= y_{11}(0.8) + y_{12}(1.2) + y_{13}(1.5) \\
x_2 &= y_{21}(0.8) + y_{22}(1.1) + y_{23}(1.4) \\
\delta x_1 &= y_{11}(0.8 - 1.2) + y_{12}(1.2 - 1.2) + y_{13}(1.5 - 1.2) \\
\delta x_2 &= y_{21}(0.8 - 1.1) + y_{22}(1.1 - 1.1) + y_{23}(1.4 - 1.1) \\
f &\approx 6.51 + \{4.8 \quad 6.6\} \begin{Bmatrix} -0.4y_{11} + 0.3y_{13} \\ -0.3y_{21} + 0.3y_{23} \end{Bmatrix} \\
g &\approx -2.26 + \{-0.69 \quad -0.83\} \begin{Bmatrix} -0.4y_{11} + 0.3y_{13} \\ -0.3y_{21} + 0.3y_{23} \end{Bmatrix}
\end{aligned}$$

Thus the first approximate problem becomes (in terms of the unknowns y_{11}, y_{12}, y_{13}, y_{21}, y_{22}, and y_{23}):

$$\text{Minimize } f = 6.51 - 1.92y_{11} + 1.44y_{13} - 1.98y_{21} + 1.98y_{23}$$

subject to

$$\begin{gathered}
-2.26 + 0.28y_{11} + 0.21y_{13} + 0.25y_{21} - 0.25y_{23} \le 0 \\
y_{11} + y_{12} + y_{13} = 1 \\
y_{21} + y_{22} + y_{23} = 1 \\
y_{ij} = 0 \text{ or } 1, \qquad i = 1,2, \quad j = 1,2,3
\end{gathered}$$

In this problem, there are only nine possible solutions and hence they can all be enumerated and the optimum solution can be found as

$$y_{11} = 1, \quad y_{12} = 0, \quad y_{13} = 0, \quad y_{21} = 1, \quad y_{22} = 0, \quad y_{23} = 0$$

Thus the solution of the first approximate problem, in terms of original variables, is given by

$$x_1 = 0.8, \quad x_2 = 0.8, f(\mathbf{X}) = 2.61, \quad \text{and} \quad g(\mathbf{X}) = -1.5$$

This point can be used to generate a second approximate problem and the process can be repeated until the final optimum solution is found.

10.8 GENERALIZED PENALTY FUNCTION METHOD

The solution of an integer nonlinear programming problem, based on the concept of penalty functions, was originally suggested by Gellatly and Marcal in 1967 [10.5]. This approach was later applied by Gisvold and Moe [10.4] and Shin et al. [10.24] to solve some design problems that have been formulated as nonlinear mixed-integer programming problems. The method can be considered as an extension of the interior penalty function approach considered in Section 7.13. To see the details of the approach, let the problem be stated as follows:

$$\text{Find } \mathbf{X} = \begin{Bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{Bmatrix} = \begin{Bmatrix} \mathbf{X}_d \\ \mathbf{X}_c \end{Bmatrix} \text{ which minimizes } f(\mathbf{X})$$

subject to the constraints (10.72)

$$g_j(\mathbf{X}) \geqslant 0, \qquad j = 1,2,\ldots,m$$
$$\mathbf{X}_c \in S_c \quad \text{and} \quad \mathbf{X}_d \in S_d,$$

where the vector of variables ($\mathbf{X}$) is composed of two vectors $\mathbf{X}_d$ and $\mathbf{X}_c$, with $\mathbf{X}_d$ representing the set of integer variables and $\mathbf{X}_c$ representing the set of continuous variables. Notice that $\mathbf{X}_c$ will not be there if all the variables are constrained to take only integer values and $\mathbf{X}_d$ will not be there if none of the variables is restricted to take only integer values. The sets S_c and S_d denote the feasible sets of continuous and integer variables, respectively. To extend the interior penalty function approach to solve the present problem, given by Eq. (10.72), we first define the following transformed problem.

$$\text{Minimize } \phi_k(\mathbf{X},r_k,s_k)$$

where

$$\phi_k(\mathbf{X},r_k,s_k) = f(\mathbf{X}) + r_k \sum_{j=1}^{m} G_j[g_j(\mathbf{X})] + s_k Q_k(\mathbf{X}_d) \qquad (10.73)$$

In this equation, r_k is a weighing factor (penalty parameter) and

$$r_k \sum_{j=1}^{m} G_j[g_j(\mathbf{X})]$$

is the contribution of the constraints to the ϕ_k function, which can be taken as

$$r_k \sum_{j=1}^{m} G_j[g_j(\mathbf{X})] = +r_k \sum_{j=1}^{m} \frac{1}{g_j(\mathbf{X})} \tag{10.74}$$

It can be noted that this term is positive for all $\mathbf{X}$ satisfying the relations $g_j(\mathbf{X}) > 0$ and tends to $+\infty$ if any one particular constraint tends to have a value of zero. This property ensures that once the minimization of the ϕ_k function is started from a feasible point, the point always remains in the feasible region. The term $s_k Q_k(\mathbf{X}_d)$ can be considered as a penalty term with s_k playing the role of a weighing factor (penalty parameter). The function $Q_k(\mathbf{X}_d)$ is constructed so as to give a penalty whenever some of the variables in $\mathbf{X}_d$ take values other than integer values. Thus the function $Q_k(\mathbf{X}_d)$ has the property that

$$Q_k(\mathbf{X}_d) = \begin{cases} 0 & \text{if } \mathbf{X}_d \in S_d \\ \mu > 0 & \text{if } \mathbf{X}_d \notin S_d \end{cases} \tag{10.75}$$

We can take, for example,

$$Q_k(\mathbf{X}_d) = \sum_{x_i \in \mathbf{X}_d} \left\{ 4\left(\frac{x_i - y_i}{z_i - y_i}\right)\left(1 - \frac{x_i - y_i}{z_i - y_i}\right)\right\}^{\beta_k} \tag{10.76}$$

where $y_i \leq x_i$, $z_i \geq x_i$, and $\beta_k \geq 1$ is a constant. Here y_i and z_i are the two neighboring integer values for the value x_i. The function $Q_k(\mathbf{X}_d)$ is a normalized, symmetric beta function integrand. The variation of each of the terms under summation sign in Eq. (10.76) for different values of β_k is shown in Fig. 10.9. The value of β_k has to be greater than or equal to 1 if the function Q_k is to be continuous in its first derivative over the discretization or integer points.

The use of the penalty term defined by Eq. (10.76) makes it possible to change the shape of the ϕ_k function by changing β_k, while the amplitude can be controlled by the weighting factor s_k. The ϕ_k function given in Eq. (10.73) is now minimized for a sequence of values of r_k and s_k such that for $k \to \infty$, we obtain

$$\begin{gathered} \text{Min } \phi_k(\mathbf{X}, r_k, s_k) \to \text{Min } f(\mathbf{X}) \\ g_j(\mathbf{X}) \geq 0, \quad j = 1,2,\ldots,m \\ Q_k(\mathbf{X}_d) \to 0 \end{gathered} \tag{10.77}$$

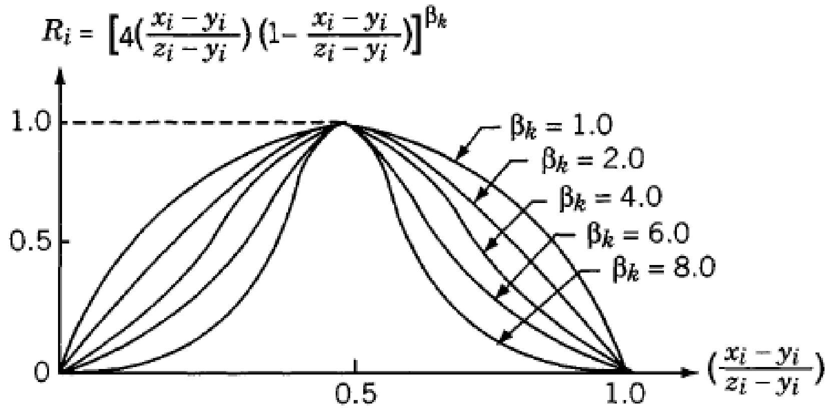


Figure 10.9 Contour of typical term in Eq. (10.62). [10.4] (Reprinted with permission from ASME.)

In most of the practical problems, one can obtain a reasonably good solution by carrying out the minimization of ϕ_k even for 5 to 10 values of k. The method is illustrated in Fig. 10.10 in the case of a single-variable problem. It can be noticed from Fig. 10.10 that the shape of the ϕ function (also called the response function) depends strongly on the numerical values of r_k, s_k, and β_k.

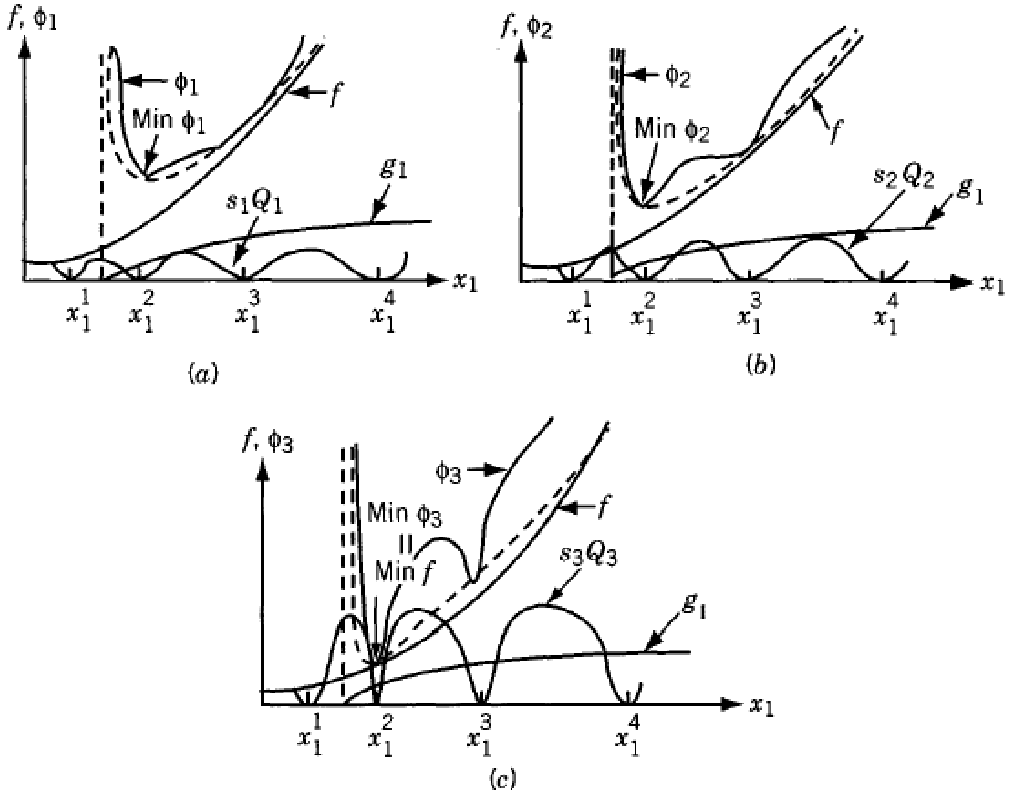


Figure 10.10 Solution of a single-variable integer problem by penalty function method. x_1, discrete variable; x_1^j, jth value of x_1. [10.4] (Reprinted with permission from ASME.)

Choice of the Initial Values of r_k, s_k, and β_k. The numerical values of r_k, s_k, and β_k have to be chosen carefully to achieve fast convergence. If these values are chosen such that they give the response surfaces of ϕ function as shown in Fig. 10.10*c*, several local minima will be introduced and the risk in finding the global minimum point will be more. Hence the initial value of s_k (namely, s_1) is to be chosen sufficiently small to yield a unimodal response surface. This can be achieved by setting

$$s_k Q_k' \ll P_k' \tag{10.78}$$

where Q_k' is an estimate of the maximum magnitude of the gradient to the Q_k surface and P_k' is a measure of the gradient of the function P_k defined by

$$P_k = f(\mathbf{X}) + r_k \sum_{j=1}^{m} G_j[g_j(\mathbf{X})] \tag{10.79}$$

Gisvold and Moe [10.4] have taken the values of Q_k' and P_k' as

$$Q_k' = \tfrac{1}{2} \cdot 4^{\beta_k} \beta_k (\beta_k - 1)^{\beta_k - 1} (2\beta_k - 1)^{1/2 - \beta_k} \tag{10.80}$$

$$P_k' = \left(\frac{\nabla P_k^T \nabla P_k}{n} \right)^{1/2} \tag{10.81}$$

where

$$\nabla P_k = \begin{Bmatrix} \partial P_k / \partial x_1 \\ \partial P_k / \partial x_2 \\ \vdots \\ \partial P_k / \partial x_n \end{Bmatrix} \tag{10.82}$$

The initial value of s_1, according to the requirement of Eq. (10.78), is given by

$$s_1 = c_1 \frac{P_1'(\mathbf{X}_1, r_1)}{Q_1'(\mathbf{X}_1^{(d)}, \beta_1)} \tag{10.83}$$

where $\mathbf{X}_1$ is the initial starting point for the minimization of ϕ_1, $\mathbf{X}_1^{(d)}$ the set of starting values of integer-restricted variables, and c_1 a constant whose value is generally taken in the range 0.001 and 0.1.

To choose the weighting factor r_1, the same consideration as discussed in Section 7.13 are to be taken into account. Accordingly, the value of r_1 is chosen as

$$r_1 = c_2 \frac{f(\mathbf{X}_1)}{+ \Sigma_{j=1}^{m} 1/g_j(\mathbf{X}_1)} \tag{10.84}$$

with the value of c_2 ranging between 0.1 and 1.0. Finally, the parameter β_k must be taken greater than 1 to maintain the continuity of the first derivative of the function ϕ_k over the discretization points. Although no systematic study has been conducted to find the effect of choosing different values for β_k, the value of $\beta_1 \simeq 2.2$ has been found to give satisfactory convergence in some of the design problems.

Once the initial values of r_k, s_k, and β_k (for $k = 1$) are chosen, the subsequent values also have to be chosen carefully based on the numerical results obtained on similar formulations. The sequence of values r_k are usually determined by using the relation

$$r_{k+1} = c_3 r_k, \qquad k = 1,2,\ldots \tag{10.85}$$

where $c_3 < 1$. Generally, the value of c_3 is taken in the range 0.05 to 0.5. To select the values of s_k, we first notice that the effect of the term $Q_k(\mathbf{X}_d)$ is somewhat similar to that of an equality constraint. Hence the method used in finding the weighting factors for equality constraints can be used to find the factor s_{k+1}. For equality constraints, we use

$$\frac{s_{k+1}}{s_k} = \frac{r_k^{1/2}}{r_{k+1}^{1/2}} \tag{10.86}$$

From Eqs. (10.85) and (10.86), we can take

$$s_{k+1} = c_4 s_k \tag{10.87}$$

with c_4 approximately lying in the range $\sqrt{1/0.5}$ and $\sqrt{1/0.05}$ (i.e., 1.4 and 4.5). The values of β_k can be selected according to the relation

$$\beta_{k+1} = c_5 \beta_k \tag{10.88}$$

with c_5 lying in the range 0.7 to 0.9.

A general convergence proof of the penalty function method, including the integer programming problems, was given by Fiacco [10.6]. Hence the present method is guaranteed to converge at least to a local minimum if the recovery procedure is applied the required number of times.

Example 10.6 [10.24] Find the minimum weight design of the three-bar truss shown in Fig. 10.11 with constraints on the stresses induced in the members. Treat the areas of cross section of the members as discrete variables with permissible values of the parameter $A_i\sigma_{\max}/P$ given by 0.1, 0.2, 0.3, 0.5, 0.8, 1.0, and 1.2.

SOLUTION By defining the nondimensional quantities f and x_i as

$$f = \frac{W\sigma_{\max}}{P\rho l}, \qquad x_i = \frac{A_i\sigma_{\max}}{P}, \qquad i = 1,2,3$$

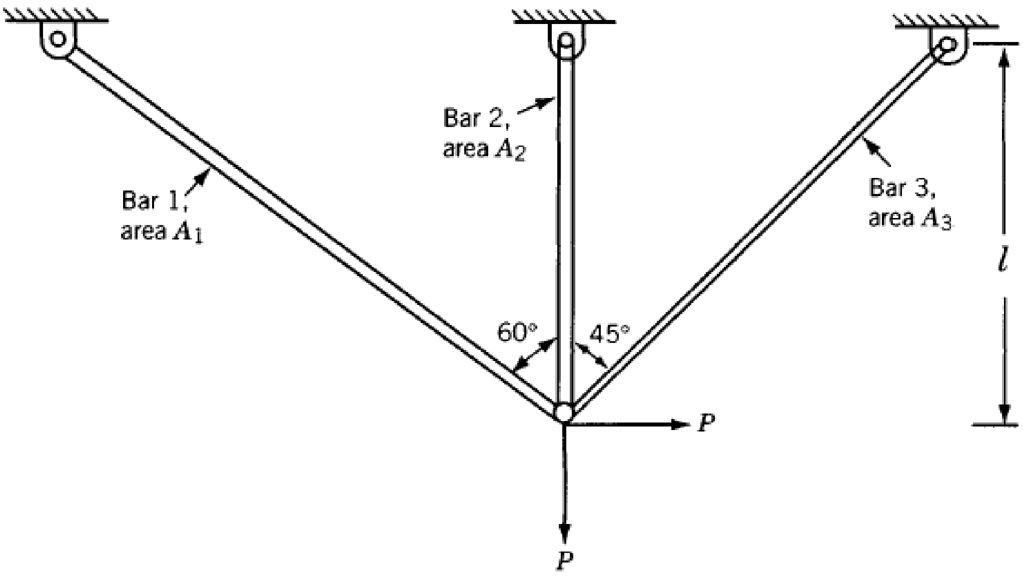


Figure 10.11 Three-bar truss.

where W is the weight of the truss, $\sigma_{\max}$ the permissible (absolute) value of stress, P the load, ρ the density, l the depth, and A_i the area of cross section of member i ($i = 1,2,3$), the discrete optimization problem can be stated as follows:

$$\text{Minimize } f = 2x_1 + x_2 + \sqrt{2}\, x_3$$

subject to

$$g_1(\mathbf{X}) = 1 - \frac{\sqrt{3}\, x_2 + 1.932x_3}{1.5x_1x_2 + \sqrt{2}\, x_2x_3 + 1.319x_1x_3} \geq 0$$

$$g_2(\mathbf{X}) = 1 - \frac{0.634x_1 + 2.828x_3}{1.5x_1x_2 + \sqrt{2}\, x_2x_3 + 1.319x_1x_3} \geq 0$$

$$g_3(\mathbf{X}) = 1 - \frac{0.5x_1 - 2x_2}{1.5x_1x_2 + \sqrt{2}\, x_2x_3 + 1.319x_1x_3} \geq 0$$

$$g_4(\mathbf{X}) = 1 + \frac{0.5x_1 - 2x_2}{1.5x_1x_2 + \sqrt{2}\, x_2x_3 + 1.319x_1x_3} \geq 0$$

$$x_i \in \{0.1,0.2,0.3,0.5,0.8,1.0,1.2\}, \qquad i = 1,2,3$$

The optimum solution of the continuous variable problem is given by $f^* = 2.7336$, $x_1^* = 1.1549$, $x_2^* = 0.4232$, and $x_3^* = 0.0004$. The optimum solution of the discrete variable problem is given by $f^* = 3.0414$, $x_1^* = 1.2$, $x_2^* = 0.5$, and $x_3^* = 0.1$.

REFERENCES AND BIBLIOGRAPHY

10.1 M. L. Balinski, Integer programming: methods, uses, computation, *Management Science*, Vol. 12, pp. 253–313, 1965.

10.2 L. J. Watters, Reduction of integer polynomial programming problems to zero–one linear programming problems, *Operations Research*, Vol. 15, No. 6, pp. 1171–1174, 1967.

10.3 S. Retter and D. B. Rice, Discrete optimizing solution procedures for linear and nonlinear integer programming problems, *Management Science*, Vol. 12, pp. 829–850, 1966.

10.4 K. M. Gisvold and J. Moe, A method for nonlinear mixed-integer programming and its application to design problems, *Journal of Engineering for Industry, Transactions of ASME*, Vol. 94, pp. 353–364, May 1972.

10.5 R. A. Gellatly and P. B. Marcal, *Investigation of Advanced Spacecraft Structural Design Technology*, NASA Report 2356-950001, 1967.

10.6 A. V. Fiacco, Penalty methods for mathematical programming in E^n with general constraints sets, *Journal of Optimization Theory and Applications*, Vol. 6, pp. 252–268, 1970.

10.7 R. E. Gomory, *An Algorithm for the Mixed Integer Problem*, Rand Report R.M. 25797, July 1960.

10.8 A. H. Land and A. Doig, An automatic method of solving discrete programming problems, *Econometrica*, Vol. 28, pp. 497–520, 1960.

10.9 E. Balas, An additive algorithm for solving linear programs with zero–one variables, *Operations Research*, Vol. 13, pp. 517–546, 1965.

10.10 R. E. Gomory, An all-integer integer programming algorithm, Chapter 13 in *Industrial Scheduling*, J. F. Muth and G. L. Thompson, Eds., Prentice-Hall, Englewood Cliffs, N.J., 1963.

10.11 H. A. Taha, *Operations Research: An Introduction*, Macmillan, New York, 1971.

10.12 S. Zionts, *Linear and Integer Programming*, Prentice-Hall, Englewood Cliffs, N.J., 1974.

10.13 C. McMillan, Jr., *Mathematical Programming: An Introduction to the Design and Application of Optimal Decision Machines*, Wiley, New York, 1970.

10.14 N. K. Kwak, *Mathematical Programming with Business Applications*, McGraw-Hill, New York, 1973.

10.15 H. Greenberg, *Integer Programming*, Academic Press, New York, 1971.

10.16 G. B. Dantzig and A. F. Veinott, Jr., Eds., *Mathematics of the Decision Sciences*, Part 1, American Mathematical Society, Providence, R.I., 1968.

10.17 R. S. Garfinkel and G. L. Nemhauser, *Integer Programming*, Wiley, New York, 1972.

10.18 C. A. Trauth, Jr., and R. E. Woolsey, Integer linear programming: a study in computational efficiency, *Management Science*, Vol. 15, No. 9, pp. 481–493, 1969.

10.19 E. L. Lawler and M. D. Bell, A method for solving discrete optimization problems, *Operations Research*, Vol. 14, pp. 1098–1112, 1966.

10.20 P. Hansen, Quadratic zero–one programming by implicit enumeration, pp. 265–278 in *Numerical Methods for Nonlinear Optimization*, F. A. Lootsma, Ed., Academic Press, London, 1972.

10.21 R. R. Meyer, Integer and mixed-integer programming models: general properties, *Journal of Optimization Theory and Applications*, Vol. 16, pp. 191–206, 1975.

10.22 R. Karni, Integer linear programming formulation of the material requirements planning problem, *Journal of Optimization Theory and Applications*, Vol. 35, pp. 217–230, 1981.

10.23 R. J. Dakin, A tree-search algorithm for mixed integer programming problems, *Computer Journal*, Vol. 8, No. 3, pp. 250–255, 1965.

10.24 D. K. Shin, Z. Gürdal, and O. H. Griffin, Jr., A penalty approach for nonlinear optimization with discrete design variables, *Engineering Optimization*, Vol. 16, pp. 29–42, 1990.

10.25 O. K. Gupta and A. Ravindran, Nonlinear mixed integer programming and discrete optimization, pp. 27–32 in *Progress in Engineering Optimization—1981*, R. W. Mayne and K. M. Ragsdell, Eds., ASME, New York, 1981.

10.26 G. R. Olsen and G. N. Vanderplaats, Method for nonlinear optimization with discrete variables, *AIAA Journal*, Vol. 27, No. 11, pp. 1584–1589, 1989.

10.27 K. V. John, C. V. Ramakrishnan, and K. G. Sharma, Optimum design of trusses from available sections: use of sequential linear programming with branch and bound algorithm, *Engineering Optimization*, Vol. 13, pp. 119–145, 1988.

10.28 M. W. Cooper, A survey of methods for pure nonlinear integer programming, *Management Science*, Vol. 27, No. 3, pp. 353–361, 1981.

10.29 A. Glankwahmdee, J. S. Liebman, and G. L. Hogg, Unconstrained discrete nonlinear programming, *Engineering Optimization*, Vol. 4, pp. 95–107, 1979.

10.30 K. Hager and R. Balling, New approach for discrete structural optimization, *ASCE Journal of the Structural Division*, Vol. 114, No. ST5, pp. 1120–1134, 1988.

10.31 S. S. Rao, T. S. Pan, and V. B. Venkayya, Optimal placement of actuators in actively controlled structures using genetic algorithms, *AIAA Journal*, Vol. 29, No. 6, pp. 942–943, 1991.

10.32 R. G. Parker and R. L. Rardin, *Discrete Optimization*, Academic Press, Boston, 1988.