ADDIS ABABA UNIVERSITY

COLLEGE OF COMPUTATIONAL AND NATURAL SCIENCES

DEPARTMENT OF MATHEMATICS

Mixed Integer Programming Problems using Branch & Bound and Gomory's Methods

By: Zinash Getatchew Assen ID.No.:GSE/4277/10
Stream: Optimization

Advisor: Berhanu Guta(PhD)

"A Project Submitted to the Department of Mathematics of Addis Ababa University in Partial Fulfillment of the Requirements for the Master of Science(MSc.) Degree in Mathematics."

June, 2020
Addis Ababa, Ethiopia

Addis Ababa University
College Of Computational And Natural Sciences
School of Graduate Studies
Department of Mathematics

The undersigned hereby certify that they have read and recommend to the department of Mathematics for acceptance of this project entitled "**Mixed Integer Programming Problems using Branch & Bound and Gomory's Methods**" by Zinash Getachew in partial fulfillment of the requirements for the degree of Master of Science in Mathematics.

By: Zinash Getachew Sign._____ Date_____

Advisor: Dr. Berhanu Guta Sign._____ Date_____

This project is approved by board of examiners:

Chairman of the department or Graduate committee: _____

Sign._____ Date_____

Examiner 1: _____

Sign._____ Date_____

Examiner 2: _____

Sign._____ Date_____

Contents

Acknowledgements

First, I would like to express my sincere gratitude to my advisor Dr. Berhanu Guta for his support throughout this work and many helpful assisting in finishing this project. Without his support, expertise and guidance, this project's completion would not have been possible.

I am also thankful to all Mathematics department staff members in Addis Ababa University and to all my friends for their cooperation on the success of this work.

Finally, I would like to express my appreciation to my families for their endless love and continuous encouragement.

Abstract

Mixed integer programming is a linear programming with some of the decision variables are integer and some of them are continuous. Thus, unlike to linear programming it some what complex to solve due to the integer decision variables. Some of the complexities are minimized using different methods such as Gomory's cutting plane, Branch and bound methods. In this project, we presented the solutions of mixed integer linear programming using Gomory's Fractional cutting plane through removing the unnecessary region from its linear programming relaxation to obtain the integer solution and branch and bound by dividing the problem to smaller sub-problems which are more simple than the original problem. Therefore, based on these concepts, the corresponding algorithms are presented with illustrative examples.

Keywords: Linear Programming, Simplex method, Dual Simplex method, Mixed integer programming, Cutting plane method, Branch and bound method.

List of Tables

List of Notations

- LP: Linear programming
- LPP: Linear programming problem
- ILP: Integer Linear programming
- PILP: Pure Integer Linear programming
- PILPP: Pure Integer Linear programming problem
- MILP: Mixed Integer Linear programming
- MILPP: Mixed Integer Linear programming problem
- BFS: Basic feasible solution
- LPR: Linear Programming Relaxation
- R^n:n-dimensional real space
- Z: the set of integers
- Ip: Integer Programming
- BS: Basic Solution
- X^t: Transpose of a matrix X

Chapter 1

Introduction

In many practical problems, some of the decision variables or all the decision variables actually make sense only if they have integer values. For instance, it is often necessary to assign people, machines and vehicles to activities in integer quantities. Such type of problems are commonly known as integer programming problem (IPP). The integer programming problem may be linear or nonlinear based on the expressions used to describe it. Therefore, for an integer programming problem with linear expression only is called integer linear programming problem (ILPP), otherwise it is nonlinear integer programming problem (NIPP). Moreover, if requiring integer values is the only way in which a problem deviates from a linear programming formulation, then it is called an integer linear programming (ILP) problem.

Based on the divisibility of decision variables, integer programming (IP) problem may be pure IP or Mixed IP or Binary IP. If only some of the variables are required to have integer values (so the divisibility assumption holds for the rest), this model is referred to as mixed integer programming (MIP). When all the decision variables are required integer values the model is called pure integer programming (PIP). An integer program in which the integer variables are restricted to be 0 or 1 is called a binary IP (BIP) [10, 6, 5].

Generally, mixed integer linear programming problem (MILPP) is a linear optimization problems with two types of variables; variables taking values in an integer domain, and variables taking values in a continuous domain. It was used with the decision variables; creating a second shift, purchasing additional equipment, determining the required work force, and other alternatives involving new manners of work distribution that make it possible to separate certain operations from some workplaces and integrate them into others to minimize production costs. The fact that MILPPs naturally appear

in many contexts has led to an increased interest in the design of strong algorithms for different variants of the problem. Unfortunately, MILPPs are in general more difficult to solve than linear programming problems but these complexities can be overcome using different techniques like ILP [11, 7, 4].

The most common classical approaches for solving integer programs are branch and bound, cutting plane, and group theoretic. Although all approaches are capable of solving integer programs, their degrees of success vary in 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. 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. Branch-and-bound is a general-purpose approach capable of solving pure IP, mixed IP, and binary IP problems. In the cutting plane approach, one or more 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 [5].

In this project, we deal with the classical approaches branch and bound, cutting plane methods for mixed integer programming. mixed integer programming problems are not easy to solve we we compare with linear programming problems because of the additional restriction in the decision variables. Thus, in this project to find the optimal solution of MIP we apply these two basic methods. In this study, we presented; introduction in the first part, linear programming problems in the second part and mixed IP in the third part.

Chapter 2

Basic Concepts

2.1 Linear Programming Problems

Linear programming (LP) is concerned with the optimization (minimization or maximization) of a linear function while satisfying a set of linear equality and/or inequality constraints or restrictions. This model is characterized by proportionality, divisibility, additivity, and certainty. Generally, the mathematically formulation of LP is defined as [7, 2];

$$\max / \min z = c_1x_1 + c_2x_2 + c_3x_3 + \cdots + c_nx_n$$

subject to

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + \cdots + a_{1n}x_n \{\geq, \leq, =\} b_1 \\ a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + \cdots + a_{2n}x_n \{\geq, \leq, =\} b_2 \\ \vdots \\ a_{m1}x_1 + a_{m2}x_2 + a_{m3}x_3 + \cdots + a_{mn}x_n \{\geq, \leq, =\} b_m \\ x_1, x_2, x_3, \ldots, x_n \geq 0 \end{aligned} \tag{2.1}$$

where all the coefficients and variables are real valued.

In this formulation, $z = c_1x_1+c_2x_2+c_3x_3+\cdots+c_nx_n$ is the objective function or criterion function to be maximized; the coefficients $c_1, c_2, c_3, \ldots, c_n$ are cost coefficients; the variables $x_1, x_2, x_3, \ldots, x_n$ are decision variables to be determined; $a_{ij}, i = 1 : m, j = 1 : n$ are called technological coefficients and the equality or inequality expression denotes the restriction or constraint. The expression in (2.1) can be also expressed as;

$$\max / \min z = cx$$

$$Ax\{\leq, \geq, =\}b, x \geq 0 \tag{2.2}$$

where $x = [x_1, x_2, x_3, \ldots, x_n]^t, c = [c_1, c_2, c_3, \ldots, c_n]$, $b = [b_1, 2_2, b_3, \ldots, b_m]_t$and $A = [a_{ij}]_{m\times n}$ is $m \times n$ matrix.

Definition 2.1.1.

1. The set $S = \{x \in R^n : Ax\{\leq, \geq, =\}b, x \geq 0\}$ in problem (2.2) is called feasible region or feasible set.

2. Every element in S is known as feasible solution.

3. A point $x^* \in S$ is an optimal solution of problem (2.2) if $z^* = cx^* \geq cx, \forall x \in S$ for maximization and $z^* = cx^* \leq cx, \forall x \in$ for minimization.

Moreover, the standard form of linear programming problem (LPP) is expressed as;

$$\max z = cx$$
$$A^{'}x' = b^{'}, x \geq 0 \tag{2.3}$$

where c, $A^{'}$, $b^{'}$ and x' are appropriate matrices.

The optimal solution of LPP is always occur at the corner point of a feasible region. However, for the LPP with more than two decision variable it is not simple to search all the corner points and to identify the optimal point. Thus, such complexities can be overcome through algebraic method by converting the given LPP to standard form. All linear programming problems (LPP) can convert to standard LPP through expressing the objective function as maximization problem, transforming all constraints to equality constraints by introducing non-negative variables called slack or surplus variables with non-negative right hand side and handling the unrestricted and non-positive variables. Therefore, most of the popular techniques such as simplex method, Two-phase, Big-M and dual simplex methods of LPP are considered the standard form to simplify complexity of these problems.

However, for LPP with two decision variable can be solve using graphical method without changing the problem to standard form. In this method, we can find the optimal solution by sketching the feasible region and determining optimal value of the objective function at the corner points.

2.1.1 Simplex method

Linear programming problems can be studied both algebraically and geometrically. These two approaches are equivalent, but one or the other may be more convenient for answering a particular question about a linear programming. The algebraic point of view is based on writing the linear program in a particular way, called standard form and then the coefficient matrix of the constraints of the linear programming problem can be analyzed using the tools of linear algebra. The geometric point of view is based on the geometry of the feasible region and uses ideas such as convexity to analyze the linear programming problem. It is less dependent on the particular way in which the constraints are written. Using geometry (particularly in two-dimensional problems where the feasible region can be graphed) makes many of the concepts in linear programming easy to understand, because they can be described in terms of intuitive notions such as moving along an edge of the feasible region.

The simplex method computations are particularly tedious, repetitive, and, above all, boring. As you do these computations, you should not lose track of the big picture; namely, the simplex (or algebraic) method attempts to move from one corner point of the solution space to a better corner point until the optimum is found. In this method the LPP must be expressed in the standard form. The development of the simplex method computations is facilitated by imposing two requirements on the constraints of the problem:

- All the constraints (with the exception of the non-negativity of the variables) are equations with nonnegative right-hand side and
- All the variables are nonnegative.

These two requirements are imposed here primarily to standardize and streamline the simplex method calculations. It is important to know that all commercial packages directly accept inequality constraints, non-negative right-hand side, and unrestricted variables. Any necessary preconditioning of the model is done internally in the problem before the simplex method solves the problem.

Consider the standard minimization linear programming problem,

$$\max z = cx$$

subject to

$$Ax = b, b \geq 0 \tag{2.4}$$

$$x \geq 0$$

where the matrices are all with real components or entries and $c = [c_1, c_2, \ldots, c_n]$, $A = [a_{ij}]_{m \times n}$, $b = [b_1, b_2, \ldots, b_m]^t$ and $x = [x_1, x_2, \ldots, x_n]^t$.

Definition 2.1.2. *[3]* Consider the system $Ax = b, x \geq 0$ from problem (2.4) and suppose that $rank(A, b) = rank(A) = m$. After possibly rearranging the columns of A, let $A = [B, N]$ where B is $m \times m$ invertible matrix and N is an $m \times (n-m)$ matrix. The solution $x = (x_B, x_N)^t$ to the equations $Ax = b$, where $x_B = B^{-1}b$ and $x_N = 0$, is called a basic solution (BS) of the system. If $x_B \geq 0$, then then x is called a basic feasible solution (BFS) of the system.

From the above definition; B is called the basic matrix (or simply the basis) and N is called the nonbasic matrix. The components of x_B are called basic variables (dependent variables) and the components of x_N are called nonbasic variables (independent variables). If $x_B > 0$, then x is called a non-degenerate basic feasible solution, and if at least one component of x_B is zero, then x is called a degenerate basic feasible solution.

Therefore, problem (2.4) can be rewrite as;

$$\max z = c_B x_B + c_N x_N$$

subject to

$$[B, N][x_B, x_N]^t = Bx_B + Nx_N = b \tag{2.5}$$
$$x_B, x_N \geq 0$$

Definition 2.1.3. A feasible solution that achieves the maximum value of the objective function is said to be an optimal feasible solution. If it is also basic, then it is an optimal basic feasible solution.

Based on these definitions, the following fundamental theorems has been developed.

Theorem 2.1.1 (The Fundamental Thorem of LPP)**.** Given the linear programming problem in (2.4), where A is an $m \times n$ matrix of rank m:

1. If there is any feasible solution, then there is a basic feasible solution.

2. If there is any optimal solution, then there is a basic optimal solution.

Proof. Suppose that a feasible solution exists. Choose any feasible solution among those with the fewest non-zero components. If there are no nonzero components, then $x = 0$ and x is a basic solution by definition. Otherwise, take the index set $J = \{j_1, j_2, \ldots, j_r\}$ with elements corresponding to those $x_{j_i} > 0$. Then if we denote the corresponding columns by $\{a^{(j_1)}, a^{(j_2)}, \ldots, a^{(j_r)}\}$ there are two possibilities:

1. The set of columns is linearly independent. In this case, we certainly have $r \leq m$. If $r = m$ the corresponding solution is basic. If $r < m$ then, since A has rank m, we choose $m - r$ vectors from the remaining $n - r$ columns of A so that the resulting set of m column vectors is linearly independent. Assigning the value zero to the corresponding $m - r$ variables yields a (degenerate) basic feasible solution.

   The set of columns $\{a^{(j_1)}, a^{(j_2)}, \ldots, a^{(j_r)}\}$ is linearly dependent. Then there is a choice of scalars α_{j_i}, not all zero, such that

$$\alpha_{j_1} a^{(j_1)} + \alpha_{j_2} a^{(j_2)} + \cdots + \alpha_{j_r} a^{(j_r)} = 0. \tag{2.6}$$

   Without loss of generality, we may take $\alpha_{j_1} \neq 0$ and, indeed, $\alpha_{j_1} > 0$ (otherwise multiply (2.6) by -1).
   Now, since $x_{j_i} > 0$, the corresponding feasible solution x_J is just a linear combination of the columns $a^{(j_i)}$. Hence

$$Ax = \sum_{i=1}^{r} x_{j_i} a^{(j_i)} = b$$

   Next, multiplying the dependence relation (2.6) by a real number λ and subtracting, we have

$$A(x - \lambda\alpha) = \sum_{i=1}^{r} (x_{j_i} - \lambda\alpha_{j_i}) a^{(j_i)} = b,$$

   and this equation holds for every λ although one or more components, $x_k - \lambda\alpha_{j_k}$ may violate the non-negativity condition. Now let $\bar{\alpha} = (\bar{\alpha}_{j_1}, \ldots, \bar{\alpha}_{j_n})^t$ where $\bar{\alpha}_k = \alpha_{j_i}$ for $k = j_i$ and $\bar{\alpha}_k = 0$ for $k \neq j_i$.
   Then, for each value of λ the vector $x - \lambda\bar{\alpha}$ is a solution of the constraint equations. For the special value $\lambda = 0$, this vector is the original feasible solution. Now, as λ increases from 0, the components of $x - \lambda\bar{\alpha}$ will individually change; each will either increase, decrease, or stay constant, depending on the corresponding $\bar{\alpha}_i$ is positive, negative, or zero.
   Suppose, for some $i_o, \bar{\alpha}_{i_o} \leq 0$. Then $(x_{i_o} - \lambda\bar{\alpha}_{i_o}) \geq 0$ for all $\lambda > 0$. On the other hand, if for some $i_o, \bar{\alpha}_{i_o} > 0$, then $(x_{i_o} - \lambda\bar{\alpha}_{i_o})$ remains greater than 0 only for sufficiently small $\lambda > 0$. Now, we take

$$\bar{\lambda} = \min\{\frac{x_i}{\bar{\alpha}_i} : x_i > 0\}.$$

   Then $\bar{\lambda}$ corresponds to the first value of λ for which one or more non-zero components of $x - \bar{\lambda}\bar{\alpha}$ becomes 0. For this value of λ the vector

$x - \bar{\lambda}\bar{\alpha}$ is feasible and has at most $r-1$ positive components. Repeating this process if necessary, we can obtain a feasible solution with non-zero components corresponding to linearly independent columns.

2. Now assume that x^* is an optimal solution. There is no guarantee that this optimal solution is unique. Some of these solutions may have more positive components than others. Without loss of generality, we assume that x^* has a minimum number of positive components. If $x^* = 0$, then x^* is basic and the objective value is zero. If $x^* \neq 00$ and if J is the corresponding index set, then there are two cases as before. The proof of the first case in which the corresponding columns are linearly independent is exactly as in the previous proof of this case. In the second case, we proceed just as with the second case above. To do this, however, we must show that, for any λ, the vector $x^* - \lambda\bar{\alpha}$ is optimal. To show this, we note that the associated objective value is

$$c(x^* - \lambda\bar{\alpha}) = cx^* - \lambda(c\bar{\alpha})$$

Then, for sufficiently small $|\lambda|$, the vector $x^* - \lambda\bar{\alpha}$ is a feasible solution for positive or negative values of λ. Hence, we conclude that

$$c\bar{\alpha} = 0$$

Since, were it not, then we could determine a small λ of the proper sign, to make

$$c(x^* - \lambda\bar{\alpha}) < cx^*$$

which would violate the assumption of the optimality of x^*.
Having established that the new feasible solution with fewer positive components is also optimal, the remainder of the proof may be completed exactly as in case 2 above.

□

An optimal solution of the LP, if it exists, occurs at a basic feasible solution (BFS). Thus, the primal simplex method (or simply, the simplex method) is an iterative procedure which searches and identifies an optimal solution of the LP from among its BFSs. These ideas are summarized by the follows algorithm.

Generic Simplex Procedure to solve a LP problem:

1. Start with an initial basic feasible solution (BFS): choose the starting basic feasible solution with the corresponding basis.

2. Optimality Test: Is the current basic feasible (BFS) an optimal solution?

    - If it is YES, stop and the current BFS is optimal solution.
    - If No, move to adjacent BFS with better value; and repeat from step 2.

To initiate the simplex procedure, we have to reformulate the LP in its standard form. This requires that each component of b (the right-hand side/RHS) is non-negative; and exactly one basic variable with coefficient 1 occurs in each constraint equation. However, in many cases, a starting BFS is not readily available and some work may be needed to get the simplex method started. These complexity can be overcome using the most common two procedures (the two-phase method and Big-M method), both involving artificial variables to obtained an initial basic feasible solution to a slightly modified set of constraints. The simplex method is used to eliminate the artificial variables and to solve the original problem. In particular, the simplex method converges in a finite number of steps, even in the presence of degeneracy, provided that a special rule for exiting from basis is adopted.

Specifically, a LPP is said to be in perfect standard form (canonical form) if it is written as;

$$\max z = c_1x_1 + c_2x_2 + \cdots + c_nx_n + z_B$$

Subject to

$$a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n + x_{n+1} = b_1$$
$$a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n + x_{n+2} = b_2$$

$$\vdots$$

$$a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n + x_{n+m} = b_m$$
$$x_i \geq 0, \forall i = 1, 2, 3, \ldots, n \tag{2.7}$$

where $b_i \geq 0, \forall i = 1, 2, \ldots, m$.

In this case, the simplex method starts with the initial BFS $X_B = (x_{n+1}, x_{n+2}, \ldots, x_{n+m})^t = (b_1, b_2, \ldots, b_m)^t$ and $X_N = (x_1, x_2, \ldots, x_n)^t = (0, 0, \ldots, 0)^t$. The variables in X_B are called basic variable and the variables in X_N are called non-basic variables for the current iteration. The objective value at this BFS is z_B and the condition that $b_i \geq 0$ for all i, is called primal feasibility.

Now, setting the objective function $z = c_1x_1 + c_2x_2 + \cdots + c_nx_n + z_B$, LPP (2.7) can be reformulated as;

$$\max z$$

Subject to

$$-z + c_1x_1 + c_2x_2 + \cdots + c_nx_n = -z_B$$

$$a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n + x_{n+1} = b_1$$

$$a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n + x_{n+2} = b_2$$

$$\vdots$$

$$a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n + x_{n+m} = b_m$$

$$x_i \geq 0, \forall i = 1, 2, 3, \ldots, n \tag{2.8}$$

To make the simplex method is easy to operate, LPP in (2.8) can be presented in tabular form called simplex tableau as; Thus, the current BFS

		X_N				X_B				
		x_1	x_2	...	x_n	x_{n+1}	x_{n+2}	...	x_{n+m}	RHS
	$-z$	c_1	c_2	...	c_n	0	0	...	0	$-z_B$ ← Row 0 (objective row)
X_B	x_{n+1}	a_{11}	a_{12}	...	a_{1n}	1				b_1 ← Row 1
	x_{n+2}	a_{21}	a_{22}		a_{2n}		1			b_2
		⋮						⋱		
	x_{n+m}	a_{m1}	a_{m2}	...	a_{mn}				1	b_m ← Row m

Table 2.1: Solution using tabular simplex method

can be read off directly from the tableau and starting from this initial BFS and the current objective value $z = z_B$, the simplex procedure updates the tableau step-wise; i.e., moves to new BFSs by pivot operations, in such a way that the value of the objective function is monotonically increasing until an optimal solution is reached.

Therefore, the algorithm of simplex method is organized as;

1. Construct the simplex tableau of LPP in standard form and start with a BFS.

2. **Optimality Test:** If $\bar{c_j} \leq 0$, for all j, STOP. The current BFS is optimal solution, and the optimal objective value is $z = z_B$ (the entry at RHS of row 0). Otherwise, choose a $k \in \{1, 2, \ldots, n + m\}$ such

that $\bar{c_k} > 0$,that is, $\bar{c_k} = \max\{\bar{c_j} : j = 1 : n + m\}$. The column that contains $\bar{c_k}$ is called pivoting column and the corresponding variable is called entering variable.

3. **Unboundedness:** If $\bar{a_{ik}} \leq 0$ for all $i = 1, 2, \ldots, m$, then STOP and the LPP is unbounded,i.e., it has no finite optimal value. Otherwise, determine a pivot row r in which the following minimum occurs:

$$\min\{\frac{\bar{b_i}}{\bar{a_{ik}}} : \bar{a_{ik}} > 0, i = 1, 2, \ldots, m\}$$

and perform the pivot operation with pivoting at $\bar{a_{rk}}$. Thus, the r^{th} row is called pivoting row and the entry in the k^{th} column and r^{th} row $\bar{a_{ij}}$is called pivoting entry. Then, Repeat the procedure from 2 until the optimal solution obtained.

Example 2.1.1.

$$\max z = -x_1 - x_2 + 4x_3$$

subject to

$$x_1 + x_2 + 2x_3 \leq 9$$
$$x_1 + x_2 - x_3 \leq 2$$
$$-x_1 + x_2 + x_3 \leq 4$$
$$x_1, x_2, x_3 \geq 0$$

Solution: By introduce the non-negative slack variables x_4, x_5, and x_6, the given LPP can be convert to standard LP problem:

$$\max z = -x_1 - x_2 + 4x_3$$

subject to

$$x_1 + x_2 + 2x_3 + x_4 = 9$$
$$x_1 + x_2 - x_3 + x_5 = 2$$
$$-x_1 + x_2 + x_3 + x_6 = 4$$
$$x_1, x_2, x_3, x_4, x_5, x_6 \geq 0$$

Then, the staring BFS is $X_B = (x_4, x_5, x_6)^t = (9, 2, 4)^t$ and $X_N = (x_1, x_2, x_3)^t = (0, 0, 0)^t$. This initial BFS gives the following initial tableau:

In the tableau, $\bar{c_j} \leq 0$ for all j and so the optimal solution is given by $x_1 = \frac{1}{3}$, $x_2 = 0$ and $x_3 = \frac{13}{3}$ with optimal value $z = -17$. Therefore, the minimum value of an objective function z within the restriction is $z = -17$ which occurs at the corner point $(x_1, x_2, x_3)^t = (\frac{1}{3}, 0, \frac{13}{3})^t$.

Iteration 1

	z	x_1	x_2	x_3	x_4	x_5	x_6	RHS
z	1	−1	−1	4	0	0	0	0
x_4	0	1	1	2	1	0	0	9
x_5	0	1	1	−1	0	1	0	2
x_6	0	−1	1	(1)	0	0	1	4

Iteration 2

	z	x_1	x_2	x_3	x_4	x_5	x_6	RHS
z	1	3	−5	0	0	0	−4	−16
x_4	0	(3)	−1	0	1	0	−2	1
x_5	0	0	2	0	0	1	1	6
x_3	0	−1	1	1	0	0	1	4

Iteration 3

	z	x_1	x_2	x_3	x_4	x_5	x_6	RHS
z	1	0	−4	0	−1	0	−2	−17
x_1	0	1	−1/3	0	1/3	0	−2/3	1/3
x_5	0	0	2	0	0	1	1	6
x_3	0	0	2/3	1	1/3	0	1/3	13/3

Table 2.2: Solution using tabular simplex method

2.1.2 Dual Simplex method

For every linear programming problem, there is another associated linear programming problem which is called dual LPP that can be solve simultaneously. This new linear program satisfies some very important properties and it may be used to obtain the solution of the original programming problem with extremely useful information about the set of optimal solutions to the original linear programming problem.

We shall begin by formulating this new dual linear programming problem and proceed to develop some of its important properties. These properties will lead to the new algorithm, the dual simplex method, for solving linear

programming problems.

Formulation of the Dual Problems

To construct dual problem, the two important forms of duality; the canonical form of duality and the standard form of duality are important. These two forms are completely equivalent and arise respectively from the canonical and the standard representation of linear programming problems.

Suppose that the primal linear programming problem is given in the canonical form:

$$\min z = cx$$

subject to

$$Ax \geq b, x \geq 0 \tag{2.9}$$

Then the corresponding dual linear programming is defined as:

$$\max \bar{z} = wb$$

subject to

$$wA \leq c, w \geq 0 \tag{2.10}$$

Note that there is exactly one dual variable for each primal constraint and exactly one dual constraint for each primal variable.

Another equivalent definition of duality is given with the primal linear programming problem stated in the following standard form:

$$\min z = cx$$

subject to

$$Ax = b, x \geq 0 \tag{2.11}$$

Then the associated dual linear programming problem is defined by:

$$\max \bar{z} = wb$$

subject to

$$wA \leq c, w \;\; unrestricted \tag{2.12}$$

Dual Simplex method Algorithm

The dual simplex method solves the dual problem directly on the (primal) simplex tableau. At each iteration we move from a basic feasible solution of the dual problem to an improved basic feasible solution until optimality of the dual (and also the primal) is reached, or else, until we conclude that the dual is unbounded and that the primal is infeasible.

Consider the following linear programming problem:

$$\max z = cx$$

subject to

$$Ax = b, x \geq 0 \tag{2.13}$$

where $A = (y_{ij})_{m\times n}$, $c = (c_1, c_2, \ldots, c_n)$, $b = (b_1, b_2, \ldots, b_m)^t$ and $x = (x_1, x_2, \ldots, x_n)^t$.

In some linear programming problems, it is difficult to find a starting basic solution that is feasible ($\bar{b}_i \geq 0$) to a linear programming without adding artificial variables. In these instances it might be possible to find a starting basis, which is not necessarily feasible, but that is dual feasible ($\bar{c_j} = z_j - c_j \leq 0$ for a maximization problem). In such cases it is useful to develop a variant of the simplex method that would produce a series of simplex tableau that maintain dual feasibility and complementary slackness and strive toward primal feasibility.

Suppose that the tableau is dual feasible (that is, $\bar{c_j} = z_j - c_j \leq 0$ for a maximization problem). If the tableau is also primal feasible (that is, all $\bar{b}_i \geq 0$), then we have an optimal solution. Otherwise, consider some $b_r < 0$. By selecting row r as a pivot row and some column k such that $y_{rk} < 0$ as a pivot column, we can make the new right-hand-side $\bar{b}_r' > 0$. Through a series of such pivots we hope to make all $\bar{b}_i \geq 0$ while maintaining all $z_j - c_j \leq 0$, and thus achieve optimality.

The next part is how do we select the pivot column so as to maintain dual feasibility after pivoting. The pivot column k is determined by the following minimum ratio test:

$$\frac{z_k - c_k}{y_{rk}} = \min\{\frac{z_j - c_j}{y_{rj}} : y_{rj} < 0\}. \tag{2.14}$$

The above concepts are developed based on the tableau representation of problem (2.13) with a current basic solution at some iteration;

	z	x_1	$\cdots$	x_j	$\cdots$	x_k	$\cdots$	x_n	RHS
z	1	$z_1 - c_1$	$\cdots$	$z_j - c_j$	$\cdots$	$z_k - c_k$	$\cdots$	$z_n - c_n$	$\mathbf{c}_B \bar{\mathbf{b}}$
x_{B_1}	0	y_{11}	$\cdots$	y_{1j}	$\cdots$	y_{1k}	$\cdots$	y_{1n}	$\bar{b}_1$
x_{B_2}	0	y_{21}	$\cdots$	y_{2j}	$\cdots$	y_{2k}	$\cdots$	y_{2n}	$\bar{b}_2$
$\vdots$	$\vdots$	$\vdots$		$\vdots$	$\vdots$	$\vdots$		$\vdots$	$\vdots$
x_{B_r}	0	y_{r1}	$\cdots$	y_{rj}	$\cdots$	y_{rk}	$\cdots$	y_{rn}	$\bar{b}_r$
$\vdots$	$\vdots$	$\vdots$		$\vdots$		$\vdots$		$\vdots$	$\vdots$
x_{B_m}	0	y_{m1}	$\cdots$	y_{mj}	$\cdots$	y_{mk}	$\cdots$	y_{mn}	$\bar{b}_m$

Table 2.3: tableau representation of dual simplex method

A dual simplex method is applicable to the given problem if some or all of the right-hand sides of its standard form contains negative value, i.e., $\bar{b}_i < 0$ for some or all $i = 1, 2, \ldots, m$. Moreover, in simplex method, optimality depends on the values of $z_j - c_j$ but in dual simplex method its optimality depends on the value of $\bar{b}_i$. Generally, the Dual Simplex Algorithm (DSA) for maximized LPP is summarized as follows;

1. Find a starting basis of the primal such that $z_j - c_j \leq 0$ for all j of each column for basic variables.

2. If $\bar{b}_i \geq 0$ for all i, STOP; the current solution is optimal. Otherwise, select a pivot row r with $\bar{b}_r < 0$ using $\bar{b}_r = \min\{\bar{b}_i : \bar{b}_i < 0\}$.

3. If $y_{rj} \geq 0$ for all j , STOP; the dual is unbounded and the primal is infeasible. Otherwise, select the pivot column k by the following minimum ratio test:

$$\frac{z_k - c_k}{y_{rk}} = \min\{\frac{z_j - c_j}{y_{rj}} : y_{rj} < 0\}.$$

4. Pivot at y_{rk} and return to Step 2.

Example 2.1.2.

$$\min z = 2x_1 + 3x_2 + 4x_3$$

subject to

$$x_1 + 2x_2 + x_3 \geq 3$$
$$2x_1 - x_2 + 3x_3 \geq 4$$
$$x_1, x_2, x_3 \geq 0$$

Solution: A starting basic solution that is dual feasible can be obtained by utilizing the slack variables x_4 and x_5. Applying the dual simplex method, we obtain the following series of tableaux:

	z	x_1	x_2	x_3	x_4	x_5	RHS
z	1	−2	−3	−4	0	0	0
x_4	0	−1	−2	−1	1	0	−3
x_5	0	(−2)	1	−3	0	1	−4

	z	x_1	x_2	x_3	x_4	x_5	RHS
z	1	0	−4	−1	0	−1	4
x_4	0	0	(−5/2)	1/2	1	−1/2	−1
x_1	0	1	−1/2	3/2	0	−1/2	2

	z	x_1	x_2	x_3	x_4	x_5	RHS
z	1	0	0	−9/5	−8/5	−1/5	28/5
x_2	0	0	1	−1/5	−2/5	1/5	2/5
x_1	0	1	0	7/5	−1/5	−2/5	11/5

Table 2.4: Solution of an example using dual simplex method

Since $\bar{b} = (\bar{b}_1, \bar{b}_2)^t = (\frac{2}{5}, \frac{11}{5})^t \geq (0,0)^t$ and $z_j - c_j \leq 0$ for all j, an optimal pair of primal and dual solutions are at hand. Thus, optimal solution of the primal problem is $(x_1, x_2)^t = (\frac{2}{5}, \frac{11}{5})^t$.

The dual simplex algorithm is essential to solve different LP problem where a new constraint is eventually required to be included. This arises frequently within sensitivity (or post optimal) analysis of an LP problem and within various solution methods for integer Programming Problems such as the Branch & Bound method and the Gomory's Fractional Cutting Plane procedure.

2.2 Integer linear programming (ILP)

In many situations it does not make sense for decision variables to take values other than integers. Thus, in some real-life problems all or some of the decision variables are restricted to integer requirements and such type of problem is called integer linear programming (ILP). If all the decision variables are integer valued, the problem is know as pure integer linear programming (PILP). When some of the decision variables are integer values and some of them are continuous, the problem is called mixed integer linear programming (MILP).

2.2.1 Formulation of integer linear programming

Integer linear programming (ILP) investigates linear programming problems in which all or some of the variables are restricted to integers. ILP is classified in to two categories; pure ILP in which all decision variables are integer and mixed ILP if some of the variable are integers and some of them are continuous.
Mathematically, a pure integer linear programming (PILP) is defined as:

$$\max z = \sum_j c_j x_j$$

subject to

$$\sum_j a_{ij} x_j \leq b_i, (i = 1, 2, \ldots, m)$$
$$x_j \geq 0, x_j \in Z, (j = 1, 2, \ldots, n) \tag{2.15}$$

Moreover, a mixed integer linear programming (MILP) is expressed as:

$$\max z = \sum_j c_j x_j + \sum_k d_k y_k$$

subject to

$$\sum_j a_{ij}x_j + \sum_k g_{ik}y_k \leq b_i, (i = 1, 2, \ldots, m)$$

$$x_j \geq 0, (j = 1, 2, \ldots, n)$$

$$y_k = 0, 1, 2, \ldots, (k = 1, 2, \ldots, p) \tag{2.16}$$

Note that all input parameters are real constants with positive, negative, or zero values. Using matrix notation, a mixed integer program may be expressed as;

$$\max z = cx + dy$$

subject to

$$Ax + Gy \leq b$$

$$x \geq 0$$

$$y \geq 0, integer \tag{2.17}$$

where
$c = (c_1, c_2, \ldots, c_n)$ is a row vector of n elements,
$d = (d_1, d_2, \ldots, d_p)$ is a row vector of p elements,
$A = (a_{ij})m \times n$is an $m \times n$ matrix,$G = (g_{ik})_{m \times p}$ is an $m \times p$ matrix
$b = (b_1, b_2, \ldots, b_m)^t$ is a column vector of m constants
$x = (x_1, x_2, \ldots, x_n)^t$ is a column vector of n continuous variables
$y = (y_1, y_2, \ldots, y_p)^t$ is a column vector of p integer variables.
$m =$ number of constraints, $p =$ number of integer variables.

An integer linear programming in which the variables are restricted to be 0 or 1 is called a $0 - 1$ integer programming or binary ILP (BILP). A binary ILP with a single less than or equals to linear constraint, whose objective function and constraint coefficients are all positive, is called a knapsack (or backpack) problem. An ILP with a single constraint and all positive constraint coefficients is called an integer knapsack program, in which the values of an integer variable are not restricted to $0 - 1$.

A mixed integer programming problem is said to be in standard form if

- the objective function is maximized,
- all the constraints are of less than or equal to form,
- each integer variable is defined over consecutive integer numbers whose lower bound is 0 and upper bound infinity, and

- each continuous variable is non-negative with no finite upper bound.

Any MILP that does not conform to the above conditions is considered to be in nonstandard form, but may be converted to a standard one through simple mathematical manipulations.

Chapter 3

Mixed integer linear programming problem solution methods

Unlike linear programming problems, integer linear programming problems are very difficult to solve and so some efficient algorithms are developed to minimize these complexity. The most common exact methods used to solve integer linear programming problem are branch and bound, cutting plane method and dynamic programming methods. However, in this chapter we considered only the first two methods.

3.1 Cutting plane method

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 non-negative 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. The collection of all such cutting plane methods will be called a cutting plane approach.

3.1.1 Relaxation and optimality condition

Consider the MILP problem

$$\max z = cx + dy$$

subject to

$$Ax + Gy \leq b, x, y \geq 0, y \in Z \tag{3.1}$$

Then the set $S = \{(x, y) : Ax + Gy \leq b, x, y \geq 0, y \in Z\}$ represents the feasible set of a MILPP (3.1).

Definition 3.1.1. A problem $\max\{f(x') : x' \in S_R\}$ is called a relaxation of the MILPP if $S \subseteq S_R$ and $f(x') \geq cx + dy$ for all $x' = (x, y) \in S$.

Theorem 3.1.1. The linear programming relaxation of problem in (3.1) *is*

$$\max z^{'} = cx + dy$$

subject to

$$\begin{aligned} Ax + Gy &\leq b \\ x, y &\geq 0 \end{aligned} \tag{3.2}$$

Proof. Suppose that $S = \{(x, y) : Ax + Gy \leq b, x, y \geq 0, y \in Z\}$ and $S_R = \{(x, y) : Ax + Gy \leq b, x, y \geq 0\}$. Then, we need to show that $S \subseteq S_R$ and $z^{'} \geq z$ for all $(x, y) \in S$.
Now, let $\bar{x} = (x', y') \in S$. Then, $Ax' + Gy' \leq b$ and $x', y' \geq 0, y' \in Z$. Clearly this implies that $\bar{x} = (x', y') \in S_R$. Thus, $S \subseteq S_R$.In addition to this, $\bar{z} = cx' + dy' = z$ implies that $\bar{z} = cx' + dy' \geq z = cx + dy$ for all $(x, y) \in S$. □

Based on the above definition and theorem, the relaxation problem of the MILPP (3.1) is

$$\max z = cx + dy$$

subject to

$$\begin{aligned} Ax + Gy &\leq b \\ x, y &\geq 0 \end{aligned} \tag{3.3}$$

which also called linear programming relaxation problem (LPRP) of MILPP. Thus, the relaxation problem is obtained by ignoring the requirement y is integer.

Theorem 3.1.2. Let z_R be an optimal value of the relaxation problem (RP). Then, if $z_R = cx^* + dy^*$ for some $(x^*, y^*) \in S$, then (x^*, y^*) is an optimal solution of the MILPP in (3.1).

Proof. Suppose that z_R is an optimal value of the relaxation problem and $z_R = cx^* + dy^*$ for some $(x^*, y^*) \in S$. Then, $z_R = cx^* + dy^* \geq cx + dy$ for all $(x, y) \in S_R = \{(x, y) : Ax + Gy \leq b, x, y \geq 0\}$. Since $S \subseteq S_R$ and $(x^*, y^*) \in S$, $z_R = cx^* + dy^* \geq cx + dy$ for all $(x, y) \in S$. This implies that $z_R = cx^* + dy^*$ is optimal value of the MILPP in (3.1). Therefore, (x^*, y^*) is an optimal solution of the MILPPP in (3.1). □

Theorem 3.1.3. If the relaxation linear programming in (3.3) is infeasible, then also the MILPP in (3.1) is infeasible.

Proof. If the relaxation problem is infeasible, then $S_R = \emptyset$. Thus, since $S \subseteq S_R$, $S = \emptyset$. Hence, the MILPP is infeasible. □

Optimal solution of the linear programming relaxation (LPR) problem can be found using the Simplex Method. If an optimal solution of the LPR is integer, then it is an optimal solution of the MILPP, too. Otherwise, it needs some arrangement using the cutting plane method.

3.1.2 Cutting plane algorithm

The main idea in cutting plane method is to solve the integer linear programming problem (Pure or Mixed) by solving a sequence of linear programming problem. We first solve the linear programming relaxation and find an optimal solution. If the solution we obtained is in the feasible set, then it is also optimal solution of the integer linear programming problem. If not, we find an inequality that all integer solutions satisfy and add this inequality to the linear programming relaxation to obtain a new linear programming that contains all feasible solutions of the first problem but relatively more simple to solve than the original problem and we iterate this step until optimal solution obtained.

Generally, the cutting plane algorithm can be described as;

1. Solve the LPR and get its optimal solution $X^* = (x^*, y^*)$.
   If y^* is integer STOP with $X^* = (x^*, y^*)$ is optimal solution of the MILPP.

2. If y^* is not integer, then add new constraint (cutting plane) to the LPR such that

   - The feasible set of MILP remains the same;
   - The new constraint cut off(separates) X^* from the feasible set S;(that is,X^* does not satisfy the new constraint)

3. Repeat the procedure until the required condition satisfied.

3.1.3 Gomory's cutting plane algorithm

Gomory's Cutting plane methods for MILPP work by solving a non-integer linear program, i.e., the standard form of linear programming problem, the linear relaxation of the given integer program. Then, the current non-integer solution is not feasible to the MILPP. Thus, to update the non-integer solution using Gomory's cutting plane method, first we have to defined the following.

Definition 3.1.2. Suppose that $x \in R^n$ and $y \in Z^m, y \geq 0$ are decision variables in (3.1). Let $\alpha \in R^n$ and $\beta \in R$. Then

1. An inequality $\alpha^t X \leq \beta$ is called a valid inequality for the feasible set S in (3.1) if $\alpha^t X \leq \beta$ for all $X = (x, y) \in S$.

2. A valid inequality $\alpha^t X \leq \beta$ for S is said to be a cutting plane if it cuts(separates) some point $\bar{X}$ for which the variable y is non-integer from S such that $\bar{X} \in S_R$, but $\bar{X} \notin S$ (that is $\alpha^t \bar{X} > \beta$).

Next, the Gomory's cutting plane algorithm is developed by adding valid inequality to remove out some unnecessary part of its feasible set. Since all decision variables of pure ILPP are integer valued and only some of the decision variables are integers for MILPP, each of them have their corresponding valid inequalities. Because the valid inequality of PILPP may be remove out feasible solutions of the MILPP. These valid inequalities for Pure integer linear programming and mixed integer linear programming are developed separately based on the following fundamental theorems.

Theorem 3.1.4 (Gomory's Fractional Cut (GFC))**.** Let x^* be optimal solution of the LPR of (2.15). Suppose its i^{th} component $b_i \notin Z$ and the i^{th} constraint of the LPR of (2.15) at optimality is

$$y_{B_i} + \sum_{j \in N} a_{ij} y_j = b_i. \tag{3.4}$$

Then,

$$\sum_{j \in N} f_{ij} y_j \geq f_i \tag{3.5}$$

is a valid inequality and separates X^* from S, where $f_{ij} = a_{ij} - \lfloor a_{ij} \rfloor$, $f_i = b_i - \lfloor b_i \rfloor$, y_{B_j} is basic variables, $\lfloor b_i \rfloor$ is the greatest integer less than or equal to b_i and N is the index of nonbasic variable.

Proof. From the assumptions, for any $X = (x, y) \in S$,

$$y_i + \sum_{j \in N} a_{ij} y_j = b_i. \tag{3.6}$$

Then,

$$y_{B_i} + \sum_{j \in N} \lfloor a_{ij} \rfloor y_j - \lfloor b_i \rfloor = b_i - \lfloor b_i \rfloor - \sum_{j \in N} (a_{ij} - \lfloor a_{ij} \rfloor) y_j \tag{3.7}$$

Since $0 \leq b_i - \lfloor b_i \rfloor \leq 1$ and $0 \leq a_{ij} - \lfloor a_{ij} \rfloor \leq 1$, we have

$$b_i - \lfloor b_i \rfloor - \sum_{j \in N} (a_{ij} - \lfloor a_{ij} \rfloor) y_j \leq 0 \tag{3.8}$$

This implies that,

$$\sum_{j \in N} (a_{ij} - \lfloor a_{ij} \rfloor) y_j \geq b_i - \lfloor b_i \rfloor$$

Therefore,

$$\sum_{j \in N} f_{ij} y_j \geq f_i$$

where $a_{ij} - \lfloor a_{ij} \rfloor = f_{ij}$ and $b_i - \lfloor b_i \rfloor = f_i$. □

If we multiply the expression in (3.5) both sides by -1 and adding slack variable $x_s \geq 0$, the inequality in (3.5) is equivalent to

$$-\sum_{j \in N} f_{ij} y_j + x_s = -f_i \tag{3.9}$$

which is called Gomory's Fractional Cutting Plane (GFCP).

However, the above theorem (3.1.4) is appropriate for pure integer programming only and we can not apply for mixed integer programming as it is. So, the Gomory's cutting plane for mixed integer programming is developed based on the following theorem.

Theorem 3.1.5. Let $\bar{x} = (\bar{x}_1, \ldots, \bar{x}_{n+m}) = (x_1, \ldots, x_n, y_1, \ldots, y_m)$ be optimal solution of the LPR of (2.16). Suppose its i^{th} component $\bar{b}_i \notin Z$ and the i^{th} constraint of the LPR of (2.16) at optimality is

$$\bar{x}_{B_i} + \sum_{j \in N^+} \bar{a}_{ij} \bar{x}_j + \sum_{j \in N^-} \bar{a}_{ij} \bar{x}_j = \bar{b}_i. \tag{3.10}$$

Then,

$$\sum_{j \in N^+} \bar{a}_{ij} \bar{x}_j + \frac{f_i}{f_i - 1} \sum_{j \in N^-} \bar{a}_{ij} \bar{x}_j \geq f_i \tag{3.11}$$

is the Gomory's cut in mixed integer linear programming, where $f_i = \bar{b}_i - \lfloor \bar{b}_i \rfloor$, $\bar{x_j}$'s are decision variable (both continuous and integer), $\bar{x}_{B_i}$s' are any basic decision variables (continuous or integer) and $N^+ = \{j : \bar{a}_{ij} \geq 0\}$, $N^- = \{j : \bar{a}_{ij} < 0\}$.

Proof. From the right hand side of (3.10), we have

$$\bar{b}_i = \lfloor \bar{b}_i \rfloor + \bar{b}_i - \lfloor \bar{b}_i \rfloor.$$

Then, $\bar{b}_i = \lfloor \bar{b}_i \rfloor + f_i$ and $0 < f_i < 1$ if $f_i = \bar{b}_i - \lfloor \bar{b}_i \rfloor$.
Next, relation (3.10) can be written as

$$\sum_{j \in N^+} \bar{a}_{ij}\bar{x}_j + \sum_{j \in N^-} \bar{a}_{ij}\bar{x}_j = f_i + \lfloor \bar{b}_i \rfloor - \bar{x}_{B_i}. \tag{3.12}$$

The right hand side of relation (3.12) is either ≥ 0 or < 0.
Case-1: Let $f_i + \lfloor \bar{b}_i \rfloor - \bar{x}_{B_i} \geq 0$. Since $\lfloor \bar{b}_i \rfloor$ and $\bar{x}_{B_i}$ are integers, $\lfloor \bar{b}_i \rfloor - \bar{x}_{B_i} = 0, 1, 2, \ldots$. Hence, relation (3.12) becomes

$$\sum_{j \in N^+} \bar{a}_{ij}\bar{x}_j + \sum_{j \in N^-} \bar{a}_{ij}\bar{x}_j \geq f_i. \tag{3.13}$$

Since $\bar{x}_j \geq 0$ and $\bar{a}_{ij} \leq 0$ for $j \in N^-$,

$$\sum_{j \in N^+} \bar{a}_{ij}\bar{x}_j \geq \sum_{j \in N^+} \bar{a}_{ij}\bar{x}_j + \sum_{j \in N^-} \bar{a}_{ij}\bar{x}_j. \tag{3.14}$$

The relations (3.13) and (3.14), yields

$$\sum_{j \in N^+} \bar{a}_{ij}\bar{x}_j \geq f_i \tag{3.15}$$

Case-2: Let $f_i + \lfloor \bar{b}_i \rfloor - \bar{x}_{B_i} < 0$. this implies that $\lfloor \bar{b}_i \rfloor - \bar{x}_{B_i} = -1, -2, \ldots$.
Hence, relation (3.12) gives

$$\sum_{j \in N^+} \bar{a}_{ij}\bar{x}_j + \sum_{j \in N^-} \bar{a}_{ij}\bar{x}_j \leq f_i - 1 \tag{3.16}$$

Clearly

$$\sum_{j \in N^-} \bar{a}_{ij}\bar{x}_j \leq \sum_{j \in N^+} \bar{a}_{ij}\bar{x}_j + \sum_{j \in N^-} \bar{a}_{ij}\bar{x}_j. \tag{3.17}$$

From the above two relations (3.16) and (3.17), we have

$$\sum_{j \in N^-} \bar{a}_{ij}\bar{x}_j \leq f_i - 1.$$

Since $0 < f_i < 1$, the above gives

$$\frac{f_i}{f_i - 1} \sum_{j \in N^-} \bar{a}_{ij} \bar{x}_j \geq f_i. \tag{3.18}$$

Finally, by adding the two relations (3.15) and (3.18), we get

$$\sum_{j \in N^+} \bar{a}_{ij} \bar{x}_j + \frac{f_i}{f_i - 1} \sum_{j \in N^-} \bar{a}_{ij} \bar{x}_j \geq 2f_i \geq f_i. \tag{3.19}$$

Since $\bar{x}_j$ is arbitrary component variable of $\bar{x} = (x, y)$,

$$-\sum_{j \in N^+} \bar{a}_{ij} \bar{x}_j - \frac{f_i}{f_i - 1} \sum_{j \in N^-} \bar{a}_{ij} \bar{x}_j \leq -f_i, \forall \bar{x}_j. \tag{3.20}$$

This implies that

$$\sum_{j \in N^+} \bar{a}_{ij} \bar{x}_j + \frac{f_i}{f_i - 1} \sum_{j \in N^-} \bar{a}_{ij} \bar{x}_j \geq f_i \tag{3.21}$$

is valid inequality.
Let

$$S_C = \{\bar{x} = (\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_{n+m}) : \bar{x} \in S, \sum_{j \in N^+} \bar{a}_{ij} \bar{x}_j + \frac{f_i}{f_i - 1} \sum_{j \in N^-} \bar{a}_{ij} \bar{x}_j \geq f_i\}$$

and

$$S_{CO} = \{\bar{x} = (\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_{n+m}) : \bar{x} \in S, \sum_{j \in N^+} \bar{a}_{ij} \bar{x}_j + \frac{f_i}{f_i - 1} \sum_{j \in N^-} \bar{a}_{ij} \bar{x}_j < f_i\}.$$

Then, we need to prove that $S_C \cap S_{CO} = \emptyset$.
Now, suppose that $\bar{x} \in S_C \cap S_{CO}$. This implies that

$$\sum_{j \in N^+} \bar{a}_{ij} \bar{x}_j + \frac{f_i}{f_i - 1} \sum_{j \in N^-} \bar{a}_{ij} \bar{x}_j \geq f_i, \forall i = 1, 2, \ldots, n + m \tag{3.22}$$

and

$$\sum_{j \in N^+} \bar{a}_{ij} \bar{x}_j + \frac{f_i}{f_i - 1} \sum_{j \in N^-} \bar{a}_{ij} \bar{x}_j < f_i, \forall i = 1, 2, \ldots, n + m \tag{3.23}$$

Consequently, we get

$$f_i \leq \sum_{j \in N^+} \bar{a}_{ij}\bar{x}_j + \frac{f_i}{f_i - 1} \sum_{j \in N^-} \bar{a}_{ij}\bar{x}_j < f_i, \forall i = 1, 2, \ldots, n + m$$

But it is impossible. Hence, $S_C \cap S_{CO} = \emptyset$ and so that there is no $\bar{x} = (x, y) \in S$ such that y is integer and x is continuous. Therefore, the Gomory's cut in mixed ILP is

$$\sum_{j \in N^+} \bar{a}_{ij}\bar{x}_j + \frac{f_i}{f_i - 1} \sum_{j \in N^-} \bar{a}_{ij}\bar{x}_j \geq f_i. \tag{3.24}$$

□

From theorem (3.1.5), the Gomory's Fractional cutting plane for MILPP is;

$$-\sum_{j \in N^+} \bar{a}_{ij}\bar{x}_j - \frac{f_i}{f_i - 1} \sum_{j \in N^-} \bar{a}_{ij}\bar{x}_j + x_s = -f_i. \tag{3.25}$$

for non-negative slake variable x_s.
Generally, Gomory's cutting plane algorithm is summarized as;

1. Solve the LPR in (3.3) using simplex algorithm to get an optimal solution $X^* = (x^*, y^*)$.
   - If y^* is integer, then STOP with $X^* = (x^*, y^*)$ is an optimal solution of the MILPP in(3.1).
   - Otherwise, go to step 2.
2. Choose an i^{th} row of the optimal simplex tableau, where the right-hand-side $\bar{b}_i \notin Z$, i.e., y_i^* is not integer and go to step 3.
3. Generate the GFCP of MILPP with respect to the selected row in step 2;
   $$-\sum_{j \in N^+} \bar{a}_{ij}\bar{x}_j - \frac{f_i}{f_i - 1} \sum_{j \in N^-} \bar{a}_{ij}\bar{x}_j + x_s = -f_i \tag{3.26}$$
   and include this in the final simplex tableau and go to step 4.
4. Re-solve the new tableau in step 3 using dual simplex method (Since the right-hand side of new constraint is negative, we can not solve the new problem using simplex method).
   - If the solution obtained by dual simplex method is $(\bar{x}, \bar{y})$ with $y \in Z^m$, STOP with $\bar{X} = (\bar{x}, \bar{y})$ is optimal of the MILPP in (3.1).

- Otherwise go back to step 2.

Example 3.1.1.

$$\max z = 5x + 6y$$

subject to

$$10x + 3y \leq 52$$

$$2x + 3y \leq 18$$

$$x, y \geq 0 \& x \in Z$$

Solution: First, the relaxation problem of the MILPP is;

$$\max z = 5x + 6y$$

subject to

$$10x + 3y \leq 52$$

$$2x + 3y \leq 18$$

$$x, y \geq 0$$

Thus, the standard form of the relaxation problem can be obtained by introducing the slack variables x_3 and x_4;

$$\max z = 5x + 6y$$

subject to

$$10x + 3y + x_3 = 52$$

$$2x + 3y + x_4 = 18$$

$$x, y, x_3, x_4 \geq 0$$

Next, we can find the optimal solution based on the Gomory's cutting plane algorithm as follows;
Step-1: Solve the standard form using simplex tableau algorithm as:
Iteration-1:

	x	y	x_3	x_4	RHS
x_3	10	3	1	0	52
x_4	2	3	0	1	18
z	5	6	0	0	0

Then, the entry in the second row and second column is a pivoting entry. Based on this entry, the table can be updated as follows;
Iteration-2:

	x	y	x_3	x_4	RHS
x_3	8	0	1	-1	34
y	$\frac{2}{3}$	1	0	$\frac{1}{3}$	6
z	1	0	0	-2	-36

Again, this table is not optimal and so we have to updated;
Iteration-3:

	x	y	x_3	x_4	RHS
x	1	0	$\frac{1}{8}$	$\frac{-1}{8}$	$\frac{17}{4}$
y	0	1	$\frac{-1}{12}$	$\frac{5}{12}$	$\frac{19}{6}$
z	0	0	$\frac{-1}{8}$	$\frac{-15}{8}$	$\frac{-161}{4}$

Since all values of z are less than or equal to zero, the table is optimal and $(x^*, y^*) = (\frac{17}{4}, \frac{19}{6})$ is optimal solution of the linear programming relaxation but not optimal solution of the mixed integer programming. Thus, to find the required solution, go to step-2.
Step-2: Since the value of x in the first row is not integer, select this row to develop a Gomory's cutting plane and go to step-3.
Step-3: The GFCP can be generated as follows; From the table $N^+ = \{3\}$, and $N^- = \{4\}$. Since $f_1 = \frac{17}{4} - \lfloor \frac{17}{4} \rfloor = \frac{1}{4}$, the Gomory's Fractional cutting plane is

$$\frac{1}{8}x_3 + \frac{\frac{1}{4}}{\frac{1}{4} - 1}\frac{-1}{8}x_4 \geq \frac{1}{4}$$

That is, $-\frac{1}{8}x_3 - \frac{1}{24} + x_5 = -\frac{1}{4}$ and by including this new constraint in the above optimal table, we get

	x	y	x_3	x_4	x_5	RHS
x	1	0	$\frac{1}{8}$	$\frac{-1}{8}$	0	$\frac{17}{4}$
y	0	1	$\frac{-1}{12}$	$\frac{5}{12}$	0	$\frac{19}{6}$
x_5	0	0	$\frac{-1}{8}$	$\frac{-1}{24}$	1	$\frac{-1}{4}$
z	0	0	$\frac{-1}{8}$	$\frac{-15}{8}$	0	$\frac{-161}{4}$

Step-4: Solve the table in step-3 using dual simplex algorithm as follows;By simple observation from the table, the entry in the third row and third column is pivoting entry. Then, using the dual simplex algorithm, we have the updated tableau

	x	y	x_3	x_4	x_5	RHS
x	1	0	0	$\frac{-1}{6}$	1	4
y	0	1	0	$\frac{4}{9}$	$\frac{-2}{3}$	$\frac{10}{3}$
x_3	0	0	1	$\frac{1}{3}$	-8	2
z	0	0	0	$\frac{-11}{6}$	-1	-40

Since all values of z are less than or equal to zero, the table is optimal. Therefore, an optimal solution of the original MILPP is $(x^*, y^*) = (4, \frac{10}{3})$.

Example 3.1.2. Solve the following MILPP using Gomory' cutting plane method.

$$\max z = 7x_1 + 9x_2$$

subject to

$$-x_1 + 3x_2 \leq 3$$

$$14x_1 + 2x_2 \leq 35$$

$$x_1, x_2 \geq 0 \& x_1 \in Z$$

Solution: The relaxation problem is

$$\max z = 7x_1 + 9x_2$$

subject to

$$-x_1 + 3x_2 \leq 3$$

$$14x_1 + 2x_2 \leq 35$$

$$x_1, x_2 \geq 0$$

Then using simplex method, optimal tableau of the relaxation problem is

	x_1	x_2	x_3	x_4	RHS
x_2	0	1	$\frac{7}{11}$	$\frac{1}{44}$	$\frac{7}{4}$
x_1	1	0	$\frac{-1}{22}$	$\frac{3}{4}$	$\frac{9}{4}$
z	0	0	$\frac{-28}{7}$	$\frac{-15}{22}$	$\frac{63}{2}$

with optimal solution $x^* = (x_1, x_2) = (\frac{9}{4}, \frac{7}{4})$. But, $x_1 = \frac{9}{4}$ is not integer and so that it can not be optimal solution for the MILPP.
Next, select the 2^{nd} row in which x_1 is non-integer, ie., $\bar{b_2} = \frac{9}{4} \notin Z$. Then, $f_2 = \lfloor \frac{9}{4} - \frac{9}{4} = \frac{1}{4}$ and the GFCP with respect the second row is,

$$-\frac{3}{4}x_4 - \frac{\frac{1}{4}}{\frac{1}{4} - 1}(\frac{-1}{22}x_3) + x_5 = -\frac{1}{4} \tag{3.27}$$

That is,

$$-\frac{3}{4}x_4 - \frac{1}{66}x_3 + x_5 = -\frac{1}{4} \tag{3.28}$$

By introducing the GFCP to the simplex optimal tableau, we get

	x_1	x_2	x_3	x_4	x_5	RHS
x_2	0	1	$\frac{7}{11}$	$\frac{1}{44}$	0	$\frac{7}{4}$
x_1	1	0	$\frac{-1}{22}$	$\frac{3}{4}$	0	$\frac{9}{4}$
x_5	0	0	$\frac{-1}{66}$	$\frac{-3}{4}$	1	$\frac{-1}{4}$
z	0	0	$\frac{-28}{7}$	$\frac{-15}{22}$	0	$\frac{63}{2}$

If we solve this new tableau using dual simplex method, then wet the solution $x^* = (x_1, x_2) = (2, \frac{5}{3})$. Since $x_1, x_2 = 2 \in Z$, it is optimal solution of the MILPPP.

3.2 Branch and Bound method

3.2.1 Introduction

In cutting plane method, the optimal solution can be obtained simply by removing part of the feasible set of its relaxation problem. But still the new formulated MILPP by adding cutting plane to its solution is not that much simple, because the region we removed is not stable. Therefore, cutting plane method is fast but unreliable. Due to these complexity of the cutting plane method, researchers developed another stable method called branch & bound method which is relatively slow.

Branch-and-bound is a general-purpose approach capable of solving mixed ILP problems. Branch and Bound method uses a "divide and conquer" approach to explore the set of feasible integer solutions. Moreover, Branch-and-bound can be viewed as a divide-and-conquer approach to solving the MILP problem, in which a branching process for dividing and a bounding process for conquering. Throughout the algorithm, a series of LP sub-problems are systematically generated and solved. Then upper and lower bounds are progressively limited on the objective value of the original MILP problem. However, instead of exploring the entire feasible set, it uses bounds on the optimal cost to avoid exploring certain parts of the set of feasible integer solutions.

A process of branch and bound method is represented by a branch-and-bound tree, which is a specialized enumeration tree for keeping track of how LP sub-problems are generated and solved. The branch and bound 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 MILP problem is solved. If the LP optimum solution satisfies the integer requirement, the MILP problem is solved. Otherwise, the LP objective value becomes the initial upper

bound on the MILP optimal objective value and the root node is partitioned into two sub-problems by two branches.

Dividing (branching) is done by partitioning the feasible set S of problem in (3.1) into smaller and smaller subsets $S_1, S_2, \ldots, S_r$ such that $S = S_1 \cup S_2 \cup \cdots \cup S_r$ and each sub-problems have lower and upper bounds. The branch and bound tree representation of the branched problem is presented in figure (3.1). Each sub-problem has a lower bound L^* most of the time

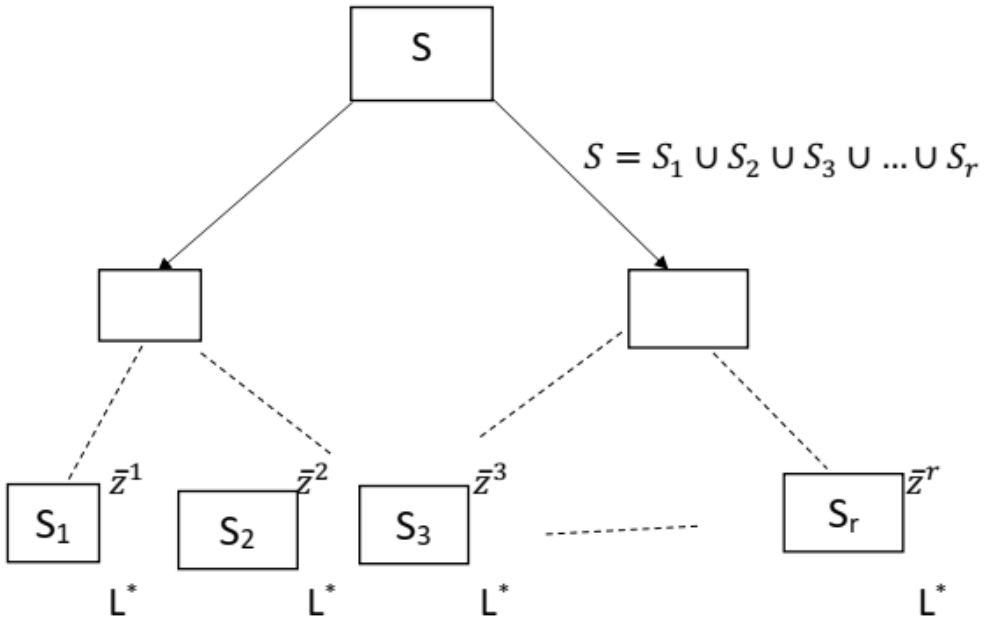

Figure 3.1: Branch and bound tree representation

$L^* = -\infty$ in the initial bounding and upper bound $\bar{z}^k, k = 1 : r$ which can be obtained by solving its LPR before further dividing it into two. Therefore, at every stage of the branch & bound $L^* \leq \bar{z}^* \leq \bar{z}^k$, where $\bar{z}^*$ is optimal value of the given MILPP in (3.1).

To describe branching mathematically, select a component $\bar{y}_i$ of $y = (y_1, \ldots, y_m)$ in problem(3.1) which has non-integer value $\bar{b}_i$. Then, the variable $\bar{y}_i$ is called branching variable and the valid inequalities that branch the original MILPP in two sub-problems are;

$$\bar{y}_i \leq \lfloor \bar{y}_i \rfloor \tag{3.29}$$

and

$$\bar{y}_i \geq \lfloor \bar{y}_i \rfloor + 1. \tag{3.30}$$

Hence, the feasible set S of the original MILPP can be divided in to two mutually exclusive sets

$$S_1 = S \cap \{(x, y) \in S : \bar{y_i} \leq \lfloor \bar{y_i} \rfloor\} \tag{3.31}$$

and

$$S_2 = S \cap \{(x, y) \in S : \bar{y_i} \geq \lfloor \bar{y_i} \rfloor + 1\}. \tag{3.32}$$

Consequently, from the above two sets in (3.31) and (3.32) we have $S = S_1 \cup S_2$ and $S_1 \cap S_2 = \emptyset$.

The solution of an LP relaxation on a node provides information about properties;

1. whether a further branching from this node is needed (or whether the node can be pruned), and
2. a better lower bound (for maximization problem) on the objective of the original MILP problem.

There are three cases indicating that a node can be pruned:

- the sub-problem has no feasible LP solution,
- the sub-problem has optimum solution that satisfy the integral values, and
- the upper bound of the sub-problem 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 MILP problem.

Whenever a solution of sub-problem obtained satisfy the integer condition , it is a candidate optimum to the original MILP problem. In the solution process of branch and bound, the best integer solution found so far is continuously updated. This feasible solution with best objective value at any stage is called incumbent solution.

When a 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:

- Variable with fractional value closest to 0.5
- Variable with highest impact on objective function
- 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;

- Depth-first (last-in first-out; solve the most recently generated sub-problem first)
- 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 is a lower bound on the given MILP problem and can be used to prune nodes by optimality. 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 branch and bound tree. Performance of these branching rules depends on the problem structure. In practice, apply the depth-first strategy first to get one feasible integer solution, followed by a mixture of both strategies.

3.2.2 Branch and Bound Algorithm

Now we describe the general branch and bound algorithm using the following notation from the MILPP in (3.1).
Assume that S = the given MILP problem, S_{LP} = the LPR of the given MILP S, y_{LP} = the solution to the LP relaxation of the given MILP, $\bar{z}$ = lowest (best) upper bound on $z*$ of the given MILP problem, $\hat{z}$ = highest (best) lower bound on $z*$ of the given MILP problem. These are global bounds that are periodically updated as the branching proceeds down the various paths in the tree.
Moreover, the following also additional assumptions to describe the branch and bound algorithm.
S_k = the sub-problem k of problem S,
S_{LP}^k = the LP relaxation of subproblem k,
z^k = the optimum objective value of problem with feasible set S_k, $\bar{z}^k$ =best (lowest) upper bound of subproblem with set S_k (shown above node k)
$\hat{z}^k$ = best (highest) lower bound of subproblem with S_k (shown below node k)

y_{LP}^k = the optimum solution of the LP subproblem with S_{LP}^k, $\bar{y}_j$ = non-integer value of integer variable y_j (current numerical value of y_j)

Now formally we describe the branch and bound algorithm as;

1. **(Initialization)**: Solve the linear programming relaxation (S_{LP}) of the given MILP problem. If it is infeasible, so is the MILP problem and then terminate. If the LP optimum solution satisfies the integer requirement, the MILP problem is solved and then terminate. Otherwise, initialize the best upper bound ($\bar{z}$) by the optimal objective value of problem S_{LP} and the best lower bound by $\hat{z} = -\infty$. Place the problem S_{LP}^k on the active list of nodes (sub-problems). Initially, there is no incumbent solution.

2. **(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 (depth-first or best-bound-first).

3. **(Updating Upper Bound)**: Solve and set z_k equal to the LP optimum objective value. Keep the optimum LP solution y_{LP}^k.

4. **(Prune by Infeasibility)**: If S_{LP}^k has no feasible solution, prune the current node and go to step 2. Otherwise, go to step 5.

5. **(Prune by Bound)**: If $z^k \leq \hat{z}$, prune the current node and go to step 2. Otherwise, go to step 6.

6. **(Updating Lower Bound and Prune by Optimality).**

   - If the LP optimum y_{LP}^k is integer, a feasible solution to S is found, an incumbent solution to the given problem. Set $\hat{y}^* = y_{LP}^k$ and compare $\hat{z}^k$ with $\hat{z}$. If $\hat{z}^k > \hat{z}$, set $\hat{z} = \hat{z}^k$, otherwise $\hat{z}$ does not change. The current node is pruned because no better solution can be branched down from this node. Go to step 2.
   - If the LP optimum y_{LP}^k is non-integer, go to step 7.

7. **(Branching)**: From the current node S^k choose a variable y_j with fractional value to generate two sub-problems S_1^k and S_2^k defined by

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

$$S_2^k = S^k \cap \{y : y_j \geq \lfloor \bar{y}_j \rfloor + 1\}$$

   Place both of these two nodes in the active list and go to step 2.

Example 3.2.1.

$$\max z = 7x_1 + 9x_2$$

Subject to

$$-x_1 + 3x_2 \leq 3$$

$$14x_1 + 2x_2 \leq 35$$

$$x_1, x_2 \geq 0, x_1 \in Z$$

Solution: By introducing slak variables x_3, x_4 and using the primal simplex method, the optimal solution of the linear programming relaxation of the given MILPP is obtained as follows;

	x_1	x_2	x_3	x_4	RHS
x_3	-1	3	1	0	3
x_4	14	2	0	1	35
z	7	9	0	0	0

From the above table, the entry in the first row and second column is a pivoting entry and x_2 is entering, x_3 is leaving variables. Then, the updated table becomes;

	x_1	x_2	x_3	x_4	RHS
x_2	$\frac{-1}{3}$	1	$\frac{1}{3}$	0	1
x_4	$\frac{44}{3}$	0	$\frac{-2}{3}$	1	33
z	10	0	-3	0	-9

Again in this table, the entry in the second row and first column is a pivoting entry. So, the updated table becomes;

	x_1	x_2	x_3	x_4	RHS
x_2	0	1	$\frac{7}{11}$	$\frac{1}{44}$	$\frac{7}{4}$
x_1	1	0	$\frac{-1}{22}$	$\frac{3}{4}$	$\frac{9}{4}$
z	0	0	$\frac{-28}{7}$	$\frac{-15}{22}$	$\frac{63}{2}$

Since all values of z in the last row of the last table are ≤ 0, the table is optimal and an optimal solution of the linear programming relaxation is $x^* = (\frac{9}{4}, \frac{7}{4})$ with $z^* = \frac{63}{2}$. But, the value of $x_1^* = \frac{9}{4}$ is not integer. So that, x^* is not optimal solution of the MILPP.
Choosing x_1 to be the branching variable set: $x_1 \leq \lfloor \frac{9}{4} \rfloor$ and $x_1 \geq \lfloor \frac{9}{4} \rfloor + 1$ as new constraints to be include in the new problem,

$$S_1 = \max\{z = 7x_1 + 9x_2 : (x_1, x_2) \in S, x_1 \leq 2\}$$

$$S_2 = \max\{z = 7x_1 + 9x_2 : (x_1, x_2) \in S, x_1 \geq 3\}$$

So, $LIST = \{S_1, S_2\}$.
Now, taking S_1 from $LIST$ and solving its LPR using simplex method we get: $x^1_{LP} = (2, \frac{5}{3})$ with $\bar{z}^1 = 29$.
Since $\bar{z}^1 = 29 \leq z^* = \frac{63}{2}$, prune the problem S_1 from $LIST$. So that, $LIST = \{S_2\}$.
Next, solve S_2 using dual simplex method to obtain its optimal: the problem is infeasible. Then, prune problem S_2 from $LIST$ and so $LIST = \emptyset$. Therefore, optimal solution of the given mixed integer programming is $x^* = (x_1, x_2) = (2, \frac{5}{3})$.

Example 3.2.2. Using the branch and bound method find optimal solution of

$$\max z = 5x_1 + 6x_2$$

subject to

$$10x_1 + 3x_2 \leq 52$$

$$2x_1 + 3x_2 \leq 18$$

$$x_1, x_2 \geq 0 \& x_1 \in Z$$

Solution: Using simplex method as (3.1.1), the optimal tableau of the linear programming relaxation (S_{LP}) is

	x_1	x_2	x_3	x_4	RHS
x_1	1	0	$\frac{1}{8}$	$\frac{-1}{8}$	$\frac{17}{4}$
x_2	0	1	$\frac{-1}{12}$	$\frac{5}{12}$	$\frac{19}{6}$
-z	0	0	$\frac{-1}{8}$	$\frac{-15}{8}$	$\frac{-161}{4}$

Then, the optimal solution of the relaxation problem is $x^* = (x_1, x_2) = (\frac{17}{4}, \frac{19}{6})$ with optimal value $\bar{z} = \frac{161}{4}$.
Since $\frac{17}{4}$ is not integer, $x^* = (x_1, x_2) = (\frac{17}{4}, \frac{19}{6})$ is not optimal solution of the original MILPP (S). Then, we have a branching constraints corresponding to x_1; $x_1 \leq \lfloor \frac{17}{4} \rfloor = 4$ and $x_1 \geq \lfloor \frac{17}{4} \rfloor + 1 = 5$. Based on these new constraints, we form two new problems

$$S_1 = \{\max z = 5x_1 + 6x_2 : x \in S, x_1 \leq 4\}$$

$$S_2 = \{\max z = 5x_1 + 6x_2 : x \in S, x_1 \geq 5$$

Thus, $LIST = \{S_1, S_2\}$.

- Using dual simplex method, if we solve problem S_1 , we obtained an optimal solution $x^* = (x_1, x_2) = (4, \frac{10}{3})$ with $\hat{z}^1 = 40$. Since $\hat{z}^1 = 40 \leq \bar{z}^1 = \frac{161}{4}$,

prune the current node S_1 from $LIST$. Then, $LIST = \{S_2\}$.

Next, solve problem S_2 in similar procedure and we get $x^* = (x_1, x_2) = (5, \frac{2}{3})$ with $\bar{z}^2 = 29$. Since $\bar{z}^2 = 29 \leq \bar{z}^1 = 40$ and $LIST = \emptyset$, $x^* = (x_1, x_2) = (4, \frac{10}{3})$ is an optimal solution of the original MILPP.

Conclusion

Mixed integer linear programming problem is a branch of integer programming in which its solution can be found using different techniques such as branch and bound, cutting plane methods. MILPP is more complex than linear programming problem. Therefore, these complexities can be minimized by these two methods. In branch and bound method the optimal solution of MILPP is obtained by dividing it in to smaller sub-problems to have more simple problems. Moreover, in cutting plane method optimal solution of MILPP is obtained by removing the unnecessary part of a feasible region to have some what simple MILPP. However, as the number of variables and number of constraints increase the complexity increase, in some cases it may not be manageable to get its optimal solution. In addition to this, the number of required branching and cutting plane can be large (particularly for large scale MILP problems) and due to the simplex algorithm solves the LPR of sub-problems is not polynomial time algorithm.

However, by combining these methods, a faster and reliable method is developed which is called Branch & cut. This method can be used to solve MILPP by using cutting planes in addition to linear programming relaxation.

Bibliography

[1] A. Bachem and W. Kern, Linear Programming Duality:An Introduction to Oriented Matroids., Springer-Verlag, 1991.

[2] M.S. Bazaraa and J.J. Jarvis, Linear Programming and Network Flows .,John Wiley and Sons, Inc.,1977.

[3] M.S. Bazaraa, J.J. Jarvis and H. D. Sherali, Linear Programming and Network Flows .,John Wiley and Sons, Inc.,Fourth Edition, 2009.

[4] D. Berthimas and J.N. Thitsiklis, Introduction to linear optimization, Massachusetts Institute of Technology, Second Edition, 1999.

[5] D. Chen, R.G. Batson and Y. Dang, Applied integer programming:modeling and Solution , John Wiley and Sons Inc., 2010.

[6] F. S. Hillier and G. J. Lieberman, Introduction to operations research., McGraw-Hill, Seventh edition, 2001.

[7] T. C. Hu Andrew and B. Kahng, Linear and Integer Programming Made Easy., Springer, 2016.

[8] A. Koberstein, The Dual Simplex Method, Techniques for a fast and stable implementation., PhD. Dissertation, 2005.

[9] Y. Pochet and L. A. Wolsey, Production Planning by Mixed Integer Programming., Springer, 2006.

[10] A. Schrijver, Theory of linear and integer programming.,John Wiley and Sons, Third Edition, 1986.

[11] H.A. Taha, Operations research: An introduction., Pearson Prentice Hall, Eighth Edition, 2007.

[12] L. A. Wolsey, Integer Programming., John Wiley and Sons Inc., 1998.