

PART III

SOLUTIONS

11

CLASSICAL SOLUTION APPROACHES

This chapter introduces three classical approaches for solving integer programs, namely, *branch-and-bound*, *cutting plane*, and *group theoretic*. Although all approaches are capable of solving integer programs, their degrees of success vary in software implementation. The cutting plane approach, when used as a stand-alone solver, has potential to solve IP programs of limited size, but may not work well in large-scale application. Similarly limited is the group theoretic approach, which has not been implemented as a stand-alone solver in practice. However, the valid inequality cuts generated by both cutting plane and group theoretic approaches can be useful when combined with branch-and-bound to yield a powerful branch-and-cut approach.

For over three decades, the branch-and-bound had been the prevailing solution method until the emergence of the *branch-and-cut* in early 1990s. Branch-and-cut combined branch-and-bound with the generated cutting planes into a much more efficient “hybrid” approach. Similarly, the group cuts generated from the group theoretical approach have also been incorporated, but at a lesser degree of integration. As a whole, extracting the strengths of these two approaches and injecting them into the branch-and-bound may greatly increase the modern solution power for integer programs. In what follows, we will introduce the concepts and background of these three solution approaches, and then exploit the potential strengths of each approach.

11.1 BRANCH-AND-BOUND APPROACH

Branch-and-bound is a general-purpose approach capable of solving pure IP, mixed IP, and binary IP problems. For ease of exposition, sometimes we shall assume that the given problem is a pure IP problem because a similar algorithm can be applied to a mixed or binary IP problem. We also assume that the given problem is a maximization problem because modification of the algorithm for the minimization problem is straightforward.

11.1.1 Basic Concepts

Theoretically, any pure IP problem with *finite* bounds on integer variables can be solved by enumerating *all* possible combinations of integer values and determining a combination (solution) that satisfies all constraints and yields the maximal objective value—hence the name of *complete enumeration*. Unfortunately, the number of all possible combinations is prohibitively large to be evaluated even for a small problem. A problem of n integer variables with m values each has a total of m^n possible combinations (feasible and infeasible solutions). Therefore, complete enumeration is theoretically simple but practically intractable.

As a better alternative, *implicit enumeration* applies an intelligent enumeration scheme that can cover all possible solutions by explicitly evaluating only a small number of solutions while ignoring (or implicitly enumerating) a large number of *inferior* solutions. One such strategy is called *divide and conquer*. Basically, this strategy *divides* the given problem into a series of easier to solve subproblems that are systematically generated and solved (or *conquered*). The solutions of these generated subproblems are then put together to solve the original problem.

Branch-and-bound can be viewed as a divide-and-conquer approach to solving the IP problem, in which a branching process for dividing and a bounding process for conquering. Throughout the algorithm, a series of LP subproblems are systematically generated and solved. Then upper and lower bounds are progressively tightened on the objective value of the original IP problem.

A typical way to represent such a process is via a *branch-and-bound (B&B) tree*, which is a specialized enumeration tree for keeping track of how LP subproblems are generated and solved. The B&B tree by convention is drawn upside down with its root node at the top. The root node that represents the LP relaxation of the original IP problem (denoted by S_{LP}) is solved. If the LP optimum solution satisfies the integer requirement, the IP problem is solved. Otherwise, the LP objective value becomes the initial upper bound on the IP optimal objective value and the root node is partitioned into two successor nodes (subproblems) by two branches. These branches are valid cuts in terms of simple inequality constraints that have the following properties: (a) they cut off the current noninteger LP optimum point and other fractional region, and (b) the two successor nodes are mutually exclusive and their union contains the same integer feasible region as that of their predecessor (i.e., no integer points are eliminated). The solution of an LP relaxation on a node provides information about (a) whether a further branching from this node is needed (or whether the node can be *pruned*), and (b) a better lower bound (for maximization problem) on the objective of the original IP problem.

Note that in some texts, the term *pruned* may be replaced by *fathomed*, to indicate that no further exploration beyond that point is necessary.

There are three cases indicating that a node can be pruned: (1) the subproblem has no feasible LP solution, (2) the subproblem has an integer optimum solution, and (3) the upper bound of the subproblem optimum is less than or equal to the lower bound of the original problem. These three cases are, respectively, referred to as *pruned by infeasibility*, *pruned by optimality*, and *pruned by bound*. If a node is pruned by optimality, its optimum solution can be used to increase the lower bound on the objective value of the original IP problem.

Whenever an integer solution to a subproblem is obtained, it is a *candidate* optimum to the original IP problem. In the solution process of B&B, the best integer solution found so far is continuously updated. Such a solution is called an *incumbent solution*. To illustrate how the B&B algorithm works, we use the following two examples—one for pure and one for mixed IP problem.

Example 11.1 Solve the following pure IP problem by branch-and-bound approach.

$$\begin{aligned} \text{Maximize} \quad & z = 5y_1 - 2y_2 \\ \text{subject to} \quad & -y_1 + 2y_2 \leq 5 \\ & 3y_1 + 2y_2 \leq 19 \\ & y_1 + 3y_2 \geq 9 \\ & y_1, y_2 \geq 0 \text{ and integer} \end{aligned}$$

We first solve the LP relaxation S_{LP} . As shown in Figure 11.1, the shaded area represents the LP feasible region and the solid lattice points the IP feasible solutions.

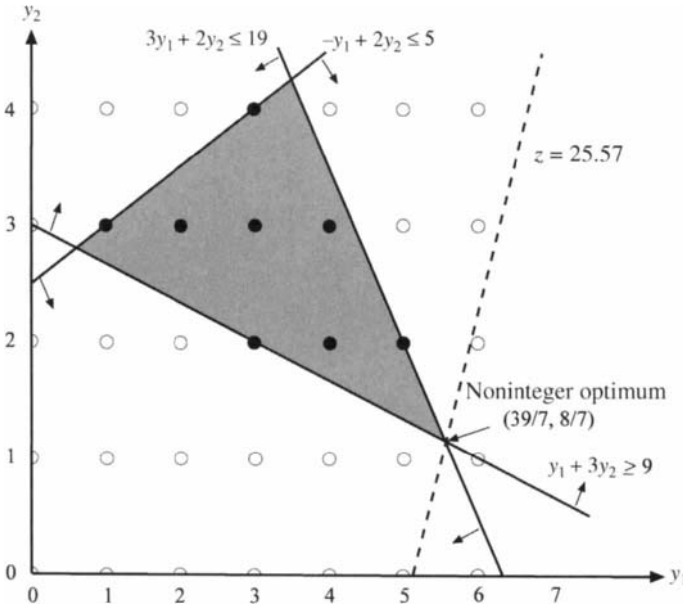


FIGURE 11.1 LP and IP feasible regions for Example 11.1.

We obtain the noninteger optimum $y_1 = 39/7$, $y_2 = 8/7$, and $z = 25.57$. Then the objective value 25.57 becomes an upper bound to the IP problem. We set the lower bound to $-\infty$. Since both variables are fractional, we need to branch on them in an attempt to obtain an integer optimum.

We arbitrarily select y_1 as the variable to be branched. Two subproblems are generated by adding the constraint of $y_1 \geq 6$ and $y_1 \leq 5$, respectively, to the LP relaxation. From Figure 11.2 we can see that the triangle area S' is cut off by $y_1 \leq 5$. Clearly, the branch with the added constraint $y_1 \geq 6$ is infeasible, so it is *pruned by infeasibility*. The other branch with the added constraint $y_1 \leq 5$ is optimized at $(y_1, y_2) = (5, 4/3)$, with objective value 22.33. So the new *upper bound* is updated to 22.33.

Again, the variable y_2 is fractional, so this time we branch on y_2 . The two constraints $y_2 \geq 2$ and $y_2 \leq 1$ are then added. This time the area S'' is cut off, as shown in Figure 11.3.

The branch with $y_2 \leq 1$ is infeasible, and hence is pruned by infeasibility. The branch with $y_2 \geq 2$ is optimized at $(y_1, y_2) = (5, 2)$, with objective value 21. Since this is a feasible solution to the IP problem, the value 21 becomes a new *lower bound* to the problem, replacing the initial lower bound $-\infty$, and $(5, 2)$ is a *candidate solution*. Checking the tree, all branches are evaluated, so $(y_1, y_2) = (5, 2)$ is the optimal solution to the IP problem, and the optimal objective value is 21.

The branch-and-bound algorithm is usually depicted as an enumeration tree, in which the nodes denote the subproblems, and the branches correspond to constraints

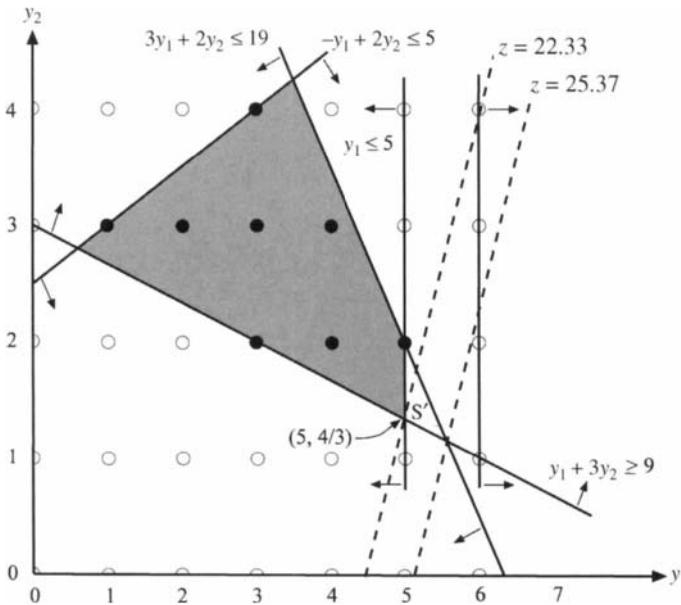


FIGURE 11.2 LP and IP feasible regions after the first branching.

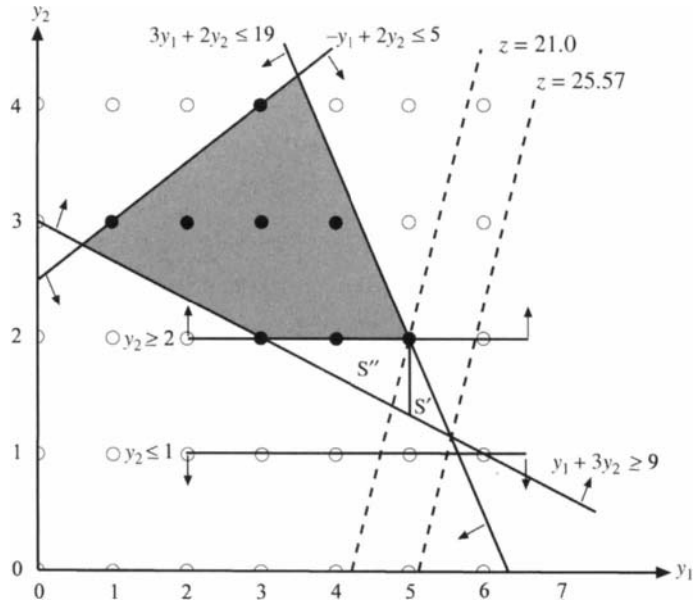


FIGURE 11.3 LP and IP solution regions after the second branching.

(cuts) that separate the subproblems from their parent subproblems. The number above each node is the optimal solution to the LP subproblem generated on that node (which is also the upper bound on that branch). The number below the node indicates the best lower bound on the original IP problem found so far. The previous procedure is depicted in Figure 11.4.

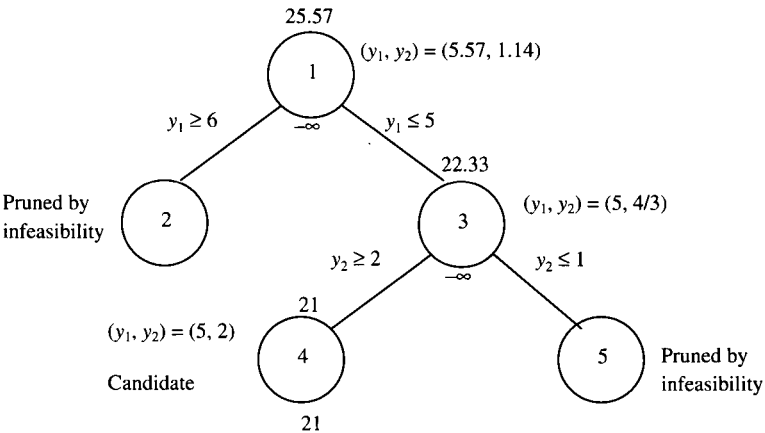


FIGURE 11.4 Branch-and-bound tree for Example 11.1.

When an LP solution contains several fractional integer variables, the decision of which integer variable to branch on next is needed. The following rules are commonly used for choosing a branching variable:

1. Variable with fractional value closest to 0.5
2. Variable with highest impact on objective function
3. Variable with the least index

A decision is also needed as to which unpruned node to explore first. The most commonly used search strategies include

1. Depth-first (last-in first-out; solve the most recently generated subproblem first)
2. Best-bound-first (best upper bound; branch on the active node with greatest z -value)

The goal of the depth-first strategy is to quickly obtain a primal feasible integer solution whose objective function value z^k is a lower bound on the given IP problem and can be used to prune nodes by optimality (rule 3). The best-bound-first strategy chooses the active node with the best upper bound (for maximization problem). The goal is to minimize the total number of nodes evaluated in the B&B tree. Performance of these branching rules depends on the problem structure. In practice, a compromise between the two is adopted. That is, apply the depth-first strategy first to get one feasible integer solution, followed by a mixture of both strategies.

Example 11.2 Solve the following mixed integer problem using branch-and-bound approach. At each step, apply the rule of best-bound-first, and at each node, select the variable with least index to branch first.

$$\begin{array}{ll}
 \text{Maximize} & z = -y_1 + 2y_2 + y_3 + 2x_1 \\
 \text{subject to} & y_1 + y_2 - y_3 + 3x_1 \leq 7 \\
 & y_2 + 3y_3 - x_1 \leq 5 \\
 & 3y_1 + x_1 \geq 2 \\
 & y_1, y_2, y_3 \geq 0 \text{ and integer} \\
 & x_1 \geq 0
 \end{array}$$

After solving the LP relaxation, we obtain an LP optimum $y_1 = 6/11$, $y_2 = 59/11$, $y_3 = 0$, $x_1 = 4/11$, and $z = 120/11$. This solution violates the integer requirements of y_1 and y_2 . We use this solution as the root node the branch-and-bound tree in Figure 11.5. The number of each node indicates the sequence of subproblems evaluated. Note

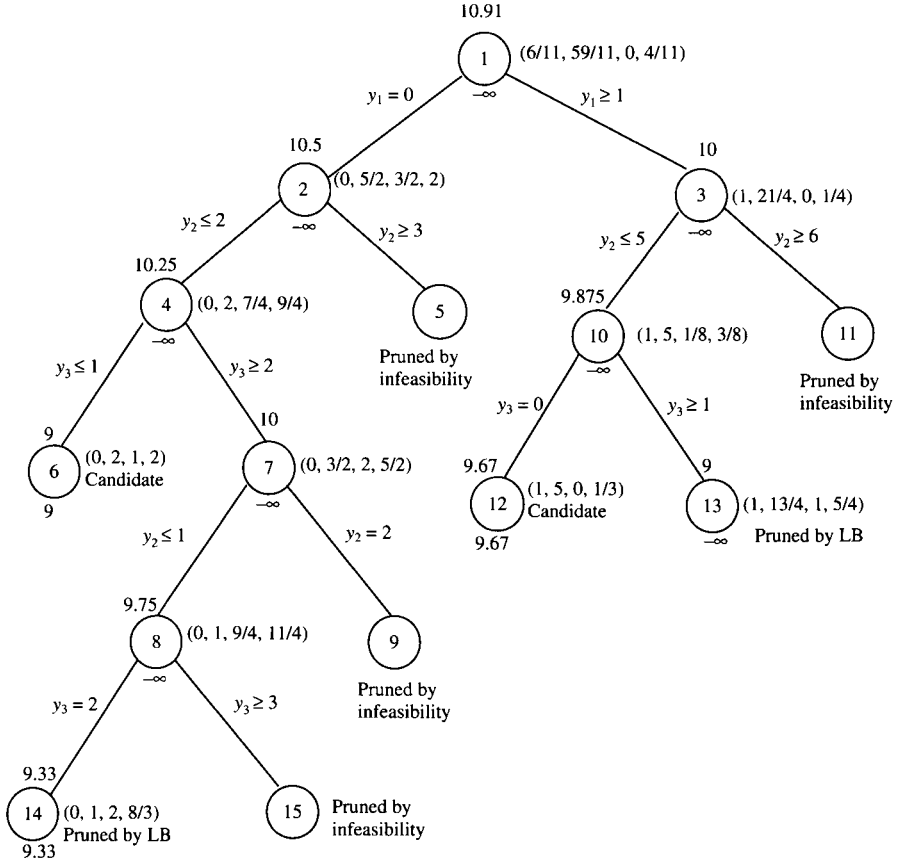


FIGURE 11.5 Branch-and-bound tree for Example 11.2 using best-bound-first.

that at node 1, the constraint $y_1 \leq 0$ was indicated on the left branch, but since $y_1 \geq 0$, y_1 has to be fixed at 0. At node 7, the constraint $y_2 \geq 2$ was intended to be added, but if we trace back along node 7, we would see that the constraint of $y_2 \leq 2$ was already added at node 2. Combining these two constraints, we have $y_2 = 2$. So is the constraint of $y_3 = 2$ at node 8. The problem is finally optimized at node 12, where $(1, 5, 0, 1/3)$ is the optimal solution, with objective value 9.67.

Figure 11.6 depicts the branch-and-bound tree for the same problem, where the “depth-first” rule is applied, and at each node, the variable (violating an integer constraint) with the largest absolute value cost coefficient is chosen to branch first. Ties are broken arbitrarily.

Depth-first is sometimes called last-in first-out (LIFO) because it solves the most recently generated subproblem first. It tends to pursue paths to the depths of the tree, then backtrack to where that path started, and finally plunge down into another

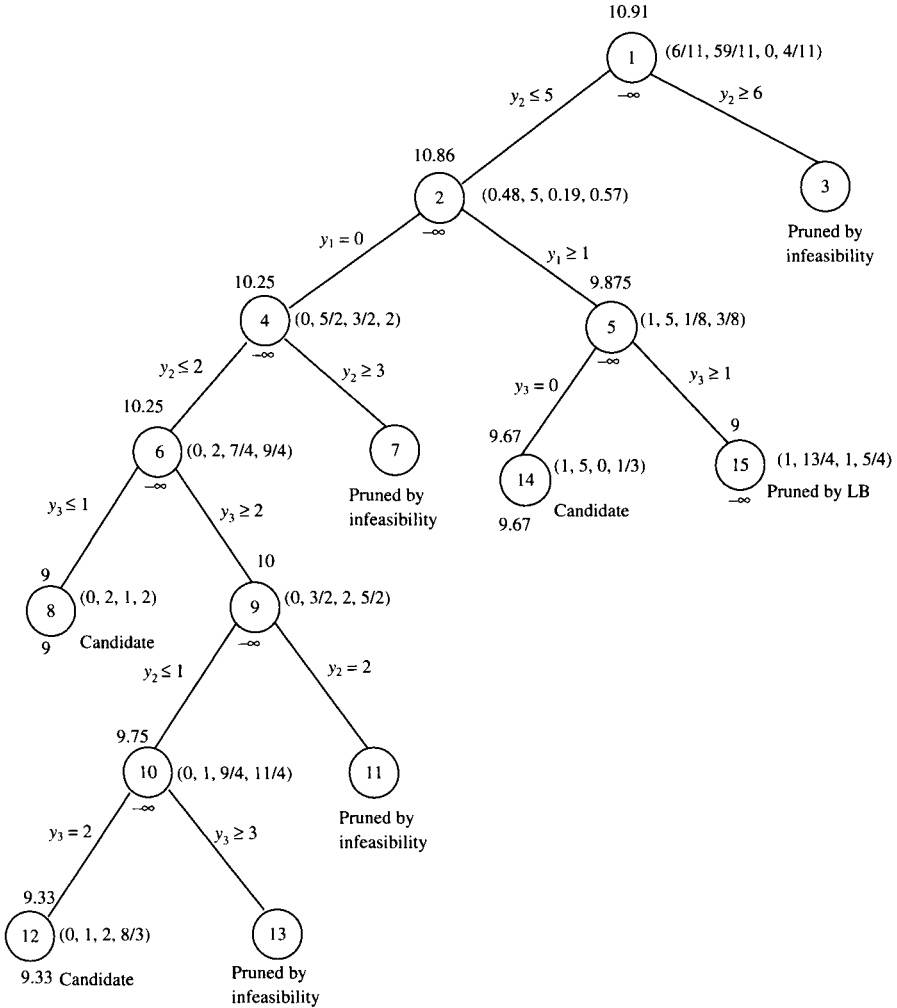


FIGURE 11.6 Branch-and-bound tree for Example 11.2 using depth-first.

depth search. Yet another name for depth-first is “backtracking.” Best-bound-first is sometimes called “jumptracking” because it leads to searches that jump back and forth across the tree.

11.1.2 Branch-and-Bound Algorithm

Now we describe the general branch-and-bound algorithm using the following notation.

- S = the given IP problem
- S_{LP} = the LP relaxation of S

y_{LP} = the solution to the LP relaxation of the given IP

$\bar{z}$ = lowest (best) upper bound on z^* of the given IP problem

$\underline{z}$ = highest (best) lower bound on z^* of the given IP problem

These are *global* bounds that are periodically updated as the branching proceeds down the various paths in the tree, but are *not* shown on the tree. In Example 11.2, $\bar{z} = 10.91$ throughout and $\underline{z}$ was $-\infty$, then 9, 9.33, and 9.67 at nodes 1, 8, 12, and 14, respectively. Next comes more notation. Let

S^k = subproblem k of problem S

S_{LP}^k = the LP relaxation of subproblem k

z^k = the optimum objective value of S^k

$\bar{z}^k$ = best (lowest) upper bound of subproblem S^k (shown above node k)

$\underline{z}^k$ = best (highest) lower bound of subproblem S^k (shown below node k)

y_{LP}^k = the optimum solution of the LP subproblem S_{LP}^k

$\bar{y}_j$ = noninteger value of integer variable y_j (current numerical value of y_j)

$\lfloor a \rfloor$ = the largest integer $\leq a$ (or rounding down a)

$\lceil a \rceil$ = the smallest integer $\geq a$ (or rounding up a)

We now formally describe the B&B algorithm

Step 0 (Initialization). Solve the LP relaxation (S_{LP}) of the given IP problem (S). If it is infeasible, so is the IP problem—terminate. If the LP optimum solution satisfies the integer requirement, the IP problem is solved—terminate. Otherwise, initialize the best upper bound ($\bar{z}$) by the optimal objective value of problem S_{LP} and the best lower bound by $\underline{z} = -\infty$. Place S_{LP}^k on the active list of nodes (subproblems). Initially, there is no incumbent solution.

Step 1 (Choosing a Node). If the active list is empty, terminate. The incumbent solution y^* is optimal. Otherwise, choose a node (subproblem) S^k with S_{LP}^k by one of the rules (e.g., depth-first, best-bound-first, etc.)

Step 2 (Updating Upper Bound). Solve and set $\bar{z}^k$ equal to the LP optimum objective value. Keep the optimum LP solution y_{LP}^k .

Step 3 (Prune by Infeasibility). If S_{LP}^k has no feasible solution, prune the current node and go to step 1. Otherwise, go to step 4.

Step 4 (Prune by Bound). If $z^k \leq \underline{z}$, prune the current node and go to step 1. Otherwise, go to step 5.

Step 5 (Updating Lower Bound and Prune by Optimality).

(a) If the LP optimum y_{LP}^k is integer, a feasible solution to S is found, an incumbent solution to the given problem. Set $\underline{z}^k = y_{LP}^k$ and compare $\underline{z}^k$ with $\underline{z}$. If $\underline{z}^k > \underline{z}$, set $\underline{z} = \underline{z}^k$, otherwise $\underline{z}$ does not change. The current node is pruned because no better solution can be branched down from this node. Go to step 1.

(b) If the LP optimum y_{LP}^k is noninteger, go to step 6.

Step 6 (Branching). From the current node S^k choose a variable y_j with fractional value to generate two subproblems S_1^k and S_2^k defined by

$$S_1^k = S^k \cap \{y : y_j \leq \lfloor \bar{y}_j \rfloor\}$$

$$S_2^k = S^k \cap \{y : y_j \geq \lceil \bar{y}_j \rceil\}$$

Place both of these two nodes in the active list and go to step 1.

11.2 CUTTING PLANE APPROACH

In geometry, an equation in two variables is called a *plane* and an equation in n variables a *hyperplane*, strictly speaking. For simplicity, however, both in practice are often referred to as a plane, regardless of the number of variables. Strictly speaking, an inequality constraint in n variables is called a *half-space*, not a hyperplane. But an inequality constraint can always be converted to an equation by adding or subtracting a nonnegative slack variable. The term *cutting plane* is often used for an equality or inequality constraint that can cut off a fractional part of an LP feasible region, without excluding any *integer* feasible solution. In the cutting plane approach, one or more such cutting planes are added to the current LP simplex tableau, which in turn are resolved for a new LP optimum. This process is repeated until the prescribed integer requirements are satisfied. In this text, the collection of all such cutting plane methods will be called a *cutting plane approach* (more specifically, a *dual cutting plane approach*, due to the use of the dual simplex method for LP reoptimization).

11.2.1 Dual Cutting Plane Approach

A large variety of cutting plane methods were developed during the 1950s and 1960s. Among them, the most prominent ones belong to the class of *dual* cutting plane approach such as the fractional and mixed cutting plane methods developed by Gomory. This class shares a common solution algorithm when they are utilized as a *stand-alone* solver.

- Step 1.* Solve the integer program as if it were a linear program. If it is infeasible, so is the integer program and then stop. Else if an LP optimal solution satisfying the integer requirements is found, then the IP is solved. Otherwise, go to step 2.
- Step 2.* Select a row to be a *generating row* (or *source row*) from the LP optimum simplex tableau.
- Step 3.* Derive a cut constraint from the generating row and augment it to the current tableau, resulting in a primal infeasible solution.
- Step 4.* Apply the dual simplex method to reoptimize the augmented linear program. If the new LP optimum satisfies the integer requirements, the original MIP program is solved. Otherwise, go to step 2.

The main difference among various methods of cutting plane is *how* a cut constraint is generated. The main requirement is that a generated cut constraint must be *valid*, meaning that its addition will result in cutting off the current LP optimal point but will not eliminate any *integer feasible* solution. In other words, every valid cut has two properties:

- The current optimal solution to the LP relaxation problem will violate the cut constraints.
- Any feasible point to the corresponding IP or MIP problems will satisfy the cut constraint.

The class of dual cutting plane methods begins with an optimal LP solution and requires application of a series of dual simplex steps to reoptimize a series of new LP problems, each adding one or more constraints to the current simplex tableau (although some cuts may be dropped from later considerations).

There is another class of cutting plane methods known as the *primal cutting plane approach*. This approach commonly begins with a primal simplex tableau and creates a series of primal simplex tableaux, from which cuts are generated. As a result, all the subsequent simplex tableaux will remain primal feasibility and dual infeasibility. The primal simplex method is applied throughout the process until both primal and dual solutions are feasible, in which case an optimum solution is found. We shall not describe them in detail. The interested reader may refer the Section 11.5.

11.2.2 Fractional Cutting Plane Method

The fractional cutting plane method is capable of solving *pure* integer programs. This method requires that the starting IP problem must contain *all-integer coefficients* so that all slack variables, including those that are added after introduction of cuts, are ensured to be nonnegative integers. Note that the integer assumption of the starting IP problem does not limit the problem application because any coefficients consisting of *rational numbers* can always be made integers by multiplying an appropriate number.

The fractional cutting plane method begins with an *optimal* simplex tableau of the LP relaxation given below (recall that we use y_j to denote integer variables):

$$\begin{aligned}
 &\text{Maximize} && z + \sum_k \bar{d}_k y_k = \bar{d}_0 \\
 &\text{subject to} && y_{B_i} + \sum_k \bar{g}_{ik} y_k = \bar{b}_i \quad i = 1, 2, \dots, m \\
 & && y_{B_i} \geq 0 \quad i = 1, 2, \dots, m \\
 & && y_k = 0 \quad k = 1, 2, \dots, p
 \end{aligned} \tag{11.1}$$

where y_{B_i} and y_k denote basic variables and nonbasic variables, respectively. Note that the current LP optimum solution is $y_{B_i} = \bar{b}_i$ and $y_k = 0$, in which some $\bar{b}_i$ are assumed to be noninteger. Optimality conditions require that $\bar{d}_k \geq 0$ for all k .

To find an integer solution, we arbitrarily select a row with $\bar{b}_i$ noninteger. The selected row, say r , is called a *source row* or *generating row*, from which a fractional cut will be generated. Consider the source row

$$y_{B_r} + \sum_k \bar{g}_{rk} y_k = \bar{b}_r \quad (k = 1, \dots, p)$$

which can be rewritten by separating fractional and integral parts of all data:

$$y_{B_r} + \sum_k \{(\bar{g}_{rk} - \lfloor \bar{g}_{rk} \rfloor) + \lfloor \bar{g}_{rk} \rfloor\} y_k = (\bar{b}_r - \lfloor \bar{b}_r \rfloor) + \lfloor \bar{b}_r \rfloor$$

where $\lfloor a \rfloor$ denotes the largest integer $\leq a$. For example, $\lfloor 5.4 \rfloor = 5$, $\lfloor -1.8 \rfloor = -2$, and $\lfloor 3 \rfloor = 3$. The fractional part is always ≥ 0 . For simplicity, we let

$$\begin{aligned} f_{rk} &= \bar{g}_{rk} - \lfloor \bar{g}_{rk} \rfloor \\ f_{r0} &= \bar{b}_r - \lfloor \bar{b}_r \rfloor \end{aligned}$$

be the fractional parts of tableau coefficients and the RHS of row r . Rearranging the terms, we have

$$y_{B_r} + \sum_k \lfloor \bar{g}_{rk} \rfloor y_k - \lfloor \bar{b}_r \rfloor = f_{r0} - \sum_k f_{rk} y_k \quad (11.2)$$

Now in order for y_{B_r} and y_k ($k = 1, \dots, p$) to be integer, both the left-hand and right-hand sides of (11.2) must be integer. By the definition of congruence, we have

$$f_{r0} - \sum_k f_{rk} y_k = 0 \pmod{1} \quad (11.3)$$

But since $f_{r0} - \sum_k f_{rk} y_k \leq f_{r0} < 1$, the necessary condition for integrality becomes

$$f_{r0} - \sum_k f_{rk} y_k \leq 0$$

$$\text{or } \sum_k f_{rk} y_k = f_{r0} \quad (\text{Gomory fractional cut}) \quad (11.4)$$

$$\text{or } \sum_k -f_{rk} y_k + s = -f_{r0} \quad (\text{Gomory fractional cut}) \quad (11.5)$$

where $s \geq 0$ is Gomory's slack variable associated with the fractional cut. Note that all f_{rk} and f_{r0} must be nonnegative fractions, that is, $0 \leq f_{rk}$ ($k = 1, \dots, p$) and $f_{r0} \leq 1$.

Example 11.3 Solve the pure IP problem in (11.1) using the cutting plane method.

TABLE 11.1 Tableau 1 for Example 11.3

Basic Variable	z	y_1	y_2	y_3	y_4	y_5	RHS
z	1	0	0	0	$17/7$	$16/7$	$179/7$
y_1	0	1	0	0	$3/7$	$2/7$	$39/7$
y_2	0	0	1	0	$-1/7$	$-3/7$	$8/7$
y_3	0	0	0	1	$5/7$	$8/7$	$58/7$

We first add a nonnegative slack variable to each inequality constraint:

Maximize $z = 5y_1 - 2y_2$
subject to $-y_1 + 2y_2 + y_3 = 5$
 $3y_1 + 2y_2 + y_4 = 19$
 $-y_1 - 3y_2 + y_5 = -9$
 $y_1, y_2, y_3, y_4, y_5 \geq 0$ and integer

We then solve the LP relaxation, yielding the optimal simplex tableau shown in Table 11.1.

The optimum solution is noninteger: $y_1 = 39/7$, $y_2 = 8/7$, $y_3 = 58/7$, $y_4 = y_5 = 0$, and $z = 179/7$. We arbitrarily select y_1 row as the source row and generate the following fractional cut.

$$\frac{3}{7}y_4 + \frac{2}{7}y_5 \geq \frac{4}{7}$$

or

$$-\frac{3}{7}y_4 - \frac{2}{7}y_5 + s_1 = -\frac{4}{7}$$

where s_1 is called *Gomory's slack variable* to differentiate from the ordinary slack variable. Appending the equation to tableau 1, we obtain tableau 2 (Table 11.2).

Applying the dual simplex iteration, s_1 is replaced by y_4 yielding tableau 3 (Table 11.3).

Repeating generation of fractional cuts and application of dual simplex iterations, the reader may verify the subsequent simplex tableaux 4 through 6 (Tables 11.4–11.6).

TABLE 11.2 Tableau 2 for Example 11.3

Basic Variable	z	y_1	y_2	y_3	y_4	y_5	s_1	RHS
z	1	0	0	0	$17/7$	$16/7$	0	$179/7$
y_1	0	1	0	0	$3/7$	$2/7$	0	$39/7$
y_2	0	0	1	0	$-1/7$	$-3/7$	0	$8/7$
y_3	0	0	0	1	$5/7$	$8/7$	0	$58/7$
s_1	0	0	0	0	$-3/7$	$-2/7$	1	$-4/7$

TABLE 11.3 Tableau 3 for Example 11.3

Basic Variable	z	y_1	y_2	y_3	y_4	y_5	s_1	RHS
z	1	0	0	0	0	$2/3$	$17/3$	$67/3$
y_1	0	1	0	0	0	0	1	5
y_2	0	0	1	0	0	$-1/3$	$-1/3$	$4/3$
y_3	0	0	0	1	0	$2/3$	$5/3$	$22/3$
y_4	0	0	0	0	1	$2/3$	$-7/3$	$4/3$

TABLE 11.4 Tableau 4 for Example 11.3

Basic Variable	z	y_1	y_2	y_3	y_4	y_5	s_1	s_2	RHS
z	1	0	0	0	0	$2/3$	$17/3$	0	$67/3$
y_1	0	1	0	0	0	0	1	0	5
y_2	0	0	1	0	0	$-1/3$	$-1/3$	0	$4/3$
y_3	0	0	0	1	0	$2/3$	$5/3$	0	$22/3$
y_4	0	0	0	0	1	$2/3$	$-7/3$	0	$4/3$
s_2	0	0	0	0	0	$-2/3$	$-2/3$	1	$-1/3$

TABLE 11.5 Tableau 5 for Example 11.3

Basic Variable	z	y_1	y_2	y_3	y_4	y_5	s_1	s_2	RHS
z	1	0	0	0	0	0	5	1	22
y_1	0	1	0	0	0	0	1	0	5
y_2	0	0	1	0	0	0	0	$-1/2$	$3/2$
y_3	0	0	0	1	0	0	1	1	3
y_4	0	0	0	0	1	0	-3	1	1
y_5	0	0	0	0	0	1	1	$-3/2$	$1/2$
s_3	0	0	0	0	0	0	0	$-1/2$	$-1/2$

TABLE 11.6 Tableau 6 for Example 11.3

Basic Variable	z	y_1	y_2	y_3	y_4	y_5	s_1	s_2	s_3	RHS
z	1	0	0	0	0	0	5	0	2	21
y_1	0	1	0	0	0	0	1	0	0	5
y_2	0	0	1	0	0	0	0	0	1	2
y_3	0	0	0	1	0	0	1	0	2	2
y_4	0	0	0	0	1	0	-3	0	2	0
y_5	0	0	0	0	0	1	1	0	-3	2
s_2	0	0	0	0	0	0	0	1	-2	1

Because all basic variables are integer, we have an integer optimum $y_1 = 5$, $y_2 = 2$, and $z = 21$. The solution is the same as that obtained by branch-and-bound approach.

The cutting plane approach often takes a large number of cuts to reach an integer solution even for a small or moderate sized IP problem, although it can be shown that the fractional cutting plane method is ensured to converge to an IP optimum after a finite number of cuts. Here, we arbitrarily select a source row, although alternative rules may be applied to select other source rows. For example, we may select a source row r with f_{r0} closest to 0.5, or select a row with the largest f_{r0} . However, no evidence shows that a certain selection rule is better than the others in all cases.

11.2.3 Mixed Integer Cutting Plane Method

The mixed integer cutting plane method, also due to Gomory, can be used to solve the following MIP problem:

$$\begin{aligned} \text{Maximize} \quad & z = \sum_j c_j x_j + \sum_k d_k y_k \\ \text{subject to} \quad & \sum_j a_{ij} x_j + \sum_k g_{ik} y_k \leq b_i \quad (i = 1, 2, \dots, m) \\ & x_j \geq 0 \quad (j = 1, 2, \dots, n) \\ & y_k \geq 0 \text{ and integer} \quad (k = 1, 2, \dots, p) \end{aligned}$$

Essentially, the solution procedure is similar to that of the fractional cutting plane method. It generates a valid cut from the optimal simplex tableau of the LP relaxation of the MIP problem. Any row r associated with y_k , which is basic but has fractional right-hand side, may be chosen to generate the cut. Just like the fractional cuts, each of the generated mixed cuts will violate primal feasibility and will be restored to primal feasibility after applying the dual simplex method.

Let the nonzero coefficients $(\bar{a}_{rj})$ of the continuous variables x_j ($j \in J$) be partitioned into two sets: positive coefficients $(\bar{a}_{rj} > 0)$ and negative coefficients $(\bar{a}_{rj} < 0)$. Also, let $f_{rk} = \bar{g}_{rk} - \lfloor \bar{g}_{rk} \rfloor$ and $f_{r0} = \bar{b}_r - \lfloor \bar{b}_r \rfloor$ as before. It can be shown that a mixed cut due to Gomory can be derived (see Section 11.5) as

$$\sum_{j: \bar{a}_{rj} > 0} \bar{a}_{rj} x_j + \sum_{j: \bar{a}_{rj} < 0} \left(\frac{f_{r0}}{f_{r0} - 1} \right) \bar{a}_{rj} x_j + \sum_{k: f_{rk} \leq f_{r0}} f_{rk} y_k + \sum_{k: f_{rk} > f_{r0}} \frac{f_{r0}(1 - f_{rk})}{1 - f_{r0}} y_k \geq f_{r0}$$

We use the following numerical problem to show how to generate a mixed integer cut. The remaining procedure will be similar to that in Example 11.3, except that no rows corresponding to continuous variables are used for source rows to generate cuts.

Example 11.4 Solve the given MIP problem using a cutting plane approach.

$$\begin{aligned}
 &\text{Maximize} && z = 5x_1 + 3x_2 + 7y_1 + 2y_2 \\
 &\text{subject to} && 7x_1 + 8x_2 + 9y_1 + 3y_2 \leq 43 \\
 &&& 11x_1 + 4x_2 + 4y_1 + 5y_2 \leq 51 \\
 &&& \mathbf{x} \geq \mathbf{0} \\
 &&& \mathbf{y} \geq \mathbf{0} \text{ and integer}
 \end{aligned}$$

Solving the LP relaxation, we obtain an LP optimum $y_1 = 43/9$, with the following source row:

$$\frac{7}{9}x_1 + \frac{8}{9}x_2 + y_1 + \frac{1}{3}y_2 + \frac{1}{9}s_1 = \frac{43}{9}$$

Here, we have all positive coefficients and no negative coefficients for the continuous variables. Compute

$$\begin{aligned}
 f_{r0} &= \frac{7}{9} \\
 f_{r1} &= 0 \\
 f_{r2} &= \frac{1}{3}
 \end{aligned}$$

and we obtain a mixed integer cut

$$\frac{7}{9}x_1 + \frac{8}{9}x_2 + \frac{1}{3}y_2 + \frac{1}{9}s_1 \geq \frac{7}{9}$$

In Exercise 11.10, the reader is asked to continue on this example.

11.3 GROUP THEORETIC APPROACH

Gomory showed that the coefficient *row* vectors of the derived inequalities form a finite set that is closed under the operation of addition when the arithmetic operations are taken modulo 1 (i.e., integer parts are dropped). Such a set forms what is called a *group*. Furthermore, this group can have at most D elements, where D is the absolute value of the determinant of the current LP basis. If the starting basis is an identity matrix, then this group contains exactly D elements.

Gomory also showed that by relaxing nonnegative (but not integer) requirements of the current basic variables, an integer program can be transformed into one in which the *columns* of constraint coefficients and the right-hand side are elements of an abelian group. If this group problem (in terms of nonbasic variables only) is solved and a solution containing nonnegative values for all variables is obtained, then the

original integer program is solved. Before describing the role of group theory in integer programming, we need some definitions from group theory.

11.3.1 Group Theory Terminology

Definition 11.1 A *group* is a set of *elements* with a single operation (e.g., ordinary addition is taken modulo 1) defined on pairs of elements such that the operation is *closed*, *associative*, and for each element there exist unique *identity* and *inverse* elements.

Specifically, elements here are column vectors. *Closure* means that the sum of any two elements is also an element in the group. *Associativity* means that the operation satisfies the law of association such that $\mathbf{a} + (\mathbf{b} + \mathbf{c}) = (\mathbf{a} + \mathbf{b}) + \mathbf{c}$. The identity element has the property that any element in the group added to the identity (on the left or right) will result in itself. The inverse of an element has the property that the sum of the inverse and this element will result in the identity element.

Definition 11.2 A group is *finite* if it contains a finite number of elements. The *order* of a finite group is the number of elements comprising the group.

Definition 11.3 An *additive group* is a group whose operation is an ordinary addition (with modulo 1). An *abelian group* is one in which the operation is *commutative* such that $\mathbf{a} + \mathbf{b} = \mathbf{b} + \mathbf{a}$ for all elements $\mathbf{a}$ and $\mathbf{b}$ in the group.

Definition 11.4 A *cyclic group* is one in which there exists an element such that successive additions of the element to itself (or a scalar multiple of itself) will generate the entire group. If a group does not have such an element, then the group is called *noncyclic* or *acyclic*.

Definition 11.5 The number a is *congruent* to b modulo c if there exists an integer n such that $a - b = nc$. The *congruence relationship* (or simply *congruence*) is written as $a \equiv b \pmod{c}$. For example, $2 \equiv 5 \pmod{3}$, $-1 \equiv 3 \pmod{2}$, and $1 \equiv 0 \pmod{1}$.

To explain the above group definitions, we utilize tableau 1 of Example 11.3. The group contains three *elements*: $(1/3, 2/3, 2/3)^T$, $(2/3, 1/3, 1/3)^T$, and $(0, 0, 0)^T$. Hence, the group is *finite* with the *order* of 3. The group is *closed* because the sum (modulo 1) of any two elements will also result in an element in the group. For example, $(1/3, 1/3, 1/3)^T + (0, 0, 0)^T = (1/3, 1/3, 1/3)^T \pmod{1}$; $(2/3, 2/3, 2/3)^T + (0, 0, 0)^T = (2/3, 2/3, 2/3)^T \pmod{1}$; and $(1/3, 1/3, 1/3)^T + (2/3, 2/3, 2/3)^T = (0, 0, 0)^T \pmod{1}$. It can be easily shown that the *associative law* also holds:

$$\begin{aligned} & [(1/3, 1/3, 1/3)^T + (2/3, 2/3, 2/3)^T] + (0, 0, 0)^T \\ &= (1/3, 1/3, 1/3)^T + [(2/3, 2/3, 2/3)^T + (0, 0, 0)^T] \\ &= (0, 0, 0)^T \pmod{1} \end{aligned}$$

Element $(0, 0, 0)^T$ is the identity element for the entire group because any other element added to it will result in itself. Element $(1/3, 1/3, 1/3)^T$ is the *inverse element* of $(2/3, 2/3, 2/3)^T$ and vice versa because $(1/3, 1/3, 1/3)^T + (2/3, 2/3, 2/3)^T = (1, 1, 1)^T \pmod{1} = (0, 0, 0)^T$. The group is *additive* because its single operation on any coordinate is an ordinary addition. The group is abelian because the operation is communicative:

$$\begin{aligned} & [(1/3, 1/3, 1/3)^T + (2/3, 2/3, 2/3)^T] \pmod{1} \\ &= [(2/3, 2/3, 2/3)^T + (1/3, 1/3, 1/3)^T] \pmod{1} = (0, 0, 0)^T \end{aligned}$$

11.3.2 Deriving the Group (Minimization) Problem

Consider the integer program with *inequality* constraints

$$\begin{aligned} & \text{Maximize} && z = \mathbf{c}'^T \mathbf{y}' \\ & \text{subject to} && \mathbf{A}' \mathbf{y}' \leq \mathbf{b} \\ & && \mathbf{y}' \geq \mathbf{0} \text{ and integer} \end{aligned} \quad (11.6)$$

where $\mathbf{A}'$ is an $m \times n$ integer matrix, $\mathbf{y}'$ an n column vector, $\mathbf{b}$ an m integer column vector, and $\mathbf{c}'^T$ an n integer row vector. After adding m nonnegative slack variables, $y_{n+1}, y_{n+2}, \dots, y_{n+m}$, one to each inequality, we have the equivalent IP problem with *equality* constraints

$$\begin{aligned} & \text{Maximize} && z = \mathbf{c}^T \mathbf{y} \\ & \text{subject to} && \mathbf{A} \mathbf{y} = \mathbf{b} \\ & && \mathbf{y} \geq \mathbf{0} \text{ and integer} \end{aligned} \quad (11.7)$$

where $\mathbf{A} = (\mathbf{A}', \mathbf{I})$ is an $m \times (n + m)$ integer matrix, $\mathbf{I}$ an $m \times m$ identity matrix, $\mathbf{y}$ an $(n + m)$ column vector, $\mathbf{b}$ an m integer column vector, and $\mathbf{c}^T = (\mathbf{c}'^T, \mathbf{0}^T)$ an $(n + m)$ integer row vector.

Partitioning variables into sets of basic and nonbasic such that $\mathbf{y} = (\mathbf{y}_B, \mathbf{y}_N)^T$ with associated coefficients $\mathbf{c}^T = (\mathbf{c}_B^T, \mathbf{c}_N^T)$ and $\mathbf{A} = (\mathbf{B}, \mathbf{N})$, we have the partitioned problem

$$\begin{aligned} & \text{Maximize} && z = \mathbf{c}_B^T \mathbf{y}_B + \mathbf{c}_N^T \mathbf{y}_N \\ & \text{subject to} && \mathbf{B} \mathbf{y}_B + \mathbf{N} \mathbf{y}_N = \mathbf{b} \\ & && \mathbf{y}_B, \mathbf{y}_N \geq \mathbf{0} \text{ and integer} \end{aligned} \quad (11.8)$$

where $\mathbf{B}$ is an $m \times m$ basis matrix and $\mathbf{N}$ an $m \times n$ nonbasis matrix. To express $\mathbf{y}_B$ in terms of $\mathbf{y}_N$, we premultiply $\mathbf{B}^{-1}$ on the equalities in problem (11.8):

$$\mathbf{I} \mathbf{y}_B + \mathbf{B}^{-1} \mathbf{N} \mathbf{y}_N = \mathbf{B}^{-1} \mathbf{b}$$

or

$$\mathbf{y}_B = \mathbf{B}^{-1} \mathbf{b} - \mathbf{B}^{-1} \mathbf{N} \mathbf{y}_N$$

Substituting $\mathbf{y}_B$ into the objective function, we obtain the IP problem in terms of $\mathbf{y}_N$:

$$\begin{aligned} \text{Maximize } z &= \mathbf{c}_B^T \mathbf{B}^{-1} \mathbf{b} - (\mathbf{c}_B^T \mathbf{B}^{-1} \mathbf{N} - \mathbf{c}_N^T) \mathbf{y}_N \\ \text{subject to } \mathbf{I} \mathbf{y}_B + \mathbf{B}^{-1} \mathbf{N} \mathbf{y}_N &= \mathbf{B}^{-1} \mathbf{b} \\ \mathbf{y}_B, \mathbf{y}_N &\geq \mathbf{0} \text{ and integer} \end{aligned} \quad (11.9)$$

Relaxing integer requirements and letting $\mathbf{B}$ be the LP optimum basis, we have the optimum solution $\mathbf{y}_B = \mathbf{B}^{-1} \mathbf{b}$, $\mathbf{y}_N = \mathbf{0}$ with $z = \mathbf{c}_B^T \mathbf{B}^{-1} \mathbf{b}$. The optimality conditions are $\mathbf{c}_B^T \mathbf{B}^{-1} \mathbf{N} - \mathbf{c}_N^T \geq \mathbf{0}$. If $\mathbf{B}^{-1} \mathbf{b}$ happens to be an integer vector, the LP optimum solution is also an IP optimum. When the solution vector $\mathbf{B}^{-1} \mathbf{b}$ contains noninteger components, we must increase some $y_j \in \mathbf{y}_N$ from value 0 to some positive amount while maintaining the following conditions:

$$\mathbf{y}_B = \mathbf{B}^{-1} \mathbf{b} - \mathbf{B}^{-1} \mathbf{N} \mathbf{y}_N \geq \mathbf{0} \text{ and integer}$$

This poses two questions concerning the vector of basic variables $\mathbf{y}_B$:

- (1) When is $\mathbf{y}_B$ an integer vector?
- (2) When is $\mathbf{B}^{-1} \mathbf{b} - \mathbf{B}^{-1} \mathbf{N} \mathbf{y}_N \geq \mathbf{0}$?

We first address question (1) concerning integer vector. Denote the columns of matrix $\mathbf{B}^{-1} \mathbf{N}$ as $(\alpha_1, \alpha_2, \dots, \alpha_n)^T$ and the right-hand side column $\mathbf{B}^{-1} \mathbf{b}$ as α_0 . Then question (1) can be posed as a problem of finding some nonnegative components y_j ($j = 1, 2, \dots, n$) of $\mathbf{y}_N$ such that

$$\begin{aligned} \sum_j \alpha_j y_j &= \alpha_0 \pmod{1} \\ y_j &\geq 0 \text{ and integer } y_j \in \mathbf{y}_N \end{aligned} \quad (11.10)$$

Note that an integer vector α_j multiplied by any integer scalar y_j will yield a null vector $\mathbf{0} \pmod{1}$ and that the addition or subtraction of multiples of $y_j = 0 \pmod{1}$ will not destroy the congruence relationships. Thus, (11.10) can be reduced to one that contains only nonnegative fractional parts of α_j and α_0 , denoted by $\hat{\alpha}_j$ and $\hat{\alpha}_0$. We have the equivalent form

$$\begin{aligned} \sum_j \hat{\alpha}_j y_j &= \hat{\alpha}_0 \pmod{1} \\ y_j &\geq 0 \text{ and integer } y_j \in \mathbf{y}_N \end{aligned} \quad (11.11)$$

with the objective function

$$\text{Maximize } z = \mathbf{c}_B^T \mathbf{B}^{-1} \mathbf{b} - (\mathbf{c}_B^T \mathbf{B}^{-1} \mathbf{N} - \mathbf{c}_N^T) \mathbf{y}_N$$

Dropping $\mathbf{y}_B \geq \mathbf{0}$, letting $\hat{z} = -z + \mathbf{c}_B^T \mathbf{B}^{-1} \mathbf{b}$ and $\hat{\mathbf{c}}_N = \mathbf{c}_B^T \mathbf{B}^{-1} \mathbf{N} - \mathbf{c}_N^T$, and changing to a minimization problem, we obtain

$$\begin{aligned} \text{Minimize} \quad & \hat{z} = \sum_j \hat{c}_j y_j \\ \text{subject to} \quad & \sum_j \hat{\alpha}_j y_j = \hat{\alpha}_0 \pmod{1} \\ & y_j \geq 0 \text{ and integer } y_j \in \mathbf{y}_N \end{aligned} \quad (11.12)$$

where $\hat{c}_j \in \hat{\mathbf{c}}_N$. Note that all components of $\hat{\alpha}_j$ and $\hat{\alpha}_0$ are nonnegative fractions and $\hat{c}_j \geq 0$ for all j . Equation (11.12) is termed a *group minimization problem* (or *group problem*, for short).

11.3.3 Formulating a Group Problem

Example 11.5 Construct the group minimization problem for IP problem (11.1). For ease of reference, we restate it here.

$$\begin{aligned} \text{Maximize} \quad & z = 5y_1 - 2y_2 \\ \text{subject to} \quad & -y_1 + 2y_2 \leq 5 \\ & 3y_1 + 2y_2 \leq 19 \\ & y_1 + 3y_2 \geq 9 \\ & y_1, y_2 \geq 0 \text{ and integer} \end{aligned}$$

Adding a nonnegative slack variable to each inequality constraint, we have

$$\begin{aligned} \text{Maximize} \quad & z = 5y_1 - 2y_2 \\ \text{subject to} \quad & -y_1 + 2y_2 + y_3 = 5 \\ & 3y_1 + 2y_2 + y_4 = 19 \\ & -y_1 - 3y_2 + y_5 = -9 \\ & y_1, y_2, y_3, y_4 \geq 0 \text{ and integer} \end{aligned}$$

After solving the LP relaxation, we have the optimal basis $\mathbf{B}$, basic variables $\mathbf{y}_B$, and nonbasic variables $\mathbf{y}_N$. Representing in partitioned form, we have

$$\begin{aligned} \mathbf{y}_B = \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} &= \begin{pmatrix} 39/7 \\ 8/7 \\ 58/7 \end{pmatrix}, \mathbf{y}_N = \begin{pmatrix} y_4 \\ y_5 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \mathbf{B} = \begin{pmatrix} -1 & 2 & 1 \\ 3 & 2 & 0 \\ -1 & -3 & 0 \end{pmatrix}, \\ \mathbf{N} &= \begin{pmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{pmatrix}, \mathbf{b} = \begin{pmatrix} 5 \\ 19 \\ -9 \end{pmatrix}, \mathbf{c}_B^T = (5, -2, 0), \mathbf{c}_N^T = (0, 0) \end{aligned}$$

Calculating the components of the optimum tableau, we have

$$\mathbf{B}^{-1} = \begin{pmatrix} 0 & 3/7 & 2/7 \\ 0 & -1/7 & -3/7 \\ 1 & 5/7 & 8/7 \end{pmatrix}, \mathbf{B}^{-1}\mathbf{N} = \begin{pmatrix} 3/7 & 2/7 \\ -1/7 & -3/7 \\ 5/7 & 8/7 \end{pmatrix} = (\alpha_4, \alpha_5),$$

$$\mathbf{B}^{-1}\mathbf{b} = \begin{pmatrix} 39/7 \\ 8/7 \\ 58/7 \end{pmatrix} = \alpha_0, \mathbf{c}_B^T \mathbf{B}^{-1} \mathbf{N} - \mathbf{c}_N^T = (17/7, 16/7), \text{ and } \mathbf{c}_B^T \mathbf{B}^{-1} \mathbf{b} = 179/7$$

Converting to the notation for the group problem, we have

$$\hat{\alpha}_4 = \begin{pmatrix} 3/7 \\ 6/7 \\ 5/7 \end{pmatrix}, \hat{\alpha}_5 = \begin{pmatrix} 2/7 \\ 4/7 \\ 1/7 \end{pmatrix}, \hat{\alpha}_0 = \begin{pmatrix} 4/7 \\ 1/7 \\ 2/7 \end{pmatrix}, \hat{\mathbf{c}}_N^T = \mathbf{c}_B^T \mathbf{B}^{-1} \mathbf{N} - \mathbf{c}_N^T = (17/7, 16/7)$$

$$\hat{z} = -z + 179/7$$

Thus, we have the following associated group problem:

$$\begin{aligned} \text{Minimize } \hat{z} &= \frac{17}{7}y_4 + \frac{16}{7}y_5 \\ \text{subject to } &\begin{pmatrix} 3/7 \\ 6/7 \\ 5/7 \end{pmatrix}y_4 + \begin{pmatrix} 2/7 \\ 4/7 \\ 1/7 \end{pmatrix}y_5 = \begin{pmatrix} 4/7 \\ 1/7 \\ 2/7 \end{pmatrix} \pmod{1} \\ &y_4, y_5 \geq 0 \text{ and integer} \end{aligned} \quad (11.13)$$

Note that the system of congruence equations in (11.13) is equivalent to

$$\begin{pmatrix} 3 \\ 6 \\ 5 \end{pmatrix}y_4 + \begin{pmatrix} 2 \\ 4 \\ 1 \end{pmatrix}y_5 = \begin{pmatrix} 4 \\ 1 \\ 2 \end{pmatrix} \pmod{7} \quad (11.14)$$

11.3.4 Solving Group Problem as a Shortest Route Problem

Essentially there are two approaches available for solving the group minimization problem: (1) treating it as a special shortest route problem and solving it by a more efficient algorithm than the standard one, and (2) treating it as a variant of knapsack problem and solving it by a dynamic programming algorithm. Here, we shall describe how to construct the group problem as a special shortest route problem and leave the knapsack approach for the interested reader (see references in Section 11.5).

Now we want to construct a shortest route problem from the group minimization problem (11.13). First, we construct a directed network with nodes equal to the number of group elements D , where D is the absolute value of the determinant of

TABLE 11.7 Group Elements for Example 11.5

Group Element	$\mathbf{g}_1$ $\begin{pmatrix} 3 \\ 6 \\ 5 \end{pmatrix}$	$\mathbf{g}_2$ $\begin{pmatrix} 6 \\ 5 \\ 3 \end{pmatrix}$	$\mathbf{g}_3$ $\begin{pmatrix} 2 \\ 4 \\ 1 \end{pmatrix}$	$\mathbf{g}_4$ $\begin{pmatrix} 5 \\ 3 \\ 6 \end{pmatrix}$	$\mathbf{g}_5$ $\begin{pmatrix} 1 \\ 2 \\ 4 \end{pmatrix}$	$\mathbf{g}_6$ $\begin{pmatrix} 4 \\ 1 \\ 2 \end{pmatrix}$	$\mathbf{g}_0$ $\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$
$K_4\hat{\alpha}_4$	$1\hat{\alpha}_4$	$2\hat{\alpha}_4$	$3\hat{\alpha}_4$	$4\hat{\alpha}_4$	$5\hat{\alpha}_4$	$6\hat{\alpha}_4$	$7\hat{\alpha}_4$
$K_5\hat{\alpha}_5$	$5\hat{\alpha}_5$	$3\hat{\alpha}_5$	$1\hat{\alpha}_5$	$6\hat{\alpha}_5$	$4\hat{\alpha}_5$	$2\hat{\alpha}_5$	$7\hat{\alpha}_5$

the optimal basis $\mathbf{B}$. For (11.13), or $D = |\det \mathbf{B}| = 7$. Each group element is a 3-tuple column vector. To generate the entire group elements, we calculate

$$K_4\hat{\alpha}_4(\text{mod } 7) \text{ for } K_4 = 1, 2, \dots, 7$$
$$K_5\hat{\alpha}_5(\text{mod } 7) \text{ for } K_5 = 1, 2, \dots, 7$$

where

$$\hat{\alpha}_4 = \begin{pmatrix} 3 \\ 6 \\ 5 \end{pmatrix} \text{ and } \hat{\alpha}_5 = \begin{pmatrix} 2 \\ 4 \\ 1 \end{pmatrix}$$

resulting in Table 11.7. Here, each $\hat{\alpha}_j$ can generate the entire group $(\mathbf{g}_0, \mathbf{g}_1, \dots, \mathbf{g}_6)$. Thus, the group is cyclic and contains no cyclic subgroups.

Each directed arc $(\mathbf{g}_i, \mathbf{g}_j)$ is constructed by connecting the initial node $\mathbf{g}_i$ to the terminal node by setting $y_j = 1$ in the congruence relationship

$$\mathbf{g}_j = (\mathbf{g}_i + \hat{\alpha}_j y_j)(\text{mod } D)$$

which incurs a distance $\hat{c}_j$. Examples are

$$\mathbf{g}_1 = (\mathbf{g}_0 + \hat{\alpha}_4 y_4)(\text{mod } 7) \text{ with a cost } \hat{c}_4 = 17/7$$
$$\mathbf{g}_5 = (\mathbf{g}_0 + \hat{\alpha}_5 y_5)(\text{mod } 7) \text{ with a cost } \hat{c}_5 = 16/7$$
$$\mathbf{g}_2 = (\mathbf{g}_1 + \hat{\alpha}_4 y_4)(\text{mod } 7) \text{ with a cost } \hat{c}_4 = 17/7$$
$$\mathbf{g}_6 = (\mathbf{g}_3 + \hat{\alpha}_5 y_5)(\text{mod } 7) \text{ with a cost } \hat{c}_5 = 16/7$$

Similarly, we can construct the remaining arcs of a complete directed network as shown in Figure 11.7. Let node $\mathbf{g}_0$ be the origin (or source) node and $\mathbf{g}_6 (= \hat{\alpha}_0)$ be the destination (or sink) node of the shortest route problem. The objective of the problem is to find a route from the origin to the destination such that the total distance is minimal. To solve this problem, any standard shortest route algorithm will do. However, due to the special structure of the network, standard algorithms can be simplified in order to drastically increase the computational efficiency. For details about the algorithms, see Section 11.5.

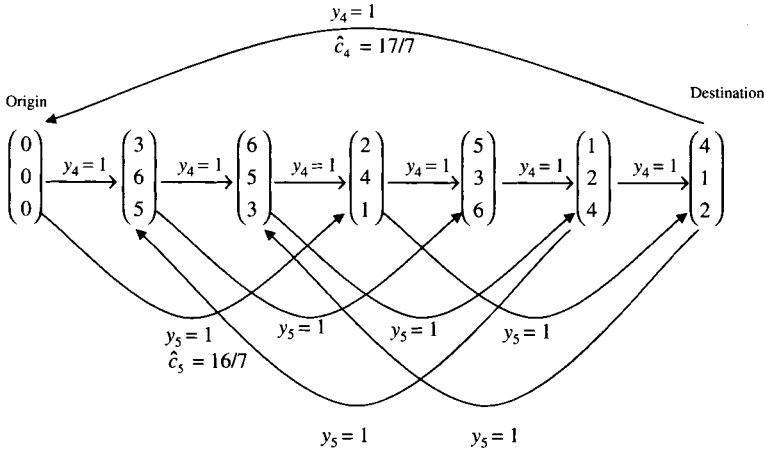


FIGURE 11.7 Directed network constructed by group problem.

By inspection of the above network, we can easily find the shortest route with $y_5 = 2$, $y_4 = 0$, and $\hat{z} = 32/7$.

11.3.5 Solving the Original Integer Program

Recall that when the group problem was derived from an integer program, the constraints $\mathbf{y}_B \geq \mathbf{0}$ were ignored. This implies that if the optimal solution $\mathbf{y}_N$ (which is a nonnegative integer) to the group problem also yields $\mathbf{y}_B \geq \mathbf{0}$, then we have already solved the integer program. If some basic variable turns out to be negative, then the optimal solution to the group problem is not an optimum to the original IP problem. To verify this for our example, we calculate

$$\mathbf{y}_B = \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} = \mathbf{B}^{-1}\mathbf{b} - \mathbf{B}^{-1}\mathbf{N}\mathbf{y}_N = \begin{pmatrix} 39/7 \\ 8/7 \\ 58/7 \end{pmatrix} - \begin{pmatrix} 3/7 & 2/7 \\ -1/7 & -3/7 \\ 5/7 & 8/7 \end{pmatrix} \begin{pmatrix} 0 \\ 2 \end{pmatrix} = \begin{pmatrix} 5 \\ 2 \\ 6 \end{pmatrix}$$

Since $\mathbf{y}_B \geq \mathbf{0}$, this optimal solution to the group problem is also an optimal solution to the original IP problem with $\mathbf{y} = (\mathbf{y}_B, \mathbf{y}_N)^T = (5, 2, 6, 0, 2)^T$. Should any component of $\mathbf{y}$ turns out to be negative, we continue to find the next shortest route and test for $\mathbf{y}_B \geq \mathbf{0}$ until such condition is obtained or no more route can be found, in which case the original IP problem has no feasible solution. However, this process is quite complicated and difficult to implement.

An alternative method is apply a branch-and-bound enumeration scheme that implicitly examines all possible integer solutions to the group problem by successively adding constraints of the form $y_j \geq C$ for each j , where C begins with value 0 and is incremented by 1. Morito (1976) utilized this basic bookkeeping scheme with a particular branching rule and information from the optimal solution to the group problem to create an efficient branch-and-bound algorithm. Again, the scheme is not implemented in practice.

In conclusion, there are still several technical hurdles to overcome before the group theoretic approach can be implemented as a general, stand-alone IP solver. However, the faces of corner polyhedron generated by the group problem are strong cuts (or valid inequalities) that can be incorporated into the novel branch-and-cut approach, in the same manner as other cutting planes are utilized.

11.4 GEOMETRIC CONCEPTS

Now we discuss the solution space to the pure integer program defined by the structural variables y' in (11.6), and the solution space to the group problem defined by nonbasic variables y_N in (11.12). Many important geometric concepts will be covered, including polyhedron, convex hull, corner polyhedron, and faces. For illustration, we reduce Example 11.5 to a two-constraint problem by dropping the first constraint.

Example 11.6

$$\begin{aligned} \text{Maximize} \quad & z = 5y_1 - 2y_2 \\ \text{subject to} \quad & 3y_1 + 2y_2 \leq 19 \\ & -y_1 - 3y_2 \leq -9 \\ & y_1, y_2 \geq 0 \text{ and integer} \end{aligned} \quad (11.15)$$

Adding nonnegative slack variables y_3 and y_4 to constraints 1 and 2, respectively, we find the optimum LP solution containing basic variables y_1 and y_2 , and so

$$\begin{aligned} y_B &= \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} 39/7 \\ 8/7 \end{pmatrix}, y_N = \begin{pmatrix} y_3 \\ y_4 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, B = \begin{pmatrix} 3 & 2 \\ -1 & -3 \end{pmatrix}, \\ N &= \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, b = \begin{pmatrix} 19 \\ -9 \end{pmatrix}, B^{-1} = \begin{pmatrix} 3/7 & 2/7 \\ -1/7 & -3/7 \end{pmatrix}, \\ B^{-1}N &= \begin{pmatrix} 3/7 & 2/7 \\ -1/7 & -3/7 \end{pmatrix}, B^{-1}b = \begin{pmatrix} 39/7 \\ 8/7 \end{pmatrix}, D = |\det B| = 7 \\ c_B^T B^{-1}N - c_N^T &= (5, -2) \begin{pmatrix} 3/7 & 2/7 \\ -1/7 & -3/7 \end{pmatrix} - (0, 0) = (17/7, 16/7), \\ z &= c_B^T B^{-1}b = 179/7 \end{aligned}$$

From (11.12), we obtain the associated group problem

$$\begin{aligned} \text{Minimize} \quad & \hat{z} = \frac{17}{7}y_3 + \frac{16}{7}y_4 \\ \text{subject to} \quad & \begin{pmatrix} 3/7 \\ 6/7 \end{pmatrix}y_3 + \begin{pmatrix} 2/7 \\ 4/7 \end{pmatrix}y_4 = \begin{pmatrix} 4/7 \\ 1/7 \end{pmatrix} \pmod{1} \\ & y_3, y_4 \geq 0 \text{ and integer} \end{aligned} \quad (11.16)$$

Solving the problem as a shortest route problem, we obtain the solution $y_3 = 0$, $y_4 = 2$, which gives

$$\mathbf{y}_B = \mathbf{B}^{-1}\mathbf{b} - \mathbf{B}^{-1}\mathbf{N}\mathbf{y}_N = \begin{pmatrix} 39/7 \\ 8/7 \end{pmatrix} - \begin{pmatrix} 3/7 & 2/7 \\ -1/7 & -3/7 \end{pmatrix} \begin{pmatrix} 0 \\ 2 \end{pmatrix} = \begin{pmatrix} 5 \\ 2 \end{pmatrix} \geq \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

Since all components of vector $\mathbf{y}_B$ are nonnegative, the optimal solution to the original integer program is $\mathbf{y} = (5, 2, 0, 2)^T$ with objective value

$$z = -\hat{z} + \mathbf{c}_B^T \mathbf{B}^{-1} \mathbf{b} = -32/7 + 179/7 = 21$$

11.4.1 Various Polyhedrons in Original Space

From Chapter 8, we know that the solution region of the LP relaxation defined by the constraints and the nonnegative restrictions in (11.15) is a polyhedron, say P . In our example, P is the triangle area enclosed by points $\mathbf{a}'$, $\mathbf{b}'$, and $\mathbf{c}'$, shown in Figure 11.8. A basis $\mathbf{B}$ selected from the coefficient matrix yields the basic feasible solution $\mathbf{y} = (\mathbf{y}_B, \mathbf{y}_N)^T = (\mathbf{B}^{-1}\mathbf{b}, \mathbf{0})^T$, which is an extreme point to P . In our example, $\mathbf{a}'$, $\mathbf{b}'$, and $\mathbf{c}'$ are extreme points of P . Taking the objective function into consideration, we obtain an optimal extreme point $\mathbf{c}' = (39/7, 8/7)^T$ with objective value $z = 25.57$. The objective function, indicated by a dotted line and passing through vertex $\mathbf{c}'$, is plotted in Figure 11.8.

Recall that the congruence constraints of the group problem (11.16) are obtained from those of the IP program by ignoring the nonnegative requirements on the basic

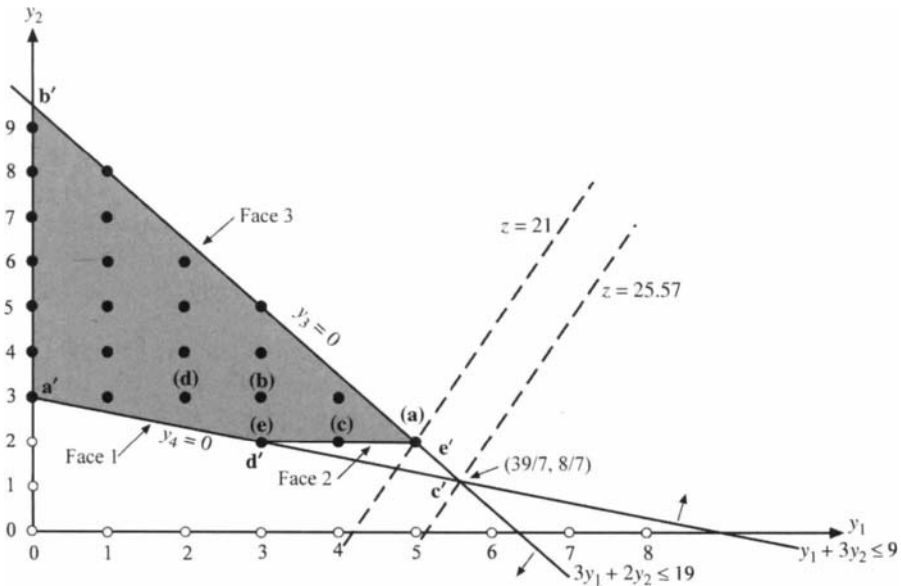


FIGURE 11.8 Corner polyhedron in original space.

variables $\mathbf{y}_B$. In the original solution space, this is equivalent to dropping those inequalities from $\mathbf{A}'\mathbf{y}' \leq \mathbf{b}$ and $\mathbf{y}' \geq \mathbf{0}$ in (11.6), which are *nonbinding* at vertex $\mathbf{e}'$. In this example, there are two *binding* constraints at $\mathbf{c}'$, namely,

$$3y_1 + 2y_2 < 19$$

$$-y_1 - 3y_2 < -9$$

and two nonbinding constraints $y_3 > 0$, $y_4 > 0$.

An inequality constraint that is nonbinding at $\mathbf{v}'$ implies that it takes on a strict inequality ($<$) when substituting $\mathbf{v}'$ into it. As a result, the associated slack variable is strictly positive because only a basic variable can be positive. In general, when constraint i is binding at $\mathbf{v}'$, the associated slack variable y_{n+i} must take on value 0 and the corresponding hyperplane $y_{n+i} = 0$ must pass through $\mathbf{v}'$. In our example, the hyperplanes $y_3 = 0$ and $y_4 = 0$ must pass through $\mathbf{c}'$. This implies that these hyperplanes intersect at $\mathbf{c}'$. As shown in Figure 11.8, the hyperplanes $3y_1 + 2y_2 = 19$ and $y_1 + 3y_3 = 9$ in the original solution space correspond to the hyperplanes $y_3 = 0$ and $y_4 = 0$ in the space of nonbasic variables $\mathbf{x}_N$.

On the other hand, if $y_i > 0$ or y_i is a basic variable (where $i: y_i \in \mathbf{y}_B$), then the hyperplane $y_i = 0$ does not intersect the vertex $\mathbf{v}'$. Thus, ignoring the nonnegativity requirement on a basic variable y_i is equivalent to allowing y_i to take on any value when the corresponding hyperplane does not pass through $\mathbf{v}'$. In our example, we have strict inequalities $y_1 > 0$ and $y_2 > 0$, so hyperplanes $y_1 = 0$ and $y_2 = 0$ do not intersect at $\mathbf{c}'$. Therefore, these hyperplanes can be dropped with affecting the solution.

After dropping these hyperplanes, we obtain a polyhedron $\mathbf{P}$ with vertex $\mathbf{c}' = (39/7, 8/7)^T$ as shown in Figure 11.8. Adding the integer requirements yields the lattice points inside $\mathbf{P}$, indicated by dots. The convex hull of these lattice points may be viewed as the feasible region to the group problem. The *corner polyhedron* is obtained by dropping the hyperplanes $y_1 = 0$ and $y_2 = 0$, and taking the convex hull of the integer solutions. In Figure 11.8, the polyhedron bounded by integer points $\mathbf{a}'$, $\mathbf{d}'$, $\mathbf{e}'$, and $\mathbf{b}'$ forms a corner polyhedron, which is denoted by $\mathbf{P}^{\mathbf{y}'}$ and indicated by the shaded area. The hyperplanes that bound $\mathbf{P}^{\mathbf{y}'}$ are the *faces* of the corner polyhedron. Recall that in n -dimensional space, a face is an $(n - 1)$ -dimensional hyperplane. Since $n = 2$ in our example, so a face must be a line such as face 1, face 2, and face 3, indicated in Figure 11.8.

Note that any integer point $\mathbf{y}'$ in $\mathbf{P}^{\mathbf{y}'}$ will produce nonnegative integer values for the nonbasic variables $\mathbf{y}_N$ that satisfy the congruence relationship in (11.16). Furthermore, a vertex of the corner polyhedron is an optimal solution for the integer program. The vertex $\mathbf{e}'$ is at the intersection of faces 2 and 3 in Figure 11.8. These inequalities are the strongest that can be obtained from the integer program without using the requirements of $\mathbf{y}_B \geq \mathbf{0}$. Therefore, it is worthwhile finding the face inequalities and using them in a cutting plane algorithm or in a branch-and-bound algorithm. To enable generating these valid inequalities, we next investigate the corner polyhedron in $\mathbf{y}_N$ space.

11.4.2 Corner Polyhedron in Solution Space of Nonbasic Variables

The solution region defined by the constraints in (11.12), or (11.16) in particular for Example 11.6, can also be plotted in $\mathbf{y}_N$ space. To do this, recall that the congruence relationship

$$\sum_j \hat{\alpha}_j y_j = \hat{\alpha}_0 \pmod{1}$$

is equivalent to

$$\sum_j f_{ij} y_j = f_{i0} \pmod{1} \quad (i = 1, 2, \dots, m) \quad (11.17)$$

where $y_j \in \mathbf{y}_N$, and f_{ij} is the i th component ($i = 1, 2, \dots, m$) of $\hat{\alpha}_i$ ($j = 1, 2, \dots, n$). Because the left-hand side in (11.12) must differ from the right-hand side by an integer amount and because $y_j \geq 0$ and integer, the congruence can hold only when the left-hand side is $f_{i0}, 1 + f_{i0}, 2 + f_{i0}$, and so on. That is, equation (11.12) implies

$$\sum_{j=1}^n f_{ij} y_j \geq f_{i0} \quad (i = 1, 2, \dots, m) \quad (11.18)$$

Note that the inequalities (11.18) are the Gomory fractional cuts obtained from the m equalities $\mathbf{y}_B = \mathbf{B}^{-1}\mathbf{b} - \mathbf{B}^{-1}\mathbf{N}\mathbf{y}_N$. For Example 11.6, we have

$$\begin{aligned} 3/7y_3 + 2/7y_4 &\geq 4/7 \\ 6/7y_3 + 4/7y_4 &\geq 1/7 \\ y_3, y_4 &\geq 0 \text{ and integer} \end{aligned} \quad (11.19)$$

Note that the second constraint is redundant and may be dropped without affecting the solution.

Every y_j that satisfies (11.17) will also satisfy (11.18). However, the converse is not true because $y_1, y_2, \dots, y_n$ satisfying (11.18) may yield a value for $\sum_j \hat{\alpha}_j y_j$ that is not an integer plus $\hat{\alpha}_0$. Therefore, the constraints

$$\sum_j \hat{\alpha}_j y_j = \hat{\alpha}_0 \pmod{1}$$

$$\mathbf{y}_N \geq 0 \text{ and integer}$$

for the group problem may be viewed in $\mathbf{y}_N$ space by plotting the inequalities (11.18) along with $\mathbf{y}_N \geq \mathbf{0}$ and integer, and taking the convex hull of the points satisfying (11.18). This region is termed the *corner polyhedron* $\mathbf{P}_N$.

We plot (11.19), or (11.16) to be exact, in the solution space of $\mathbf{y}_N = (y_3, y_4)$ shown in Figure 11.9. The solution set is an unbounded polyhedron with two extreme points at $(4/3, 0)$ and $(0, 2)$. The circled lattice points are those that satisfy

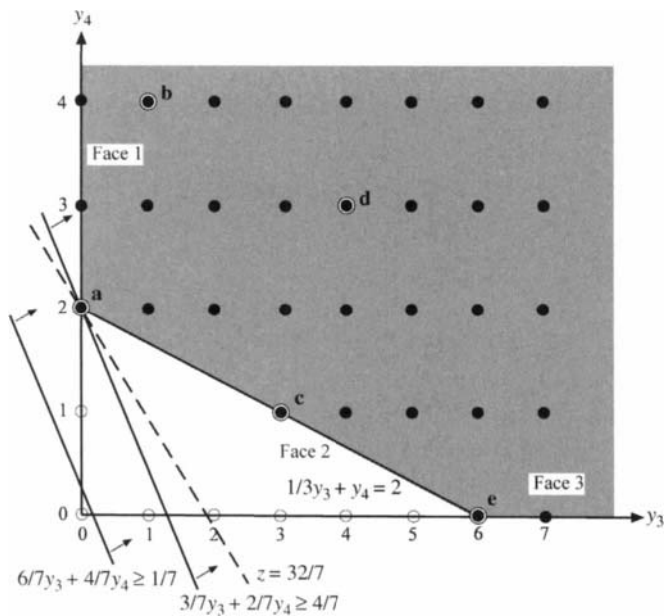


FIGURE 11.9 Corner polyhedron in y_N space.

the congruence relationship. Some of them are marked by **a**, **b**, **c**, and **d**, and **e** in Figure 11.9:

$$\mathbf{a} = \begin{pmatrix} 0 \\ 2 \end{pmatrix}, \mathbf{b} = \begin{pmatrix} 1 \\ 4 \end{pmatrix}, \mathbf{c} = \begin{pmatrix} 3 \\ 1 \end{pmatrix}, \mathbf{d} = \begin{pmatrix} 4 \\ 3 \end{pmatrix}, \mathbf{e} = \begin{pmatrix} 6 \\ 0 \end{pmatrix}$$

Using

$$\mathbf{y}_B = \mathbf{B}^{-1} \mathbf{b} - \mathbf{B}^{-1} \mathbf{N} \mathbf{y}_N$$

we obtain $\mathbf{y}_B$ components, as shown in Table 11.8, for the corresponding points in the original solution space.

TABLE 11.8 Integer Solution Points

Point	a	b	c	d	e
$\mathbf{y}_N = \begin{pmatrix} y_3 \\ y_4 \end{pmatrix}$	$\begin{pmatrix} 0 \\ 2 \end{pmatrix}$	$\begin{pmatrix} 1 \\ 4 \end{pmatrix}$	$\begin{pmatrix} 3 \\ 1 \end{pmatrix}$	$\begin{pmatrix} 4 \\ 3 \end{pmatrix}$	$\begin{pmatrix} 6 \\ 0 \end{pmatrix}$
$\mathbf{y}_B = \begin{pmatrix} y_1 \\ y_2 \end{pmatrix}$	$\begin{pmatrix} 5 \\ 2 \end{pmatrix}$	$\begin{pmatrix} 4 \\ 3 \end{pmatrix}$	$\begin{pmatrix} 4 \\ 2 \end{pmatrix}$	$\begin{pmatrix} 3 \\ 3 \end{pmatrix}$	$\begin{pmatrix} 3 \\ 2 \end{pmatrix}$

To see the correspondence of these points in the original solution space in Figure 11.8, we label such points with a bracket (a), (b), (c), (d), (e).

A convex set may be formed by the convex combinations of these circled lattice points (integer solutions for both $\mathbf{y}_B$ and $\mathbf{y}_N$) in the unbounded polyhedral cone. That set is a corner polyhedron, denoted by P_N . The corner polyhedron for this example problem is the shaded area in Figure 11.9. Unlike the corner polyhedron in the original space, the corner polyhedron in $\mathbf{y}_N$ space always satisfies $\mathbf{y}_N \geq \mathbf{0}$. Also, note that this corner polyhedron is an unbounded convex set. This is always true because if $\mathbf{y}_N$ satisfies (11.17), then for any integer K , the value KD (where D is the absolute value of the determinant of B) added to each of its components will also satisfy (11.18). To show their correspondences, we use the same labels for the faces in Figure 11.8. The faces (faces 1 through 3) of this corner polyhedron are marked in Figure 11.9.

It can be shown (Salkin and Marthur, 1989) that $\mathbf{y}_B$ is an extreme point of corner polyhedron $P^{y'}$ if and only if the corresponding $\mathbf{y}_N$ is an extreme point of P_N using the relationship between faces. In our example, $\mathbf{y}_N = (0, 2)^T$ is an extreme point in P_N denoted by a, which corresponds to $\mathbf{y}_B = (5, 2)^T$, an extreme point e' in $P^{y'}$. Similarly, extreme point e in P_N corresponds to extreme point d' in $P^{y'}$.

11.5 NOTES

Sections 11.1

The branch-and-bound method was originated by Land and Doig (1960) and refined by Dakin (1965), which is regarded as the basis of the current algorithm.

Sections 11.2

Three fundamental cutting plane methods, dual fractional, dual mixed integer, and primal all-integer are developed by Gomory (1958, 1960, 1963). For details on these and other cutting plane methods, see Garfinkel and Nemhauser (1972), and Salkin and Mathur (1989).

Sections 11.3

Gomory (1960, 1963, 1965) developed group properties of IP program. For exposition of group theory approach to solving integer programs, see Hu (1969), Chen (1970), Chen and Zionts (1972), and Salkin and Mathur (1989).

Section 11.4

Special shortest route algorithms for solving the group problem are given in Hu (1968) and Chen and Zionts (1976). Shapiro (1968) treated the group problem as a variant of knapsack problem that is solved by a dynamic programming algorithm.

An equivalent group problem can be obtained by transforming the LP optimal basis into a Smith's normal form (Hu, 1969), which can be solved more efficiently. The form is a diagonal matrix (all elements not on the diagonal are zero) denoted by $\text{diag}(d_1, d_2, \dots, d_n)$, such that d_i ($i = 1, 2, \dots, m$) is a positive integer, d_i divides d_{i+1} ($i = 1, 2, \dots, m-1$), and $D = \det \mathbf{B}$ can be determined by the product of diagonal elements d_i . The diagonalization process is performed by the elementary row and column operations.

There are three types of group problems, depending on the number of $\hat{\alpha}_j$ elements that can be used to generate the entire group: (1) each element (like Example 11.5), (2) some but not each element, and (3) no single element. The groups determined by types 1 and 2 are cyclic. The group type 3 is acyclic, whose elements are determined by the *direct sum* of cyclic subgroups. For details see Salkin and Mathur (1989), for example.

A sufficient condition for $\mathbf{y}_B \geq \mathbf{0}$ is used to ensure that an optimum solution $\mathbf{y}_N$ to the group problem is also an optimum solution $\mathbf{y} = (\mathbf{y}_B, \mathbf{y}_N)$ to the original integer program. Several sufficient conditions have been developed. See Gomory (1965), Hu (1968), and Zionts (1974), for example. Sufficiency conditions have little use in application although are of theoretical value.

11.6 EXERCISES

- 11.1** Solve the problem in Example 11.1 again using branch-and-bound. This time, start your branching with y_2 . Graphically show the changes in the feasible region at each node. Apply the depth-first rule.
- 11.2** Use the branch-and-bound method to solve the following IP problem. Show your solution procedure graphically as in Figures 11.1–11.3.

$$\begin{array}{ll}\text{Maximize} & y_1 + 3y_2 \\ \text{subject to} & y_1 + 5y_2 \leq 12 \\ & y_1 + 2y_2 \leq 8 \\ & y_1, y_2 \geq 0 \text{ and integer}\end{array}$$

- 11.3** Solve the following IP problem using the branch-and-bound method. Apply the best-bound-first rule. At each node, branch on the variable with fraction value closest to 0.5 first. Label the nodes in the order they are generated.

$$\begin{array}{ll}\text{Maximize} & 3y_1 + 2y_2 + y_3 + 2y_4 \\ \text{subject to} & y_1 - y_2 + 2y_3 + y_4 \leq 11 \\ & y_2 + y_3 + y_4 \leq 7 \\ & 3y_1 - y_3 - 3y_4 \leq 5 \\ & \mathbf{y} \geq \mathbf{0} \text{ and integer}\end{array}$$

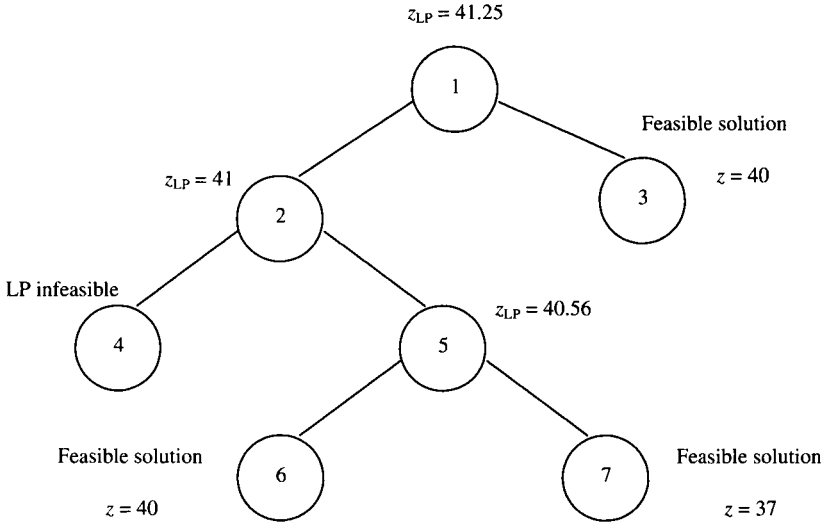


FIGURE 11.10 A branch-and-bound tree.

- 11.4** Given a branch-and-bound tree for a maximization problem (as shown in Figure 11.10), where the number of the nodes shows the order they are generated, write the reason for each pruning.
- 11.5** Mary is the cashier of a fast food restaurant. Coins of four values are used to get customers the change: 1¢, 5¢, 10¢, and 25¢. Suppose now she is to give a customer 91¢ of change. What is the minimum number of coins needed? Formulate as an IP model and solve using branch-and-bound method. (*Hint*: convert to maximization problem first). Verify your solution with LINGO®.
- 11.6** Solve the following IP problem using branch-and-bound. Apply the rule of best-bound-first. At each node, branch on the variable with least index first.

Maximize $10y_1 - y_2 + 5y_3 + 3y_4 - 7y_5$
subject to $3y_1 + 2y_2 - y_3 - y_4 - 3y_5 \leq 3$
 $3y_1 - 2y_2 + 2y_3 - 2y_4 + 5y_5 \leq 7$
 $y_1 - y_2 + y_4 - y_5 \leq 3$
 $y_1, y_2, y_3, y_5 \geq 0$ and integer, $y_4 = \{0, 1\}$

- 11.7** Assume in Exercise 11.6 that y_3, y_5 are continuous. Solve the problem again using branch-and-bound method. Apply the rule of depth-first.
- 11.8** Consider the IP problem in Exercise 11.3. Which of the two cutting planes (fractional or mixed) could be applied to it? Why? No need to solve the problem to optimum.

11.9 Let $S = \{y \geq 0 \text{ and integer } y_1 - y_2 \leq 2, 3y_1 + y_2 \leq 21, y_1 + 5y_2 \leq 34\}$.

(a) Find an inequality description of $\text{Conv}(S)$.

(b) Find the extreme points of $\text{Conv}(S)$.

11.10 Consider the problem in Example 11.4. Generate another Gomory fractional cut using the row corresponding to the basic variable of y_2 .

11.11 Consider the integer problem

$$\begin{aligned} &\text{Maximize} && 5y_1 - 2y_2 \\ &\text{subject to} && 11y_1 - y_2 \leq 21 \\ &&& 3y_1 - 3y_2 \leq 5 \\ &&& y_1 \leq 4 \\ &&& y \geq 0 \text{ and integer} \end{aligned}$$

(a) Solve the LP relaxation and show the optimal simplex tableau.

(b) Is there any basic variable fractional in the optimal tableau? If yes, find a Gomory fractional cut based on it.

11.12 Consider the following IP problem.

$$\begin{aligned} &\text{Maximize} && 3y_1 + y_2 + 2y_3 + 3y_4 \\ &\text{subject to} && -y_1 + 3y_2 + y_3 - 2y_4 \leq 17 \\ &&& 7y_1 + 3y_3 + y_4 \leq 23 \\ &&& y_1 + 2y_2 \leq 11 \\ &&& y_2 + 3y_4 \leq 13 \\ &&& y \geq 0 \text{ and integer} \end{aligned}$$

Solve it using Gomory's fractional cutting plane. Show the cutting planes you generated at each step.

11.13 Generate a group of fractional cuts based on variable y_2 in Exercise 11.2.

11.14 Assuming in Exercise 11.12 that y_2 is continuous, solve the problem using Gomory mixed integer cut.

11.15 Complete solving Example 11.4 using Gomory mixed integer cutting plane algorithm. Compare your results with the solution obtained in Example 11.2 using a branch-and-bound method.

For each of the problems in Exercises 11.16–11.18, (a) formulate the group problem, (b) construct the group problem as a directed network, and (c) solve the shortest route problem by inspection.

TABLE 11.9 An Optimal Tableau

Basic Variable	z	y_1	y_2	y_3	y_4	y_5	y_6	y_7	RHS
z	1	7/10	0	69/10	0	4/10	0	19/10	139/10
y_2	0	5/10	1	5/10	0	0	0	5/10	25/10
y_4	0	16/10	0	2/10	1	2/10	0	2/10	32/10
y_6	0	16/10	0	-8/10	0	2/10	1	2/10	2/10

11.16

Maximize $z = 4y_1 + 3y_2 - 5y_3 + 2y_4$
subject to $7y_1 - 2y_2 + 5y_4 \leq 11$
 $-y_3 - y_4 \leq -3$
 $y_1 + 2y_2 + y_3 \leq 5$
 $y_1, y_2, y_3, y_4 \geq 0$ and integer

Add slack variables $y_5, y_6,$ and y_7 to the inequality constraints, respectively. Then solve the LP relaxation and obtain the following optimum tableau (Table 11.9).

11.17 (Balinski, 1965)¹

Maximize $z = 4y_1 + 5y_2$
subject to $-y_1 - y_2 \leq -5$
 $-3y_1 - 2y_2 \leq -7$
 $y_1, y_2 \geq 0$ and integer

Adding nonnegative slack variables y_3 and y_4 to constraints 1 and 2, respectively, we find the optimum LP solution containing basic variables y_1 and y_2 .

11.18 (Zionts, 1974) (used with permission)

Maximize $z = 5y_1 + 2y_2$
subject to $2y_1 + 2y_2 \leq 9$
 $3y_1 + y_2 \leq 11$
 $y_1, y_2 \geq 0$ and integer

¹ Reprinted with permission from author (see Bibliography). Copyright 1965 The Institute for Operations Research and Management Sciences, 7240 Parkway Drive, Suite 300, Hanover, MD 21076.

Adding slack variables y_3 and y_4 to constraints 1 and 2, respectively, and ignoring the integer requirements, we solve the LP relaxation and obtain a noninteger optimum solution, written in equation form, as follows:

$$\begin{aligned} \text{Maximize} \quad & z - 1/4y_3 - 6/4y_4 = 75/4 \\ \text{subject to} \quad & y_2 + 3/4y_3 - 2/4y_4 = 5/4 \\ & y_1 - 1/4y_3 + 2/4y_4 = 13/4 \\ & y_1, y_2, y_3, y_4 \geq 0 \end{aligned}$$

where $\mathbf{y_B} = (y_2, y_1)$.

- 11.19** For the problem in Exercise 11.17, draw the solution space of the IP program and identify the LP polyhedron, convex hull, corner polyhedron, and faces.
- 11.20** For the problem in Exercise 11.17, draw the solution space of nonbasic variables and identify faces and corner polyhedron.
- 11.21** For the problem in Exercise 11.17, relate all faces between the solution space of the structural variables and the solution space of the nonbasic variables.
- 11.22** For the problem in Exercise 11.18, draw the solution space of the IP program and identify the LP polyhedron, convex hull, corner polyhedron, and faces.
- 11.23** For the problem in Exercise 11.18, draw the solution space of nonbasic variables and identify faces and corner polyhedron.
- 11.24** For the problem in Exercise 11.18, relate all faces between the solution space of the structural variables and the solution space of the nonbasic variables.