12-2003

# Contemporary Approaches to the Solution of the Integer Programming Problem

Djavanshir Gadjiev
*University of Tennessee, Knoxville*

To the Graduate Council:

I am submitting herewith a thesis written by Djavanshir Gadjiev entitled "Contemporary Approaches to the Solution of the Integer Programming Problem." I have examined the final electronic copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Mathematics.

Yueh-er Kuo, Major Professor

We have read this thesis and recommend its acceptance:

Accepted for the Council:
Carolyn R. Hodges

Vice Provost and Dean of the Graduate School

(Original signatures are on file with official student records.)

To the Graduate Council:

I am submitting herewith a thesis written by Djavanshir Djebrail Gadjiev entitled "Contemporary Approaches to the Solution of the Integer Programming Problem." I have examined the final paper copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Mathematics.

Yueh-er Kuo

Yueh-er Kuo, Major professor

We have read this thesis and
recommend its acceptance:

Don Hinton

Henry Simpson

Accepted for the Council:

Vice Provost and Dean of
Graduate Studies

# CONTEMPORARY APPROACHES TO THE SOLUTION OF THE INTEGER PROGRAMMING PROBLEM

A Thesis
Presented for the
Master of Science
Degree
The University of Tennessee, Knoxville

Djavanshir Djebrail Gadjiev
December 2003

# ACKNOWLEDGEMENTS

I wish to express my thankfulness to the faculty and staff of the Department of Mathematics at the University of Tennessee, Knoxville for their everyday support and encouragement. My special appreciation is ultimately for Dr. Yueh-er Kuo who advised me from the beginning to end of this thesis. Also, I enjoy to extend my utmost appreciation to the Committee members, Dr. Hinton, D.B. and Dr. Simpson, H., for their invaluable time and assistance.

I would also like to thank my wife Natasha and my children, Sabina and Emin, for their supportive words and vehement encouragement during the entire work on this thesis.

# ABSTRACT

The purpose of this thesis is to provide analysis of the modern development of the methods for solution to the integer linear programming problem. The thesis simultaneously discusses some main approaches that lead to the development of the algorithms for the solution to the integer linear programming problem.

Chapter 1 introduces the Generalized Linear Programming Problem alongside with the properties of a solution to the Linear Programming Problem. The simplex procedure presented to solve the Linear Programming Problem by adding slack variables along with the artificial-basis technique.

Chapter 2 refers to the primal-dual simplex procedure. The dual simplex algorithm reflects the dual simplex procedure.

Chapter 3 discusses the mixed and alternative formulations of the integer programming problem.

Chapter 4 considers the optimality conditions with the imposed relaxations to solve the Linear Programming Relaxation Problem. The methods of the Integer Programming are introduced for the Linear Programming Relaxation.

Chapter 5 discusses the concepts of the Branch-and-Bound method followed by the direct application of the Branch-and-Bound method.

Chapter 6 introduces the fundamental concepts of the cutting method. The main concept of the valid inequalities presented for the Linear Programming Problem as well as for the Integer Programming Problem. Gomory's Fractional cutting plane Algorithm represents the desired step to obtain the solution for the Integer Programming Problem. Furthermore, the mixed integer cuts generalizes the concepts to provide the corresponding solution for the Integer Programming Problem.

Chapter 7 describes the Gomory method for the pure Integer Program followed by the Gomory method for the mixed Integer Program.

In the Appendix the computer program LINDO is used. Throughout the whole thesis this computer program is applied to emphasize the very helpful tool in Linear Programming.

All above mentioned chapters include the variety of examples corresponding to the Linear Programming Problem and the Integer Program.

## TABLE OF CONTENTS

Chapter Page

I. THE GENERALIZED LINEAR PROGRAMMING PROBLEM ....................1
1.1 The Linear Programming Problem ............................................................1
1.2 Properties of a Solution to the Linear Programming Problem.....................2
1.3 The Simplex Procedure to Solve the Linear Programming Problem...........5
1.3.1 The Simplex Procedure by Adding Slack Variables......................11
1.3.2 The Artificial Basis Technique .....................................................13

II. THE DUAL SIMPLEX PROCEDURE ..........................................................20
2.1 The Primal-Dual Linear Programming Problem.......................................20
2.2 The Dual Simplex Procedure ...................................................................21

III. INTEGER PROGRAMMING.....................................................................25
3.1 Formulations of an Integer Program........................................................25
3.2 Mixed Integer Formulations ....................................................................29
3.3 Alternative Formulations.........................................................................31

IV. OPTIMALITY AND RELAXATION ...........................................................34
4.1 Optimality Conditions and Relaxations....................................................34
4.2 Linear Programming Relaxations ............................................................35
4.3 Methods of the Integer Programming.......................................................36

V. BRANCH-AND-BOUNND ALGORITHM.....................................................40
5.1 The Concepts of the Branch-and-Bound Method ....................................40
5.2 LP-based Branch-and-Bound Method .....................................................41
5.3 The Application of the Branch-and-Bound Method .................................49

VI. CUTTING PLANE ALGORITHM................................................................58
6.1 The Fundamental Concepts of the Cutting Method...................................58
6.2 Valid Inequalities.....................................................................................60
6.2.1 Some Valid Inequalities...............................................................61
6.3 Valid Inequalities for a Linear Programming Problem.............................64
6.3.1 Valid Inequalities for an IP Problem ............................................64
6.4 Gomory's Fractional Cutting Plane Algorithm..........................................66
6.4.1 Cutting Plane Algorithm..............................................................67
6.5 Mixed Integer Cuts .................................................................................69
6.5.1 The Mixed Integer Rounding Inequality (MIR) ............................69
6.5.2 Gomory's Mixed Integer Cut........................................................70

VII. **GOMORY METHOD and LP RELAXATION SIMPLEX ALGORITHM** ..72
7.1 Introduction........72
7.2 Gomory Method for Pure IP Problem........72
7.3 The LP-based Relaxation and Gomory Cuts ........77

**BIBLIOGRAPHY** ........82

**APPENDIX** ........85

**VITA** ........102

# LIST OF TABLES

Table Page

1.1 First Step of the Generalized Simplex Procedure ....6
1.2 Initial Step of Example 1.1 ....10
1.3 Second Step of Example 1.1 ....10
1.4 Third Step of Example 1.1 ....10
1.5 Initial Step of Example 1.2 ....12
1.6 Second Step of Example 1.2 ....12
1.7 First Step of Artificial Basis Technique ....14
1.8 Initial Step of Example 1.3 ....16
1.9 Second Step of Example 1.3 ....18
1.10 Third Step of Example 1.3 ....18
1.11 Fourth Step of Example 1.3 ....19
2.1 Initial Step of Example 2.1 ....24
2.2 Second Step of Example 2.1 ....24
5.1 The Simplex Tableau Representation for Example 5.1 ....44
7.1 Initial Tableau ....73
7.2 Initial Step of Example 7.1 ....78
7.3 Second Step of Example 7.1 ....80
7.4 Third Step of Example 7.1 ....80
7.5 Fourth Step of Example 7.1 ....81
7.6 Fifth Step of Example 7.1 ....81

## LIST OF FIGURES

Figure Page

3.1 Feasible and Infeasible Region for the IP Example 3.1 ........................................27
3.2 The Fixed Cost Function..........................................................................................30
3.3 Two Different Formulations for IP .........................................................................32
4.1 The Illustration of Cutting Plane Method ................................................................38
4.2 The Illustration of Branch-and-Bound Method ........................................................39
5.1 Partial Branch-and-Bound Tree 1 ............................................................................44
5.2 Partial Branch-and-Bound Tree 2 ............................................................................48
5.3 Complete Branch-and-Bound Tree ..........................................................................50
5.4 Division of the Feasible Region...............................................................................50
5.5 The Solution Tree ....................................................................................................54
6.1 The Cutting Planes I and II .....................................................................................59
6.2 The Convex Hull of X and the Valid Inequality .....................................................63

# CHAPTER 1

# THE GENERALIZED LINEAR PROGRAMMING PROBLEM

## 1.1 The Linear Programming Problem

In general, the Linear Programming Problem can be represented as it follows [10]:

Minimize the linear objective function of the form

$$z = c_1x_1 + c_2x_2 + \ldots + c_nx_n \qquad (1.1)$$

Subject to the linear constraints as it given by

$$\begin{aligned} &a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n = b_1 \\ &a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n = b_2 \\ &\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ &a_{i1}x_1 + a_{i2}x_2 + \ldots + a_{in}x_n = b_i \qquad (1.2) \\ &\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ &a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n = b_m \end{aligned}$$

where $x_j \geq 0$, j=1,2,…,n (1.3)

Note that the $a_{ij}, b_j$, and $c_j$ are prescribed constants with $m < n$. In addition, equation (1.2) can be multiplied by -1, if it is necessary to provide all $b_i$ to be ≥ 0. Using the notation of summation we can formulate the linear programming problem as it follows:

Minimize $\sum_{j=1}^{n} c_jx_j$

Subject to $\sum_{j=1}^{n} a_{ij}x_j = b_i$ , i=1,2,…,m

and $x_j \geq 0$, j=1,2,…,n

Using vector notation we can reformulate the problem in vector form:

Minimize $\mathbf{f(X) = cX}$

Subject to $\begin{array}{l} \mathbf{AX = b} \\ \mathbf{X \geq 0} \end{array}$

where $\mathbf{c} = (c_1, c_2, ..., c_n)$ is a row vector, $\mathbf{X} = (x_1, x_2, ..., x_n)$ is a column vector, $\mathbf{A} = (a_{ij})$ is an $m \times n$ matrix, $\mathbf{b} = (b_1, b_2, ..., b_n)$ is a column vector, and **0** is an n-dimensional null vector.

We have to note that the most commonly used notation is given below and such notation provides better illustration to the corresponding steps of the simplex procedure. This notation is in the form of

Minimize **cX**

Subject to $$x_1\mathbf{P}_1 + x_2\mathbf{P}_2 + ... + x_n\mathbf{P}_n = \mathbf{P}_0$$
$$\mathbf{X} \geq \mathbf{0}$$

where $\mathbf{P}_j$ is the j^th column of the matrix **A** given above and $\mathbf{P}_0 = \mathbf{b}$, j =1,2,…,n.

## 1.2. Properties of a Solution to the Linear Programming Problem

The properties of a solution to the Linear Programming Problem are contained in the articles of Dantzig ,G.B.[6] and Charnes,A.W., Cooper,W. and Henderson,A. [4]. These articles give the important properties of a solution to the general linear programming problem. There are the following definitions that describe important features of a solution to the linear programming problem.

*Definition 1.1* A feasible solution to the linear programming problem (1.1) – (1.3) is presented by the vector **X**=$(x_1, x_2, ..., x_n)$, which satisfies the constraints (1.2) and (1.3).

*Definition 1.2* A basic solution to (1.2) is obtained by setting (n-m) variables to zero and solving for the remaining m variables. These m variables are the basic variables and the determinant of the coefficients of the m variables is nonzero.

*Definition 1.3* If a basic solution satisfies (1.3) (nonnegative conditions), then such basic solution is called the basic feasible solution.

*Definition 1.4* If a basic feasible solution contains of exactly m possible $x_i$ , then such basic feasible solution is called a nondegenerate basic feasible solution; that means all basic variables are positive.

*Definition 1.5* If a feasible solution minimizes (1.1), then such solution is called a minimum feasible solution.

*Definition 1.6* If a basic feasible solution satisfies all conditions (1.1), (1.2) and (1.3), then such solution is called an *optimal basic feasible solution.*

Further, in the text, when we are going to mention a solution we will mean any feasible solution.

*Theorem 1.1* The set of all feasible solutions determines the feasible region and the latter is the convex set to the linear programming problem (the so-called LP-problem further throughout in the text).

The Theorem stated that every convex combination of any two feasible solutions is also a feasible solution. Let us assume that there are at least two solutions $\mathbf{X}_1$ and $\mathbf{X}_2$. We have $\mathbf{AX}_1 = \mathbf{b}$ for $\mathbf{X}_1 \geq \mathbf{0}$ and $\mathbf{AX}_2 = \mathbf{b}$ for $\mathbf{X}_2 \geq \mathbf{0}$. For $0 \leq \alpha \leq 1$ let $\mathbf{X} = \alpha\mathbf{X}_1 + (1-\alpha)\mathbf{X}_2$ be any convex combination of $\mathbf{X}_1$ and $\mathbf{X}_2$. Note, that all elements of **X** are nonnegative, that is $\mathbf{X} \geq \mathbf{0}$. Therefore, we have that **X** is a feasible solution and we have $\mathbf{AX} = \mathbf{A}[\alpha\mathbf{X}_1 + (1-\alpha)\mathbf{X}_2] = \alpha\mathbf{AX}_1 + (1-\alpha)\mathbf{AX}_2 = \alpha\mathbf{b} + \mathbf{b} - \alpha\mathbf{b} = \mathbf{b}$.

By Theorem if a problem has more than one solution it has an infinite number of solutions.

Our goal is to locate one solution, which minimizes the corresponding objective function. Next Theorem provides conditions required to find such solution and tells us to look at the extreme points of the convex polyhedron to determine the minimum feasible solution.

*Theorem 1.2* The objective function (1.1) gets its minimum at an extreme point of the convex set **K**. If the objective function assumes its minimum at more than one extreme point, then it acquires the same value for every convex combination of those points.
We recall that if **K** is a bounded convex polyhedron, then **K** is the convex hull of the extreme points of **K**. With the assumption that **K** is a convex polyhedron we can guess from the above discussion that we need only to look at the extreme point of the convex polyhedron in order to find the minimum feasible solution. The following theorem proves that.

*Theorem 1.3* The objective function (1.1) acquires its minimum at an extreme point of the convex set **K** of feasible solutions to the LP-problem. If the objective function gets its minimum in more than one point, then it takes on the same value for every convex combination of those particular points.

According to this Theorem, we have to consider only the extreme points of **K** in our quest for a minimum feasible solution to the LP-problem.

We know that a feasible solution is a vector $\mathbf{X} = (x_1, x_2, ..., x_n)$ with $x_i \geq 0$ such that $x_1\mathbf{P}_1 + x_2\mathbf{P}_2 + ... + x_n\mathbf{P}_n = \mathbf{P}_0$. If we assume that we found the set of linearly independent vectors $\mathbf{P}_1, \mathbf{P}_2, ..., \mathbf{P}_k$ then we have the following theorem:

*Theorem 1.4* If we found a set of $k \le m$ vectors $\mathbf{P}_1, \mathbf{P}_2, ..., \mathbf{P}_k$ which are linearly independent such that $x_1 \mathbf{P}_1 + ... + x_k \mathbf{P}_k = \mathbf{P}_0$ and all $x_i \ge 0$, then the point $\mathbf{X}=(x_1, x_2, ..., x_k, 0, ..., 0)$ is an extreme point of the convex set of feasible solutions, where **X** is n-dimensional vector, whose last (n-k) elements are zero.

*Theorem 1.5* If $\mathbf{X} = (x_1, x_2, \cdots, x_n)$ is an extreme point of **K**, then it follows that at most m of $X_i$ are positive and these $X_i$ form a linearly independent set.

From the discussion of the above presented theorems, we may assume that the set of vectors $\mathbf{P}_1, \mathbf{P}_2, ..., \mathbf{P}_n$ of the linear programming always contains a set of m linearly independent vectors. The following is a corollary that tells us about such an assumption:

*Corollary 1.1* Every extreme point of **K** has an associated set of m linearly independent vectors from the given set $\mathbf{P}_1, \mathbf{P}_2, ..., \mathbf{P}_n$.

We can summarize the preceding theorem by the following theorem:

*Theorem 1.6* If the positive $x_j$ are coefficients of linearly independent vectors $\mathbf{P}_j$ in $\sum_{j=1}^{n} x_j \mathbf{P}_j = \mathbf{P}_0$, then the $\mathbf{X} = (x_1, x_2, \cdots, x_n)$ is an extreme point of **K**.

This theorem and the preceding assumption tell us that

1. There exists an extreme point of **K** at which the objective function acquires its minimum
2. Every basic feasible solution is associated with the extreme point of **K**
3. Every extreme point of **K** has m linearly independent vectors

We can conclude from the above that we need to look at extreme point solution, and only those feasible solutions generated by m linearly independent vector. For the given set of n there are at most $\binom{n}{m}$ sets of m linearly independent vectors, and, hence, the value $\binom{n}{m}$ is an upper bound to the problem. For the large m and n it is an impossible task to evaluate all the possible solutions, which lead us to a minimum solution. That means, we need to get a computational scheme to find a minimum solution by selecting a small subsets of the possible solutions, which, eventually, converges to a minimum. The simplex procedure invented by Dantzig [7] is such an effective scheme. The idea of such procedure is to find an extreme point and determine the minimum of the objective function.

## 1.3 The Simplex Procedure to Solve the Linear Programming Problem

To begin the simplex procedure, we can arrange the problem matrix as shown in Table 1.1. We use the form as in [10].

The original equations of the problem are given by $\sum_{j=1}^{n} a_{ij}x_j = b_i, i = 1,2,\ldots,m.$ Let $x_{i0} = b_i$ and $x_{ij} = a_{ij}\ \ j = 0,1,\ldots,n$ is obtained by taking the inner product of the j<sup>th</sup> vector with the column vector labeled c, that is

$$z_0 = \sum_{i=1}^{m} c_i x_{i0}$$

$$z_j = \sum_{i=1}^{m} c_i x_{ij}, j = 1,2,\ldots,n$$

The elements $z_0$ and $z_j - c_j$ are entered in the (m+1)st row of their respective columns. Note that $z_j - c_j$ for the vectors in the basis is always equal to 0. Suppose that all the numbers $z_j - c_j \le 0$ for $j = 1,2,\cdots,n.$ Then, the solution $\mathbf{X}_0 = (x_{10}, x_{20}, \cdots, x_{m0}) = (b_1, b_2, \cdots, b_m)$ is a minimum feasible solution with the corresponding value of the objective function is $z_0$. If we assume that there is at least one $z_j - c_j > 0$ then we have to compute a new feasible solution whose basis consists of m-1 vectors of the original basis $\mathbf{P}_1, \mathbf{P}_2, \cdots, \mathbf{P}_m$. In order to find a new vector to enter the basis we need to select a vector which gives the greatest decrease in the value of the objective function. Such vector should the one for which

$$\max_j \theta_0 (z_j - c_j), \qquad (1.4)$$

where for each j, $\theta_0 = \min_i \frac{x_{i0}}{x_{ij}} > 0$. However, if there are a number of j for which $z_j - c_j > 0$, the above criterion is rather difficult to use. The simplest criterion to select the vector which corresponds to

$$\max_j (z_j - c_j). \qquad (1.5)$$

If there are ties, the rule suggests to choose the vector with the lowest index j. However, the highest index j with the corresponding vector can be selected. Further, we will employ the second criterion (1.5), where $\max_j (z_j - c_j) = z_k - c_k > 0$. Therefore, the vector $\mathbf{P}_k$ is going to be entered the basis. For the next step, we shall compute

$\theta_0 = \min_i \frac{x_{i0}}{x_{ik}}$ for $x_{ik} > 0$.

**Table 1.1 First Step of the Generalized Simplex Procedure**

|  |  |  |  | $c_1$ | $c_2$ | $\cdots$ | $c_l$ | $\cdots$ | $c_m$ | $c_{m+1}$ | $\cdots$ | $c_k$ | $\cdots$ | $c_n$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| i | Basis | c | $\mathbf{P}_0$ | $\mathbf{P}_1$ | $\mathbf{P}_2$ | $\cdot$ | $\mathbf{P}_l$ | $\cdot$ | $\mathbf{P}_m$ | $\mathbf{P}_{m+1}$ | $\cdot$ | $\mathbf{P}_k$ | $\cdot$ | $\mathbf{P}_n$ |
| 1 | $\mathbf{P}_1$ | $c_1$ | $x_{10}$ | 1 | 0 |  | 0 |  | 0 | $x_{1,m+1}$ |  | $x_{1k}$ |  | $x_{1n}$ |
| 2 | $\mathbf{P}_2$ | $c_2$ | $x_{20}$ | 0 | 1 |  | 0 |  | 0 | $x_{2,m+1}$ |  | $x_{2k}$ |  | $x_{2n}$ |
| $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ |  | $\vdots$ |  | $\vdots$ | $\vdots$ |  | $\vdots$ |  | $\vdots$ |
| $l$ | $\mathbf{P}_l$ | $c_l$ | $x_{l0}$ | 0 | 0 |  | 1 |  | 0 | $x_{l,m+1}$ |  | $x_{lk}$ |  | $x_{ln}$ |
| $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ |  | $\vdots$ |  | $\vdots$ | $\vdots$ |  | $\vdots$ |  | $\vdots$ |
| $m$ | $\mathbf{P}_m$ | $c_m$ | $x_{m0}$ | 0 | 0 |  | 0 |  | 1 | $x_{m,m+1}$ |  | $x_{mk}$ |  | $x_{mn}$ |
| $m+1$ |  |  | $z_0$ | 0 | 0 | $\cdot$ | 0 | $\cdot$ | 0 | $z_{m+1}-c_{m+1}$ | $\cdot$ | $z_k-c_k$ | $\cdot$ | $z_n-c_n$ |

In a case for all $x_{ik} \leq 0$ we can find a feasible solution that makes the value of the objective function arbitrarily small, and our computation is completed.

In a case if some $x_{ik} > 0$ and $\theta_0 = \min\limits_i \frac{x_{io}}{x_{ik}} = \frac{x_{l0}}{x_{lk}}$ we have to remove the vector $\mathbf{P}_l$ from the basis. Thus, our new feasible solution consists of $\mathbf{P}_1, \cdots, \mathbf{P}_{l-1}, \mathbf{P}_k, \mathbf{P}_{l+1}, \cdots, \mathbf{P}_m$. Next, we have to compute the new solution explicitly and express each vector not in the basis in terms of the new basis.

Our initial basis is $(\mathbf{P}_1, \mathbf{P}_2, \cdots, \mathbf{P}_m) = \mathbf{I}_m$ and we can express all the vectors $P_j$ in terms of this basis.

$$\mathbf{P}_1 = x_{10}\mathbf{P}_1 + \cdots + x_{l0}\mathbf{P}_l + \cdots + x_{m0}\mathbf{P}_m, \qquad (1.6)$$

$$\mathbf{P}_k = x_{1k}\mathbf{P}_1 + \cdots + x_{lk}\mathbf{P}_l + \cdots + x_{mk}\mathbf{P}_m, \qquad (1.7)$$

and

$$\mathbf{P}_j = x_{1j}\mathbf{P}_1 + \cdots + x_{lj}\mathbf{P}_l + \cdots + x_{mj}\mathbf{P}_m. \qquad (1.8)$$

From (1.7) we have

$$\mathbf{P}_l = \frac{1}{x_{lk}}(\mathbf{P}_k - x_{1k}\mathbf{P}_1 - \cdots - x_{mk}\mathbf{P}_m). \qquad (1.9)$$

Substituting (1.9) in (1.6) for the expression $P_l$ we have

$$\mathbf{P}_0 = x_{10}\mathbf{P}_1 + \cdots + x_{l0}\left[\frac{1}{x_{lk}}(\mathbf{P}_k - x_{1k}\mathbf{P}_1 - \cdots - x_{mk}\mathbf{P}_m)\right] + \cdots + x_{m0}\mathbf{P}_m,$$

or regrouping the like terms we have

$$\mathbf{P}_0 = \left(x_{10} - \frac{x_{l0}}{x_{lk}}x_{1k}\right)\mathbf{P}_1 + \cdots + \frac{x_{l0}}{x_{lk}}\mathbf{P}_k + \cdots + \left(x_{m0} - \frac{x_{l0}}{x_{lk}}x_{mk}\right)\mathbf{P}_m.$$

From the latter equality we can write

$$\mathbf{P}_0 = x'_{10}\mathbf{P}_1 + \cdots + x'_{k0}\mathbf{P}_k + \cdots + x'_{m0}\mathbf{P}_m \textit{ where}$$

$$x'_{i0} = x_{i0} - \frac{x_{l0}}{x_{lk}}x_{ik}, \textit{for } i = 1, 2, \cdots, l-1, l+1, \cdots m \qquad (1.10)$$

$$x'_{k0} = \frac{x_{l0}}{x_{lk}}$$

and the corresponding new feasible solution is given by $\mathbf{X}'_0 = (x'_{10}, \cdots, x'_{k0}, \cdots, x'_{m0}), x'_{i0} \geq 0$. In a similar order, if we substitute (1.9) into (1.7), we can obtain the expression for each $\mathbf{P}_j$ not in the new basis in terms of this basis. Then we have

$$\mathbf{P}_j = x'_{1j}\mathbf{P}_1 + \cdots + x'_{kj}\mathbf{P}_k + \cdots + x'_{mj}\mathbf{P}_m \ \textit{where}$$

$$x'_{ij} = x_{ij} - \frac{x_{lj}}{x_{lk}} x_{ik}, \textit{for } i \neq l \qquad (1.11)$$

$$x'_{k0} = \frac{x_{lj}}{x_{lk}}$$

Since $z'_j - c_j = x'_{1j}c_1 + \cdots + x'_{kj}c_k + \cdots + x'_{mj}c_m - c_j$ we can verify by substituting the values (1.11) for $x'_{ij}$ that

$$z'_j - c_j = z_j - c_j - \frac{x_{lj}}{x_{lk}}(z_k - c_k) \qquad (1.12)$$

and by substituting the values (1.10) for $x'_{l0}$ into $z'_0 = c_1x'_{10} + \cdots + c_kx'_{k0} + \cdots + c_mx'_{m0}$ that

$$z'_0 = z_0 - \frac{x_{l0}}{x_{lk}}(z_k - c_k)\,. \qquad (1.13)$$

Note that the transformation formulas for the new solution $\mathbf{X}'_0$, the new vector $\mathbf{X}'_j$ and the corresponding $z'_j - c_j$ for rows $i = 1, \cdots, m+1$ and columns $j = 0, 1, \cdots, n$ in Table 1.1 are given

$$x'_{ij} = x_{ij} - \frac{x_{lj}}{x_{lk}} x_{ik} \ \textit{for } i \neq l$$

$$(1.14)$$

$$x'_{ij} = \frac{x_{lj}}{x_{lk}}$$

where $z'_0 = x'_{m+1,0}, z'_j - c_j = x'_{m+1,j}$. Note that the transformation formulas defined by (1.14) is equivalent to the full elimination process with the pivot element $x_{lk}$.

This simplex procedure calls for the successive iteration by

1. The testing whether $z_j - c_j \leq 0$ for all j.
2. The selection of the vector to be entered the basis for some $z_j - c_j > 0$, i.e. selection of the $\max_j(z_j - c_j)$.
3. The selection of the vector to be removed from the basis, i.e. the vector with $\min_i\left(\frac{x_{i0}}{x_{ik}}\right)$ for $x_{ik} > 0$, where k corresponds to the vector in step 2. If all $x_{ik} \leq 0$, then the solution is unbounded.
4. The transformation of the tableau by the elimination to obtain the new solution.

Each iteration produces a new feasible solution, and, finally, we will obtain a minimum solution or determine an unbounded solution.

*Example 1.1* Let us solve the linear programming problems using the simplex procedure.

Minimize $x_2 - 4x_3 + 3x_5$

Subject to
$$\begin{aligned} &x_1 + 4x_2 - x_3 + 4x_5 = 8 \\ &-3x_2 + 4x_3 + x_4 = 9 \\ &-5x_2 + 3x_3 + 9x_5 + x_6 = 12 \\ &x_j \geq 0 \end{aligned}$$

Our initial basis (see Table 1.2) consists of $\mathbf{P}_1, \mathbf{P}_4, \mathbf{P}_6$ and the corresponding solution is $\mathbf{X}_0 = (x_1, x_4, x_6) = (8,9,12)$. The corresponding value of the objective function $z_0 = 0$ since $c_1 = c_4 = c_6 = 0$.

Since $\max_j (z_j - c_j) = z_3 - c_3 = 4 > 0$, $\mathbf{P}_3$ is selected to enter the basis with $\theta_0 = \min\left(\frac{9}{4}, \frac{12}{3}\right) = \frac{9}{4}$ and hence $\mathbf{P}_4$ is removed.

We transform the tableau (see Table 1.3) and obtain the new solution $\mathbf{X}_0' = (x_1, x_3, x_6) = \left(10\tfrac{1}{4}, \frac{9}{4}, 18\tfrac{3}{4}\right)$ with the value of the objective function is -9.

In the second step, since $\max(z_j' - c_j) = z_2' - c_2 = 3 > 0$ and $\theta_0 = \min\left(\frac{41}{13}, \frac{75}{16}\right) = \frac{41}{13}$, $P_2$ is introduced into the basis and $\mathbf{P}_1$ is removed. We transform the second-step values of Table 1.3 and obtain the third step Table 1.4.

We obtain the third solution $\mathbf{X}_0'' = (x_2, x_3, x_6) = \left(\frac{41}{13}, \frac{159}{16}, \frac{319}{52}\right)$ with the value of the objective function equal to $-\frac{240}{13}$. Since $\max(z_j'' - c_j) = 0$ the solution is a minimum feasible solution.

**Table 1.2 Initial Step of Example 1.1**

|  |  |  |  | 0 | 1 | -4 | 0 | 3 | 0 |  |
|---|---|---|---|---|---|---|---|---|---|---|
| i | Basis | c | $P_0$ | $P_1$ | $P_2$ | $P_3$ | $P_4$ | $P_5$ | $P_6$ | $\theta$ |
| 1 | $P_1$ | 0 | 8 | 1 | 4 | -1 | 0 | 4 | 0 |  |
| 2 | $P_4$ | 0 | 9 | 0 | -3 | $\boxed{4}$ | 1 | 0 | 0 | $9/4$ |
| 3 | $P_6$ | 0 | 12 | 0 | -5 | 3 | 0 | 9 | 1 | $12/3$ |
| 4 |  |  | 0 | 0 | -1 | 4 | 0 | -3 | 0 |  |

**Table 1.3 Second Step of Example 1.1**

|  |  |  | $P_0$ | $P_1$ | $P_2$ | $P_3$ | $P_4$ | $P_5$ | $P_6$ |  |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | $P_1$ | 0 | $10\,1/4$ | 1 | $\boxed{3\,1/4}$ | 0 | $1/4$ | 4 | 0 | $41/13$ |
| 2 | $P_3$ | 0 | $9/4$ | 0 | $-3/4$ | 1 | $1/4$ | 0 | 0 |  |
| 3 | $P_6$ | 0 | $18\,3/4$ | 0 | 4 | 0 | -3 | 9 | 1 | $75/16$ |
|  |  |  | -9 | 0 | 3 | 0 | -1 | -3 | 0 |  |

**Table 1.4 Third Step of Example 1.1**

|  |  |  | $P_0$ | $P_1$ | $P_2$ | $P_3$ | $P_4$ | $P_5$ | $P_6$ |
|---|---|---|---|---|---|---|---|---|---|
| 1 | $P_2$ | 0 | $41/13$ | $4/13$ | 1 | 0 | $1/13$ | $16/13$ | 0 |
| 2 | $P_3$ | 0 | $159/16$ | $3/13$ | 0 | 1 | $4/13$ | $12/13$ | 0 |
| 3 | $P_6$ | 0 | $319/52$ | $-16/13$ | 0 | 0 | $-43/13$ | $181/13$ | 1 |
|  |  |  | $-240/13$ | $-12/13$ | 0 | 0 | $-16/13$ | $-87/13$ | 0 |

### 1.3.1 The Simplex Procedure by Adding Slack Variables

If the Linear-Programming problem is given in the form $\mathbf{AX} \le \mathbf{b}$, we have to add additional variables called slack variables, in order to get an initial basis for the first step of the simplex procedure. Such slack variables usually represent the unused resource with the corresponding cost coefficients equal to zero. Next example illustrates the slack variables approach.

*Example 1.2* Let us consider the problem

Minimize $z = x_1 + x_2 - 3x_3$

$$\text{Subject to} \quad \begin{array}{l} -x_1 - 4x_3 \le 7 \\ 3x_1 - 2x_2 + x_3 \le 4 \\ 3x_1 + 6x_2 + x_3 \le 7 \\ x_j \ge 0, j = 1,2,3 \end{array}$$

If we add slack variables then the problem becomes as it follows:

Minimize $z = x_1 + x_2 - 3x_3$

$$\text{Subject to} \quad \begin{array}{l} -x_1 - 4x_3 + x_4 = 7 \\ 3x_1 - 2x_2 + x_3 + x_5 = 4 \\ 3x_1 + 6x_2 + x_3 + x_6 = 7 \\ x_j \ge 0, j = 1,\cdots,6 \end{array}$$

The set of constraints now consists of an initial basis of the slack variables $(\mathbf{P}_4, \mathbf{P}_5, \mathbf{P}_6)$ with the respective first feasible solutions of $x_1 = 0, x_2 = 0, x_3 = 0, x_4 = 7, x_5 = 4, x_6 = 7$. The associated value of the objective function is zero since all cost coefficients for the slack variables are zero. Hence, we are able to apply the simplex procedure.

From Table 1.5 we see that vector $\mathbf{P}_3$ replaces vector $\mathbf{P}_6$.

Table 1.5 and Table 1.6 show that the $\max(z_j - c_j) = 0$, i.e. that the minimum feasible solution has been obtained. The solution vector is $\mathbf{X} = (x_4, x_5, x_3) = \left(\frac{5}{7}, \frac{18}{7}, \frac{3}{7}\right)$ with the objective function value of -3.

**Table 1.5 Initial Step of Example 1.2**

|  |  |  |  | 1 | 1 | -3 | 0 | 0 | 0 |  |
|---|---|---|---|---|---|---|---|---|---|---|
| i | Basis | c | $P_0$ | $P_1$ | $P_2$ | $P_3$ | $P_4$ | $P_5$ | $P_6$ | $\theta$ |
| 1 | $P_4$ | 0 | 7 | -1 | 0 | -4 | 1 | 0 | 0 |  |
| 2 | $P_5$ | 0 | 4 | 3 | -2 | 1 | 0 | 1 | 0 | $4/1 = 4$ |
| 3 | $P_6$ | 0 | 1 | $3/7$ | $6/7$ | $\boxed{1}$ | 0 | 0 | 1 | $7/7 = 1$ |
| 4 |  |  | 0 | -1 | -1 | 3 | 0 | 0 | 0 |  |

**Table 1.6 Second Step of Example 1.2**

|  |  |  | $P_0$ | $P_1$ | $P_2$ | $P_3$ | $P_4$ | $P_5$ | $P_6$ |
|---|---|---|---|---|---|---|---|---|---|
| 1 | $P_4$ | 0 | 11 | $5/7$ | $24/7$ | 0 | 1 | 0 | $4/7$ |
| 2 | $P_5$ | 0 | 3 | $18/7$ | $-20/7$ | 0 | 0 | 1 | $-1/7$ |
| 3 | $P_3$ | -3 | 1 | $3/7$ | $6/7$ | 1 | 0 | 0 | $1/7$ |
|  |  |  | -3 | $-16/7$ | $-25/7$ | 0 | 0 | 0 | $-3/7$ |

### 1.3.2 The Artificial Basis Technique

The artificial basis technique, known as the Big-M Method [9] calls upon the problems with constraints of the form $\mathbf{AX} \geq \mathbf{b}$. To apply the simplex procedure we have to augment the general Linear-Programming problem (1.1) – (1.3) as it follows.

Minimize $c_1 x_1 + \cdots + c_n x_n + w x_{n+1} + w x_{n+2} + \cdots + w x_{n+n}$

Such that
$$\begin{aligned} a_{11}x_1 + \cdots + a_{1n}x_n + x_{n+1} &= b_1 \\ a_{21}x_1 + \cdots + a_{2n}x_n + x_{n+2} &= b_2 \\ &\vdots \\ a_{m1}x_1 + \cdots + a_{mn}x_n + x_{n+m} &= b_m \\ x_j \geq 0, j = 1, \cdots, n, n+1, \cdots n+m \end{aligned}$$

The quantity $w$ is to be an unspecified large positive number. The vectors $\mathbf{P}_{n+1}, \mathbf{P}_{n+2}, \cdots, \mathbf{P}_{n+m}$ form an artificial basis for the augmented system. The first feasible solution is $\mathbf{X}_0 = (x_{n+1,0}, x_{n+2,0}, \cdots, x_{n+m,0}) = (b_1, b_2, \cdots, b_m) \geq 0$, and the corresponding value of the objective function $z_0 = w\sum_{i=1}^{m} b_i$. The basis is a unit matrix

$\mathbf{X}_j = (x_{1j}, x_{2j}, \cdots, x_{mj}) = (a_{1j}, a_{2j}, \cdots, a_{mj})$ and $z_j = w\sum_{i=1}^{m} x_{ij}$.

For the first solution $z_j - c_j = w\sum_{i=1}^{m} x_{ij} - c_j$, each $z_j - c_j$ has a $w$ coefficient and a non-$w$ coefficient which are independent of each other. The associated simplex procedure is set up as Table 1.7, where for each j, the non-$w$ term and the $w$ term of $z_j - c_j$ have been placed in the (m+1)$^{st}$ and (m+2)$^{nd}$ rows. The difference from the original simplex tableau is given by the vector introduced into the basis with the largest positive element in the (m+2)$^{nd}$ row. For the first step, the vector corresponding to $\max_j \sum_{i=1}^{m} x_{ij}$ is introduced into the basis. The elements of the (m+2)$^{nd}$ row are also changing by the elimination procedure. The main feature corresponds to the artificial vector, which once is removed from the basis, it is never reentered it. That means, we do not need to transform the last m columns of the tableau.

We have the following criterion to get a vector introduced into a basis, using the elements in the (m+2)$^{nd}$ row until either (1) all the artificial vectors are removed from the basis, or (2) there is no positive (m+2)$^{nd}$ element. The first case implies that all the elements in the (m+2)$^{nd}$ row are zero and the corresponding basis is a feasible basis, and we can then apply the regular simplex procedure. In the second case if the artificial part of the

**Table 1.7 First Step of Artificial Basis Technique**

| | | | | $c_1$ | | $c_k$ | | $c_n$ | $w$ | | $w$ | | $w$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $i$ | *Basis* | $c$ | $\mathbf{P}_0$ | $\mathbf{P}_1$ | | $\mathbf{P}_k$ | | $\mathbf{P}_n$ | $\mathbf{P}_{n+1}$ | | $\mathbf{P}_{n+l}$ | | $\mathbf{P}_{n+m}$ |
| 1 | $\mathbf{P}_{n+1}$ | $w$ | $x_{n+1,0}$ | $x_{11}$ | | $x_{1k}$ | | $x_{1n}$ | 1 | | 0 | | 0 |
| 2 | $\mathbf{P}_{n+2}$ | $w$ | $x_{n+2,0}$ | $x_{21}$ | | $x_{2k}$ | | $x_{2n}$ | 0 | | 0 | | 0 |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | . | ⋮ | . | ⋮ | ⋮ | . | ⋮ | . | ⋮ |
| $l$ | $\mathbf{P}_{n+l}$ | $w$ | $x_{n+l,0}$ | $x_{l1}$ | | $x_{lk}$ | | $x_{ln}$ | 0 | | 1 | | 0 |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | | ⋮ | | ⋮ | ⋮ | | ⋮ | | ⋮ |
| $m$ | $\mathbf{P}_{n+m}$ | $w$ | $x_{n+m,0}$ | $x_{m1}$ | | $x_{mk}$ | | $x_{mn}$ | 0 | | 0 | | 1 |
| $m$+1 | | | 0 | $-c_1$ | . | $-c_k$ | . | $-c_n$ | 0 | . | 0 | . | 0 |
| $m$+2 | | | $\sum x_{n+i,0}$ | $\sum x_{i1}$ | . | $\sum x_{ik}$ | . | $\sum x_{in}$ | 0 | . | 0 | . | 0 |

corresponding value of the objective function is positive, then the original problem is not feasible. If the (m+2,0) element is equal to zero, then there is a degenerate feasible solution with at least one artificial vector. The artificial variables are zeros. We have not obtained the minimum feasible solution. We will introduce a vector that corresponds to the maximum positive element in the (m+1)st row which is above a zero element in the (m+2)nd row. Such criterion is supposed to be used until the minimum solution has been obtained, i.e. until there are no more positive (m+1)st elements above a zero in the (m+2)nd row. Using such criterion, we note that the elements in the (m+2)nd row are never transformed caused by the $z_j - c_j$ of the vector is $0 \cdot w + \left(z_j - c_j\right)$. Note that the unit vectors of the original problem along side with the necessary artificial vectors must be used in the initial basis in order to decrease the total number of iterations. The following example represents the artificial-basis technique.

*Example 1.3* Let us have the problem

Minimize $z = 2x_1 + 2x_2$

$$\text{Such that}\quad \begin{array}{l} 3x_1 + x_2 \geq 4 \\ 4x_1 + 3x_2 \geq 7 \\ x_1 + 2x_2 \geq 2 \\ x_j \geq 0, j = 1,2 \end{array}$$

We subtract the artificial variables $x_3, x_4$, and $x_5$ from the constraints 1, 2, and 3 respectively, to make each constraint to be equalities. In order to obtain a basis we also have to add the variables $x_6, x_7$, and $x_8$ to each constraint. Note, that we must add $wx_6, wx_7$, and $wx_8$ to the objective function. After that, the problem becomes as it follows.

Minimize $z = 2x_1 + 2x_2 + wx_6 + wx_7 + wx_8$

$$\text{Such that}\quad \begin{array}{l} 3x_1 + x_2 - x_3 + x_6 = 4 \\ 4x_1 + 3x_2 - x_4 + x_7 = 7 \\ x_1 + 2x_2 - x_5 + x_8 = 2 \\ x_j \geq 0, j = 1, \cdots, 8 \end{array}$$

The initial step is recorded in Table 1.8.

### Table 1.8 Initial Step of Example 1.3

| | | | | 2 | 2 | 0 | 0 | 0 | w | w | w | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| i | Basis | c | $\mathbf{P}_0$ | $\mathbf{P}_1$ | $\mathbf{P}_2$ | $\mathbf{P}_3$ | $\mathbf{P}_4$ | $\mathbf{P}_5$ | $\mathbf{P}_6$ | $\mathbf{P}_7$ | $\mathbf{P}_8$ | $\theta$ |
| 1 | $\mathbf{P}_6$ | $w$ | 4 | $\boxed{3}$ | 1 | -1 | 0 | 0 | 1 | 0 | 0 | $\frac{4}{3}=1\frac{1}{3}$ |
| 2 | $\mathbf{P}_7$ | $w$ | 7 | 4 | 3 | 0 | -1 | 0 | 0 | 1 | 0 | $\frac{7}{4}=1\frac{3}{4}$ |
| 3 | $\mathbf{P}_8$ | $w$ | 2 | 1 | 2 | 0 | 0 | -1 | 0 | 0 | 1 | $\frac{2}{1}=2$ |
| 4 | | | 0 | -2 | -2 | 0 | 0 | 0 | 0 | 0 | 0 | |
| 5 | | | 13 | $\boxed{8}$ | 6 | -1 | -1 | -1 | 0 | 0 | 0 | |

The $\mathbf{P}_1$ column has the largest positive element 8 in its (m+2)nd row, so vector $\mathbf{P}_1$ is entered into the basis. Calculating $\theta$ to find its minimum tells us that vector $\mathbf{P}_6$ is removed from the basis. Table 1.9 shows this result.

There is the largest positive element $\frac{10}{3}$ in the (m+2)nd row and the minimum value of $\theta$ is $\frac{2}{5}$. That means $\mathbf{P}_2$ is entered the basis, and $\mathbf{P}_8$ is removed from it (see Table 1.10).

Again, checking the (m+2)nd row for the largest positive element and the corresponding minimum $\theta$ we found that $\mathbf{P}_3$ is entered and $\mathbf{P}_7$ is removed from the basis. The fourth step is shown in Table 1.11.

In the fourth step, as all artificial variables have been removed, we use the usual simplex criterion with maximum (m+1,j) element that corresponds to the selection of the vector to enter the basis. Since all $z_j - c_j \leq 0$ in our particular example, then the solution $\mathbf{X} = (x_1, x_2, x_3)$ is the optimal solution of $z = \frac{54}{15}$ with $x_1 = \frac{24}{15}, x_2 = \frac{1}{5}, x_3 = 1$.

**Table 1.9 Second Step of Example 1.3**

| | | $P_0$ | $P_1$ | $P_2$ | $P_3$ | $P_4$ | $P_5$ | $P_7$ | $P_8$ | $\theta$ |
|---|---|---|---|---|---|---|---|---|---|---|
| $P_1$ | 2 | $\frac{4}{3}$ | 1 | $\frac{1}{3}$ | $-\frac{1}{3}$ | 0 | 0 | 0 | 0 | 4 |
| $P_7$ | $w$ | $\frac{5}{3}$ | 0 | $\frac{5}{3}$ | $\frac{4}{3}$ | -1 | 0 | 1 | 0 | 1 |
| $P_8$ | $w$ | $\frac{2}{3}$ | 0 | $\boxed{\frac{5}{3}}$ | $\frac{1}{3}$ | 0 | -1 | 0 | 1 | $\frac{2}{5}$ |
| | | $\frac{8}{3}$ | 0 | $-\frac{4}{3}$ | $-\frac{2}{3}$ | 0 | 0 | 0 | 0 | |
| | | $\frac{7}{3}$ | 0 | $\boxed{\frac{10}{3}}$ | $\frac{5}{3}$ | -1 | -1 | 0 | 0 | |

**Table 1.10 Third Step of Example 1.3**

| | | $P_0$ | $P_1$ | $P_2$ | $P_3$ | $P_4$ | $P_5$ | $P_7$ | $\theta$ |
|---|---|---|---|---|---|---|---|---|---|
| $P_1$ | 2 | $\frac{18}{15}$ | 1 | 0 | $-\frac{6}{15}$ | 0 | 0 | 0 | |
| $P_7$ | $w$ | 1 | 0 | 0 | $\boxed{1}$ | -1 | 0 | 1 | $\frac{1}{1}=1$ |
| $P_2$ | 2 | $\frac{2}{5}$ | 0 | 1 | $\frac{1}{5}$ | 0 | -1 | 0 | $\frac{2}{5}\div\frac{1}{5}=2$ |
| | | $\frac{48}{15}$ | 0 | 0 | $-\frac{6}{15}$ | 0 | 0 | 0 | |
| | | 1 | 0 | 0 | $\boxed{1}$ | -1 | -1 | 0 | |

**Table 1.11 Fourth Step of Example 1.3**

| | | $P_0$ | $P_1$ | $P_2$ | $P_3$ | $P_4$ | $P_5$ |
|---|---|---|---|---|---|---|---|
| $P_1$ | 2 | $\frac{24}{15}$ | 1 | 0 | 0 | $-\frac{6}{15}$ | $\frac{9}{15}$ |
| $P_3$ | 0 | 1 | 0 | 0 | 1 | -1 | 1 |
| $P_2$ | 2 | $\frac{1}{5}$ | 0 | 1 | 0 | $\frac{1}{5}$ | $-\frac{4}{5}$ |
| | | $\frac{54}{15}$ | 0 | 0 | 0 | $-\frac{6}{15}$ | $-\frac{6}{15}$ |
| | | 0 | 0 | 0 | 0 | 0 | 0 |

# CHAPTER 2

# THE DUAL SIMPLEX PROCEDURE

## 2.1 The Primal-Dual Linear Programming Problem

Associated with every Linear Programming problem as described in Chapter 1, there is an optimization problem called the dual problem. If the initial simplex tableau contains an $m \times m$ matrix, then the solution of the either problem results in an explicit solution of the other. The main theorems associated with the dual problem are given by Dantzig and Orden [8].

For the primal problem we shall find a column vector $\mathbf{X} = (x_1, x_2, ..., x_n)$, which minimizes the linear objective function $\mathbf{f(X)} = \mathbf{cX}$ subject to $\mathbf{AX} = \mathbf{b}$ and $\mathbf{X} \geq \mathbf{0}$.

Note that the number of rows of matrix **A** is less than the number of columns of **A**.

For the associated dual problem we have to find a row vector $\mathbf{W} = (w_1, w_2, ..., w_m)$, which maximizes the linear function $\mathbf{g(W)} = \mathbf{Wb}$ subject to $\mathbf{WA} \leq \mathbf{c}$. Note that for the dual problem the variables $w_i$ are not restricted to be nonnegative. For both problems we have that $\mathbf{c} = (c_1, c_2, ..., c_n)$ is a row vector, $\mathbf{b} = (b_1, b_2, ..., b_m)$ is a column vector, and $\mathbf{A} = (a_{ij})$ is the matrix of coefficients.

There is the following theorem associated with the primal and dual problems:

*Theorem 2.1* If the primal or dual problem has a finite solution, then the other problem has a finite optimal solution and $\min \mathbf{f(X)} = \max \mathbf{g(W)}$. In a case, if there is an unbounded optimal solution, then the other problem has no feasible solution.

In a slightly modified form this theorem was restated by Dantzig as it follows:

*Theorem 2.2* If a feasible solution to both problems exists, then there is an optimal solution to both problems and $\min \mathbf{f(X)} = \max \mathbf{g(W)}$.

As it was noted, the variables $w_i$ are not restricted to be nonnegative. If we multiply the $1 \times m$ row vector **W** by the $m \times m$ matrix **A**, then we have the following constraints:

$$
\begin{aligned}
&a_{11}w_1 + a_{12}w_2 + \ldots + a_{m1}w_m \le c_1 \\
&a_{12}w_1 + a_{22}w_2 + \ldots + a_{m2}w_m \le c_2 \\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\
&a_{1n}w_1 + a_{2n}w_2 + \ldots + a_{mn}w_m \le c_n
\end{aligned}
$$

The matrix of the coefficients of the above given constraints is $\mathbf{A}^T$, where $\mathbf{A}^T$ is a transposed of the primal matrix **A**. That means, there is a dual variable for each constraint of the primal problem.

## 2.2 The Dual Simplex Procedure

In the sense of a feasible solution leading to the optimality of the objective function for the dual problems the respective algorithm was devised by C.E. Lemke and later on Cooper and Steinberg applied the feasible point solution to either primal or dual problems [5].

Suppose that we selected a basis $\mathbf{B} = (\mathbf{P}_1\mathbf{P}_2 \ldots \mathbf{P}_m)$ such that at least one element of $\mathbf{B}^{-1}\mathbf{b}$ is negative and $\mathbf{c}^0\mathbf{B}^{-1}\mathbf{P}_j \le c_j$ for all $j$, where $\mathbf{c}^0 = (c_1, c_2, \ldots, c_m)$ is a row vector. The corresponding solution to the dual constraints is given by $\mathbf{W}^0 = \mathbf{c}^0\mathbf{B}^{-1}$ with the respective value of the objective function as $\mathbf{c}^0\mathbf{B}^{-1}\mathbf{b}$.

The dual simplex procedure is supposed to yield a maximizing solution and then, by the duality theorem will give us a minimizing solution to the primal.

Suppose we denote the rows of $\mathbf{B}^{-1}$ by $\mathbf{B}_i$. The $m$ components of the nonfeasible solution of the primal are $x_{i0} = \mathbf{B}_i\mathbf{b}$ for $i = 1, 2, \ldots, m$. If we let $x_{l0} = \mathbf{B}_l\mathbf{b} = \min\limits_i \mathbf{B}_i\mathbf{b} < 0$, then the vector $\mathbf{P}_l$ is going to be removed from the basis. For the vectors not in the basis $\mathbf{W}^0\mathbf{P}_j < c_j$, we shall compute $x_{ij} = \mathbf{B}_l\mathbf{P}_j$. We assume that at least one $x_{lj} < 0$. For the set $x_{ij} < 0$ we have to form ratios $\dfrac{z_j - c_j}{x_{lj}}$.

Let $\theta = \min\limits_{x_{lj}<0} \dfrac{z_j - c_j}{x_{lj}} = \dfrac{z_k - c_k}{x_{lk}} > 0$. The expression for θ indicates that $\mathbf{P}_k$ is selected to replace $\mathbf{P}_l$. The new basis $\overline{\mathbf{B}} = (\mathbf{P}_1 \ldots \mathbf{P}_l\mathbf{P}_k\mathbf{P}_{l+1} \ldots \mathbf{P}_m)$ and $\overline{\mathbf{B}^{-1}}$ is obtained by using the formulas for $\mathbf{B}^{-1}$.

If we denote by $b_{ij}$ the element of the i-th row and the j-th column of $\mathbf{B}^{-1}$, then the $\overline{b_{lj}}$ is the corresponding element of $\overline{\mathbf{B}^{-1}}$. The corresponding $\overline{b_{ij}}$ are given by

$$\overline{b_{ij}} = b_{ij} - \frac{b_{lj}}{x_{lk}} x_{ik} \text{ for } i \neq l$$

$$\overline{b_{lj}} = \frac{b_{lj}}{x_{lk}}.$$

The new solution to the dual problem is $\overline{\mathbf{W}} = \mathbf{W}^0 - \theta \mathbf{B}_l$. The corresponding value of the objective function is $\overline{\mathbf{Wb}} = \mathbf{W}^0\mathbf{b} - \theta x_{l0}$. The new solution to the primal constraints are given by $\overline{\mathbf{X}} = \mathbf{B}^{-1}\mathbf{b}$.

If all $\overline{\mathbf{X}} \geq 0$, the we got the optimal feasible solution to the primal. If not, then there is at least $\overline{x_{l0}} = \overline{\mathbf{B}}\mathbf{b} > 0$. In such case we have to apply the simplex procedure again in order to find out which vector is going to be removed and which one is going to be introduced into the basis. We have to iterate such steps until we will find a basis, which solves the dual problem and is appropriate basis for the primal problem also. If we will find that the dual has an unbounded solution, then there are not any existing feasible solutions to the primal problem. The dual simplex algorithm can be considered as the variation of the simplex procedure, since it uses the information contained in its tableau. The following example illustrates our above discussion of the dual simplex algorithm.

*Example 2.1* Let us consider the problem

Minimize $z = 2x_1 + x_2 + 3x_3$

Subject to
$$\begin{aligned} -x_1 + 2x_2 - x_3 + x_4 &= -5 \\ 2x_1 + x_2 + x_3 + x_5 &= 10 \\ -x_1 + x_3 + x_6 &= 0 \\ x_j \geq 0, j &= 1,\cdots,6 \end{aligned}$$

The dual problem is as it follows:

Maximize $z = -5w_1 + 10w_2$

$$-w_1 + 2w_2 - w_3 \le 2$$
$$2w_1 + w_2 \le 1$$
Subject to $-w_1 + w_2 + w_3 \le 3$
$$w_1 \ge 0$$
$$w_2 \ge 0$$

There is an initial basis $\mathbf{B} = (\mathbf{P}_4\mathbf{P}_5\mathbf{P}_6)$. The primal problem in the usual simplex tableau setting is given as Table 2.1.

Since all $z_j - c_j$ elements are nonpositive, the basis **B** is a feasible solution for the dual, i.e. optimal but not feasible for the primal. Since the initial basis is

$\mathbf{B} = (\mathbf{P}_4\mathbf{P}_5\mathbf{P}_6) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$, the dual solution is

$\mathbf{W}^0 = \mathbf{c}^0\mathbf{B}^{-1} = (0,0,0)\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} = (0,0,0)$. Applying the dual procedure, we observe that $x_l = x_4 = -5$, therefore, $\mathbf{P}_4$ is going to be removed and $\mathbf{P}_1$ is to be introduced into the basis caused by $\theta = \dfrac{z_j - c_j}{z_{41}} = 2$. The second step is presented in Table 2.2.

Since all $x_i \ge 0$ and all $z_j - c_j \le 0$, we have obtained the optimal feasible solution. The primal optimal solution is $x_1 = 5, x_2 = x_3 = 0$ and that for the dual problem is $w_1 = -2, w_2 = 0$ with the common value of 10.

**Table 2.1 Initial Step of Example 2.1**

| | | | | 2 | 1 | 3 | 0 | 0 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| *i* | Basis | *c* | $\mathbf{P}_0$ | $\mathbf{P}_1$ | $\mathbf{P}_2$ | $\mathbf{P}_3$ | $\mathbf{P}_4$ | $\mathbf{P}_5$ | $\mathbf{P}_6$ |
| 1 | $\mathbf{P}_4$ | 0 | -5 | -1 | 2 | -1 | 1 | 0 | 0 |
| 2 | $\mathbf{P}_5$ | 0 | 10 | 2 | 1 | 1 | 0 | 1 | 0 |
| 3 | $\mathbf{P}_6$ | 0 | 0 | -1 | 0 | 1 | 0 | 0 | 1 |
| 4 | | | 0 | -2 | -1 | -3 | 0 | 0 | 0 |

**Table 2.2 Second Step of Example 2.1**

| *i* | Basis | *c* | $\mathbf{P}_0$ | $\mathbf{P}_1$ | $\mathbf{P}_2$ | $\mathbf{P}_3$ | $\mathbf{P}_4$ | $\mathbf{P}_5$ | $\mathbf{P}_6$ |
|---|---|---|---|---|---|---|---|---|---|
| 1 | $\mathbf{P}_1$ | 2 | 5 | 1 | -2 | 1 | -1 | 0 | 0 |
| 2 | $\mathbf{P}_5$ | 0 | 0 | 0 | 5 | -1 | 2 | 1 | 0 |
| 3 | $\mathbf{P}_6$ | 0 | 5 | 0 | -2 | 2 | -1 | 0 | 1 |
| 4 | | | 10 | 0 | -5 | -1 | -2 | 0 | 0 |

# CHAPTER 3

# INTEGER PROGRAMMING

## 3.1 Formulations of an Integer Program

Suppose that we have a linear programming problem in which some or all variables must be required to be nonnegative integers. Such problems are called an *integer programming problems* (*IP*) [24]. If all variables are integers, then such IP is called a *pure IP problem.* A *mixed* IP problem is a problem in which only some variables are required to be integers. If one or all variables are required to be either zero or one, then such problems are called *0-1 IP problems*.

**Pure IP problem** is written as

Maximize $\mathbf{cx}$

Subject to $\mathbf{Ax} \leq \mathbf{b}$
$\mathbf{x} \geq \mathbf{0}$ and integer

where **A** is an m by n matrix, **c** is n-dimensional row vector, **b** is m-dimensional column vector, and **x** is n-dimensional column vector of integer variables.

**Mixed Integer problem** is written as

Maximize $\mathbf{cx} + \mathbf{hy}$

Subject to $\mathbf{Ax} + \mathbf{Gy} \leq \mathbf{b}$
$\mathbf{x} \geq \mathbf{0}, \mathbf{y} \geq \mathbf{0}$ and integer

where **A** is an m by n matrix, **G** is an m by p matrix, **h** is a p-row vector, and **y** is a p-column vector of integer variables.

**0-1 or Binary Integer problem** is written as

Maximize $\mathbf{cx}$

Subject to $\mathbf{Ax} \leq \mathbf{b}$
$X \in \{0,1\}$

We might initially assume that an optimal solution for an IP can be obtained by solving the corresponding Linear Programming problem (the so-called LP relaxation problem can be obtained by leaving out all integer or 0-1 constraints).

Since the feasible region for any IP problem is contained in the feasible region for the LP relaxation problem, we first assume that the rounding off to the nearest integer of each variable gives us the desired integer solution. However such approach is false and the following example will easily illustrate this insufficient approach:

*Example 3.1*

Maximize $z = 11x_1 + 19x_2$

$$\text{Subject to } \begin{array}{l} 8x_1 + 4x_2 \leq 15 \\ x_1, x_2 \geq 0 \\ x_1, x_2 \text{ integer} \end{array} \qquad (3.1)$$

Using the above-mentioned approach we can find the optimal solution to the corresponding LP relaxation: $x_1 = \frac{15}{8}, x_2 = 0$. Rounding the solution yields to $x_1 = 2, x_2 = 0$ as a possible optimal solution to (3.1). Figure 3.1 shows that $x_1 = 2, x_2 = 0$ belong to the infeasible region, and therefore cannot be the optimal solution to (3.1). If we decide to round downward yielding $x_1 = 1, x_2 = 0$, we cannot obtain the optimal solution which is $x_1 = 0, x_2 = 3$.

This example shows that rounding doesn't yield to an optimal integer solution, and other methods required obtaining it. Those methods will be described in the following chapters.

As it was done for the LP problems the systematical approach to construct the formulations for the IP problems must be done in the same manner. The systematical approach allows us to define the variables, use the variables in the set of the constraints, and define the objective function. However, some difficulties can be met. In that case we need to define an additional or an alternative set of variables and then iterate. Below are some examples of the IP problems using the above-mentioned systematical approach towards the variables and objective function.

*Example 3.2* **The Assignment Problem.**

Suppose we have people to carry out n jobs. There is an estimated cost $c_{ij}$ if a person *i* is working for job *j* and each person carries out exactly one job. We have $x_{ij} = 1$ if person *i* does job *j* and $x_{ij} = 0$ otherwise. The corresponding constraint is $\sum_{j=1}^{n} x_{ij} = 1$ for i = 1, …, n, and this means that each person does one job. The next constraint, $\sum_{i=1}^{n} x_{ij} = 1$ for j = 1,…,n, means that each job j is done by one person. The variables are 0-1: $x_{ij} \in \{0,1\}$

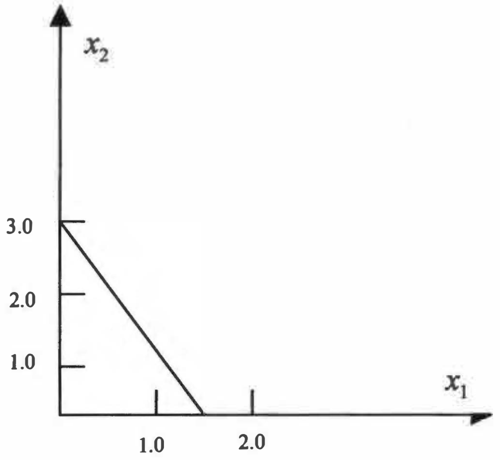

**Figure 3.1 Feasible and Infeasible Region for the IP Example 3.1**

$x_{ij} \in \{0,1\}$ $i = 1,\ldots,$ n and $j = 1, \ldots,$n. The objective function (the cost of assignment) must be minimized:

$$\text{Minimize } \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij}$$

*Example 3.3* **The Knapsack Problem.**

For the given budget *b* there are n projects available for the respective investments. Let us denote by $a_j$ the expenditures to provide the project *j*, and by $c_j$ the anticipated profit. The problem is to find a set of projects in that way that the budget b is not exceeded and the forthcoming profit is maximized. We can choose the variables as $x_j = 1$ if a project *j* is chosen, and $x_j = 0$ otherwise. If the budget *b* cannot be exceeded, then the respective constraint is the following:

$\sum_{j=1}^{n} a_j x_j \le b$, where the variables are 0-1: $x_j \in \{0,1\}$, j = 1, …,n.

Since the profit is maximized, the corresponding expression represents the objective function: Maximize $\sum_{j=1}^{n} c_j x_j$

*Example 3.4* **The Traveling Salesman Problem.**

The problem is the following: the salesman travels starting at the 1st city and then visiting *n* cities exactly once. Afterwards, he returns to the starting city. Let us denote by $c_{ij}$ the time that would be taken to travel from city *i* to city *j*.

There are the following variables as $x_{ij} = 1$ when the salesman goes from city *i* to city *j* and $x_{ij} = 0$ otherwise. We can see that $x_{ii}$ is undefined since each route implies the straight direction from city *i* to city *j* excluding, therefore, backward direction. When the salesman leaves the city *i*, then the situation can be described by the following constraint: $\sum_{j,j\neq i} x_{ij} = 1$ for i = 1,…,n.

When the salesman arrives at a city *j*, this occurs exactly once with the corresponding constraint: $\sum_{i,i\neq j} x_{ij}$ for j = 1,…,n.

By imposing the constraints that the salesman must pass from one set of the cities to another, we obtain the so-called cut-set constraints:

$\sum_{i\in s}\sum_{j\in s} x_{ij} \geq 1, s \subset N, s \neq \phi$, where N is the generic set {1,2,…,n}.

The objective function represented by the total travel time that is supposed to be minimized: Minimize $\sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij}$

### 3.2 Mixed Integer Formulations

The following examples give us the IP problems with mixed variables.

*Example 3.5*

Suppose we have the so-called fixed charge cost function: h(x) = f + p(x), if 0 < x ≤c and h(x) = 0 if x = 0.

Figure 3.2 describes this function as a linear function with the y-intercept as f for the given interval (0,c).

For the purpose of a problem we can define an additional variable $y = 1$ if $x \geq 0$ and $y = 0$ otherwise. This will add the constraints: $x \leq cy, y \in \{0,1\}$ and allow us to replace h(x) by fy + px as an objective function.

*Example 3.6* **Incapacitated Facility Location.**

For the given *n* depots (N = {1,…,n}) and *m* clients (M = {1,…,m}) there is a fixed cost $f_i$ associated with the use of depot *j* and a transportation $c_{ij}$. The order *i* is delivered from depot *j*. If depot *j* is used, then the depot opening variable is $y_j = 1$ and $y_j = 0$ otherwise. The satisfaction of the client *i* is expressed in the following constraint: $\sum_{j=1}^{n} x_{ij} = 1$ for i = 1,…,m and $\sum_{i\in M} x_{ij} \leq m$.

The link between the $x_{ij}$ and $y_j$ is the next constraint: $\sum_{i\in M} x_{ij} \leq y_j, y_j \in \{0,1\}$for $j \in N$.

Since $h_j(x_{1j},\cdots,x_{mj})$ in which $h_j(x_{1j},\cdots,x_{mj}) = f_j + \sum_{i\in M} c_{ij}x_j$, we have the following objective function:

$$\text{Minimize } \sum_{i\in M}\sum_{j\in N} c_{ij}x_{ij} + \sum_{j\in N} f_j y_j, \text{where} \sum_{i\in M} x_{ij} > 0\,.$$

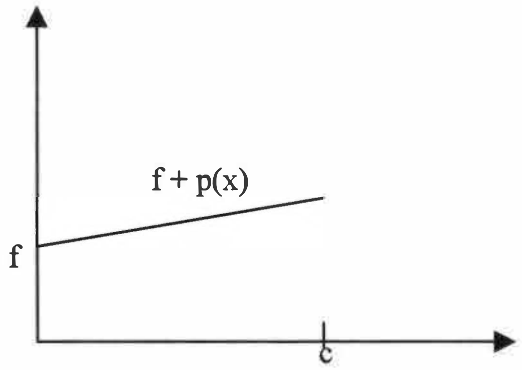

**Figure 3.2 The Fixed Cost Function**

### 3.3 Alternative Formulations

Before giving the alternative formulations for the IP problems, there are the following definitions for the LP problems that we are going to use for the corresponding formulations for the IP problems.

*Definition 3.1* The linear constraints $P = \{\mathbf{x} \in R^n : \mathbf{Ax} \le \mathbf{b}\}$ is a polyhedron, where $R^n$ is the set of n-dimensional real numbers. Applying the definition of polyhedron to the IP problems, we have the next following definition for the constraints for IP problems:

*Definition 3.2* A polyhedron $P \subseteq R^{n+p}$ is a formulation for a set $X \subseteq Z^n \times R^n$, where $X = P \cap (Z^n \times R^p)$.

The following example and the corresponding Figure 3.3 illustrate two different formulations for the IP problems.

*Example 3.7*

Suppose we have the set $X = \{(1,2),(2,2),(3,2),(4,2),(2,3),(3,3),(4,3),(2,4),(3,4)\}$. The illustration of this set with two different formulations is shown on Figure 3.3

There is the following proposition based on the Definition 3.1:

*Proposition 3.1* A polyhedron P is a convex hull of X, denoted by $\text{conv}(X) = \left\{x : x = \sum_{i=1}^{n} \lambda_i x^i, \sum_{i=1}^{n} \lambda_i = 1, \lambda_i \ge 0, i = 1, \cdots, n\right\}$ over the finite subset $\{x^1, \cdots, x^n\}$ of X.

Having this proposition in mind we would ask ourselves the following question: For the given two formulations $P_1$ and $P_2$, how can we say that one is better than the other? The next definition gives an answer for this important question since the import of this is to get tighter formulation for the IP problem.

Since $X \subseteq \text{conv}(X) \subseteq P$ we have the following definition:

*Definition 3.3* For the given two formulations $P_1$ and $P_2$, $P_1$ is better formulation than $P_2$ if $P_1 \subset P_2$.

Next example will show us the way of choosing the better formulation for the IP problem.

*Example 3.8* **Choosing better formulation for a Knapsack problem**

Suppose we have the set of X:

$X = \{(0,0,0,0),(1,0,0,0),(0,1,0,0),(0,0,1,0),(0,0,0,1),(0,1,0,1),(0,0,1,1)\}$

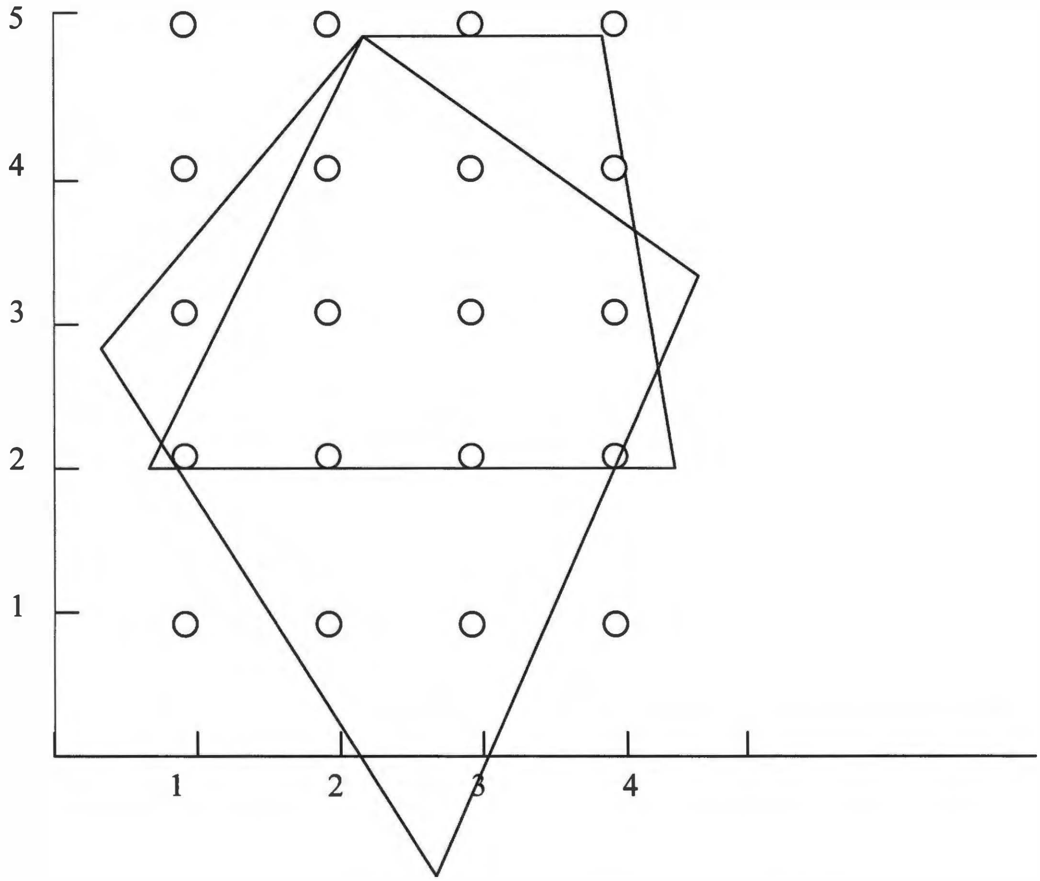

**Figure 3.3 Two Different Formulations for IP**

There are three formulations:

$$P_1 = \{x \in R^4 : 0 \le x \le 1, 75x_1 + 21x_2 + 33x_3 + 25x_4 \le 90\}$$

$$P_2 = \{x \in R^4 : 0 \le x \le 1, 2x_1 + 3x_2 + 2x_3 + x_4 \le 4\}$$

$$P_3 = \left\{\begin{array}{rl} x \in R^4 : 0 \le x \le 1, & 2x_1 + 3x_2 + 2x_3 + 1x_4 \le 4, \\ & 1x_1 + 1x_2 + 1x_3 + 1x_4 \le 1, \\ & 1x_1 \phantom{+ 1x_2} + 1x_3 \phantom{+ 1x_4} \le 1 \end{array}\right\}$$

We can easily verify that $P_3 \subset P_2 \subset P_1$ and $P_3 = \text{conv}(X)$. So, as a result $P_3$ is a better formulation.

# CHAPTER 4

# OPTIMALITY AND RELAXATION

## 4.1 Optimality Conditions and Relaxations

Suppose we have an IP problem $z = \max\{c(x)\colon x \in X \subseteq Z^n\}$. We are seeking the optimal solutions and the question arising here is the following: How will this optimality provide necessary criteria to stop an algorithm for an IP? The answer refers us to find a lower bound $\underline{z} \leq z$ and an upper bound $\overline{z} \geq z$, such that $\underline{z} = \overline{z} = z$. This means that any algorithm will seek a decreasing sequence of the upper bounds $\overline{z_1} > \overline{z_2} > \cdots > \overline{z_p} \geq z$ and an increasing sequence of the lower bounds $\underline{z_1} < \underline{z_2} < \cdots < \underline{z_h} \leq z$. Then the algorithm stops when $\overline{z_p} - \underline{z_h} < \varepsilon$, where ε is the small nonnegative value. Finding upper and lower bounds represents some approach called the "relaxation". The idea behind the relaxation is to replace the max (min) IP by a simpler one, where an optimal value is at least as large (small) as z.

There are two possibilities:

i. We need to enlarge the set of feasible solutions so that the optimization process occurs over a larger set, or,

ii. Substitute the max (min) objective function by a function that has the same or a larger (smaller) value in a given region.

There is a corresponding definition for a relaxation:

*Definition 4.1* A problem $z^r = \max\{f(x)\colon X \in T \subseteq R^n\}$ is a relaxation of

an IP $z^i = \max\{c(x)\colon x \in X \subseteq R^n\}$ if

i. $X \subseteq T$

ii. $f(x) \geq c(x)$ for all $x \in X$.

From the above definition follows the proposition:

*Proposition 4.1* If $z^r$ is a relaxation of an IP, then $z^r \geq z^i$.

Proof: Suppose that x* is an optimal solution of an IP. Therefore, if

$x^* \in X \subseteq T$ and $z^i = c(x^*) \leq f(x^*)$, and, since $x^* \in T$, $f(x^*)$ is a lower bound on $z^r$, and $z^i \leq f(x^*) \leq z^r$.

Our attention directed us to the imposed question of how to obtain the desired relaxations. The answer based on the so-called linear programming relaxation that gives us better formulations (constraints) with the objective function unchanged.

### 4.2 Linear Programming Relaxations

There is the following definition of the relaxation:

*Definition 4.2* The linear program $z^r = \max\ \{c(x): x \in P\}$, where the formulation $P = \{x \in R_t^n : Ax \leq b\}$, is the linear programming relaxation for the integer program $z^i = \max\ \{c(x): x \in P \cap Z^n\}$, where $R_t^n$ is the n-dimensional nonnegative real numbers, $Z^n$ is the n-dimensional nonnegative integers.

Since $P \cap Z^n$ and the objective function are unchanged, this is clearly a relaxation to IP. To illustrate such approach, let us consider the following example:

*Example 4.1*

Maximize $z = 3x_1 - x_2$

$$\text{Subject to} \quad \begin{array}{l} 6x_1 - 2x_2 \leq 15 \\ x_2 \leq 5 \\ 2x_1 - 2x_2 \leq 3 \\ x \in Z_t^2 \end{array}$$

To get a primal or lower bound, notice that (2,1) is a feasible solution, so the lower bound is $z \geq 5$. To obtain an upper bound, we have to consider the linear programming relaxation. The optimal solution is $x^* = \left(\frac{25}{6}, 5\right)$ with value $z^r = \frac{15}{2}$. Therefore, we obtain an upper bound $z \leq \frac{15}{2}$. Observe that the optimal value must be an integer. So, we can round down to the nearest integer $z \leq 7$.

Note that the better formulations give us the tighter dual (lower and upper) bounds. In the perspective of the better formulation we have the following proposition:

*Proposition 4.2* Let $P_1$, $P_2$ be the formulations for an IP $\max\{cx : x \in X \subseteq Z^n\}$ with $P_1$ a better formulation than $P_2$, $P_1 \subset P_2$. If $z_i^r = \max\{cx : x \in P_i\}, i = 1,2$ are the values of the linear programming relaxations, then $z_1^r \leq z_2^r$ for all c.

Observe that the relaxations do not just give us the dual bounds. The relaxations by themselves give us the possibility to prove the optimality, and the following proposition supports above-mentioned statement:

*Proposition 4.3* If x* is an optimal solution to the relaxation, $x^* \in X$, and $f(x^*) = c(x^*)$, then x* is an optimal solution of an IP.

Proof: Since $x^* \in X$, $z \geq c(x^*) = f(x^*) = z^r$, and $z \leq z^r$ we get $c(x^*) = z = z^r$

The following example illustrates the considered proposition:

*Example 4.2* Let us consider the IP

Maximize $6x_1 + 5x_2 + 5x_3 + 3x_4$

Subject to $\begin{array}{l} 2x_1 + 2x_2 + 3x_3 + 4x_4 \leq 4 \\ x \in B^4 \end{array}$, $B^4$: {0,1} the set of 4-dimensional 0,1 vectors.

The relaxation has an optimal solution x* = (1,1,0,0). Since x* is integral, it solves the integer program.

### 4.3 Methods of Integer Programming

Generally, there are two types of methods for solving IP problems. The first type is generated by the idea that the solution space contains a finite number of integer points. So, it can be considered the region, where we seek all integer points for our optimal solution. This means that we search for those points by simple exhaustive enumeration. Search methods basically include enumeration methods as well as Branch-and-Bound methods. These methods can be applied for the zero-one IP problems [11].

Cutting methods are mostly suited for the mixed and pure IP problems. These methods imply the fact that the solution to the linear program occurs at the extreme point. The basic idea is to add the new appropriate constraints, which satisfy for the feasible integer region. The successive iteration, eventually, leads us to the optimal convex region with the optimal extreme point integral solution.

The Branch-and-Bound algorithms generate the new subproblems from the current successive LP solution or the so-called continuous solution. These sub problems generated with the bounds of the chosen variables and the corresponding objective function will restrict the total number of the continuous problems, which we must solve [23].

In order to illustrate different approaches to solve an IP problem we consider the following example:

*Example 4.3*

Maximize $z = 9x_1 + 6x_2$

$$\text{Subject to } \begin{array}{l} x_1 + x_2 \leq 7 \\ 8x_1 + 5x_2 \leq 43 \\ x_1, x_2 > 0 \text{ and integer} \end{array}$$

The Figure 4.1 illustrates the feasible region for the LP continuous problem with the optimal extreme point $x_1 = \frac{8}{3}$ and $x_2 = \frac{13}{3}$. Since the optimal solution is far from the integral solution requirements, we need to apply the new cutting constraint. In our case the equation $2x_1 + 5x_2 = 29$ is such constraint that "cuts off" the part of the feasible LP region. Such cuts can be added as much as we need them in order to find the optimal IP solution.

Figure 4.2 illustrates the Branch-and-Bound approach to solve an IP problem. The original LP problem has the optimal convex solution space regarded as the optimal solution space for the subproblems. Adding constraints $x_1 = 2$ and $x_2 = 3$ we obtained the regions for subproblems 2 and 3, where we need to continue our search for the optimal integral solution. Using the idea of Branch-and-Bound method, we can subdivide further the regions of subproblems 2 and 3 respectively in order to find an optimal solution for the IP problem.

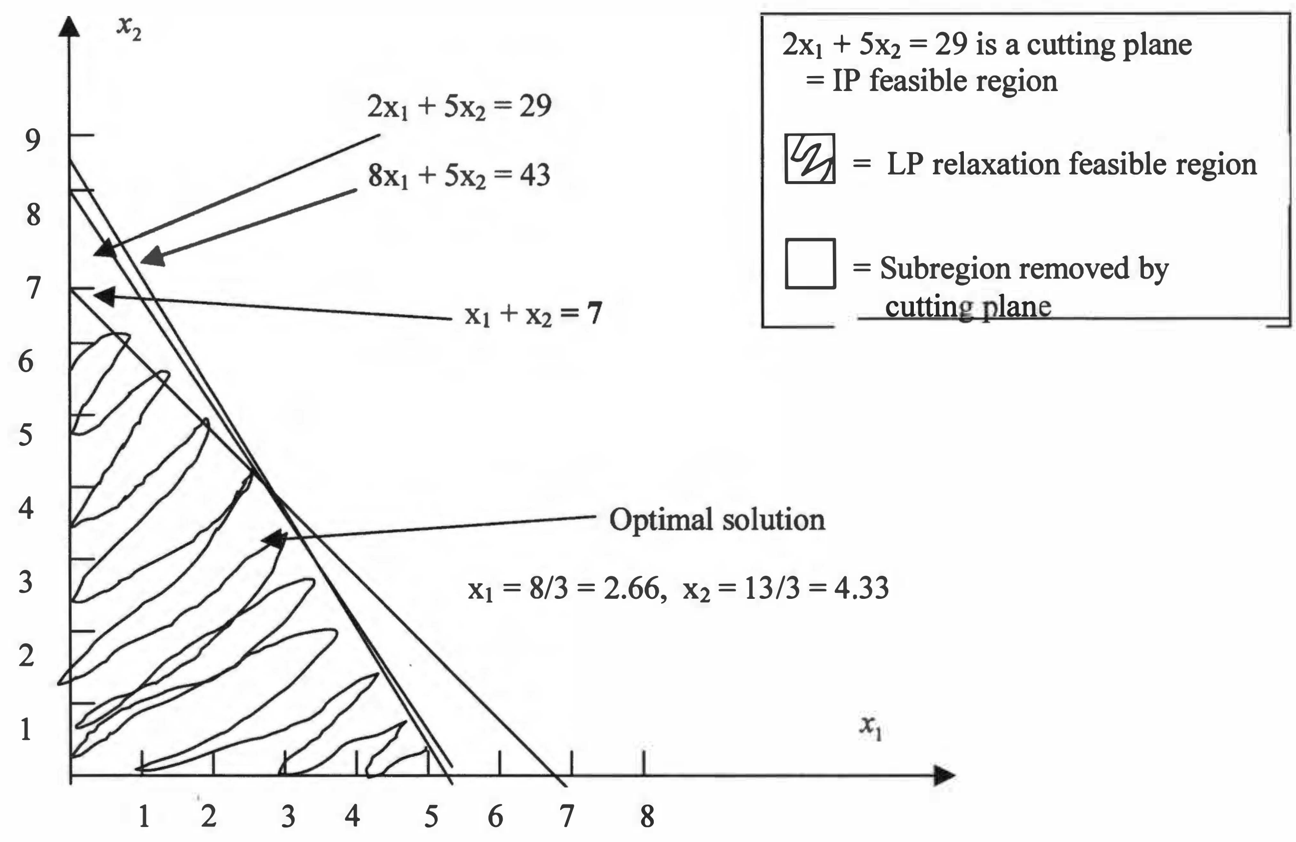

**Figure 4-1 The Illustration of Cutting Plane Method**

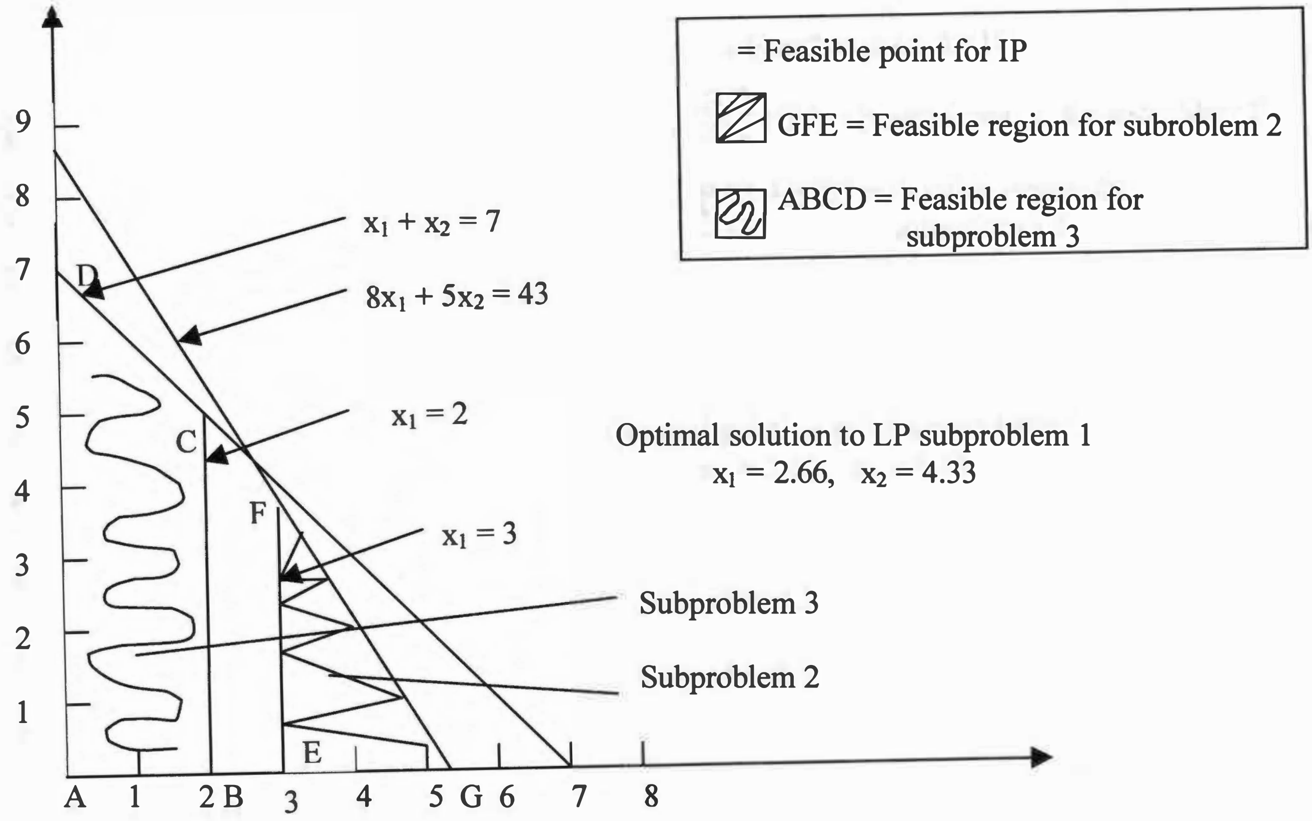
= Feasible point for IP
GFE = Feasible region for subroblem 2
ABCD = Feasible region for subproblem 3
$x_1 + x_2 = 7$
$8x_1 + 5x_2 = 43$
$x_1 = 2$
$x_1 = 3$
Optimal solution to LP subproblem 1
$x_1 = 2.66, \quad x_2 = 4.33$
Subproblem 3
Subproblem 2
D
C
F
E
9
8
7
6
5
4
3
2
1
A 1 2 B 3 4 5 G 6 7 8

**Figure 4-2 The Illustration of Branch-and-Bound Method**

# CHAPTER 5

# BRANCH-AND-BOUND ALGORITHM

## 5.1 The Concepts of the Branch-and-Bound Method

The basic concepts of the Branch-and-Bound algorithm for integer programming using the so-called linear programming relaxation, or simpler, LP-relaxation further in the text, was elaborated first by Land and Doig during 1960's [17]. The term Branch-and-Bound was used earlier by Little and others in their research work to solve the traveling salesman problem [18].

Since the solution space of an IP problem is confined, the number of integer points that must be taken into account is finite. The mainstream idea is to enumerate such points in order to solve an IP problem. As soon as enumeration is accomplished, then the optimum solution associated with the optimal value of an objective function [12]. Bertie and Roy (1965) were the first to present a general theory of the Branch-and-Bound method [3]. In the next stage of the development of the Branch-and-Bound theory Balas stated the Branch-and-Bound concepts in a simpler form [1]. Mitten (1970) [20] attempted to extend Balas' work and accomplished it in the form of generalized theory. It must be noted that the contemporary works and the theories were based on the ideas presented by both Balas and Mitten.

We can describe the Branch-and-Bound method as it follows. Let the objective function z be defined on the subset S of the set R of the real numbers:

$$z : S \rightarrow R$$

The order of the elements of S is given as $s_i \in S$. The problem is to find an element $s^* \in S$ that maximizes the return function z:

$$z(s^*) = \max \{z(s_i) \; s_i \in S\}$$

If the number of the elements is large enough, than the above-mentioned exhaustive enumeration is permitted. That means that the Branch-and-Bound principle proved to be useful.

There are existing three basic parts of the Branch-and-Bound method that provide the effectiveness to use this algorithm:

i. There exists a subset of T of the set of S: $T \supset S$ with the objective function $u: T \rightarrow R$. This function u extends z in such way that $t_j \in S$, $u(t_j) = z(t_j)$, where $t_j \in T$. The equality of two return functions u and z refers to the easier computation than with those associated with z.

ii. We need to define a branching as a rule to generate a family of subsets $\{T_i^k\}$ from T such that $F(T^k)=\{T_1^k,\cdots,T_q^k\}$ , where it is assumed that $|T^k|\geq z$ and $\bigcup_{i=1}^{q} T_i^k = T^k - \{t_{jk}\}$. The element $t_{jk}$ is defined as $u^k = u(t_{jk}) = \max\{u(t_j)\ \ t_j \in T^k\}$. The basic rule to define a branching F depends on the selection of T and u.

iii. To define a bounding rule we need to select T, u and F in a manner such that an upper bound of the objective function for every $t_j \in T^k$ can be determined for each $T^k \subset T$. Note that the upper bound is defined above.

By adding the lower bound on the objective value for every $s_j \in S$ we can enhance the efficiency of the computation. In order to determine the lower bound we need to define a subset $T^k$ generated by F as an active and let $A \subset T$ be the family of all active subsets $T^k$. The lower bound $\underline{z}$ for every $S_j \in S$ is determined as the value of u associated with any $t_j \in S$. Using the lower bound we can cancel all those active subsets that do not generate the objective value $u^k$ exceeded $\underline{z}$. It can be stated as

$\bar{u} = \max\{u^k | T^k \text{active}\}$, in which a subset $T^k$ is memorized if and only if

$$\underline{z} \leq u^k \leq \bar{u}.$$

These were the important aspects of almost all Branch-and-Bound algorithms.

## 5.2 LP-based Branch-and-Bound Method

Let us denote the feasible set for the IP-problem by $S = \{x: Ax = b, x \geq o \text{ and integer}\}$. The corresponding LP-relaxation feasible set is denoted by $T = \{x: Ax = b, x \geq 0\}$, where $S \subseteq T$. Let $S^k$ be the intersection of S. The set of points of S is satisfying the constraints given by the edges of the path $P_k$ connecting node 0 to node k.

Suppose that enumeration brought us to a node in the tree consisting of the path $P_k$. Thus we have the following problem at node i:

$$z_{lp}^i = \min\{cx : x \in S^i\}. \qquad (5.1)$$

Solving the LP-relaxation problem we can find the lower bound as

$$\underline{z}^i = z_{lp}^i = \min\{cx : x \in T_i \supseteq S\} \qquad (5.2)$$

If the solution $x^i$ at node i is not all-integer then the partition of $S^i$ is the following:

$$\{S^i \cap \{x : x \leq \lfloor x_k \rfloor\}\}, \{S^i \cap \{x : x \geq \lceil x_k \rceil\}\} \qquad (5.3)$$

where the basic variable is represented as $x_k = \lfloor x_k \rfloor + f_k, 0 < f_k < 1$. The $\lfloor x_k \rfloor$ denotes the largest integer less than or equal to $x_k$ and $\lceil x_k \rceil$ denotes the smallest integer greater or equal to $x_k$.

Suppose that an upper bound $u_j$ is known at node i for each variable $x_j$, j = 1,...n:

$$S^i = \{x : Ax = b, 0 \le \alpha^i_j \le x_j \le \beta^i_j \le u_j, x \text{integer}\}, \qquad (5.4)$$

where $\alpha^i_j$ and $\beta^i_j$ are the integrals. These integrals can be determined using partition (5.3).

This partition results that we have to add the new constraints, which are the lower and upper bounds. Then the upcoming problem must be solved either by relaxation or the dual simplex algorithm [24]. Since in the context of the Branch-and-Bound method we used the LP-relaxation, let us introduce the Branch-and-Bound algorithm in the step-by-step way:

**Step 1.** Initial problem. Start at a node i = 0, where $S^0$ is given in (5.4), $\underline{z}^0 = -\infty$ and $\overline{z}^0 = \infty$. Go to Step 2.

**Step 2.** (Branching): If there are no active nodes, go to Step 7, otherwise select an active node i. Solve the LP (5.2) and go to Step 3, otherwise go to Step 4.

**Step 3.** (Partitioning): Choose a variable $x_k$ with $f_k > 0$ and subdivide $S^i$ as in (5.3). Then, go to Step 2.

**Step 4.** (Solving LP-relaxation): If the LP-relaxation solution is not feasible, prune a node i, go to Step 2. If the LP-relaxation has an optimal solution $x^i$, let $\underline{z}^i = \lceil z^i_{lp} \rceil$, then go to Step 5.

**Step 5.** (Pruning by Optimality): If $x^i$ is not all integer, go to Step 6; otherwise select $\overline{z}^i = z^i$ and prune a node i. Let $\overline{z}^0 = \min\{\overline{z}^0, \overline{z}^i\}$ and go to Step 6.

**Step 6.** (Pruning by Bounds): Prune any node i, where $\underline{z}^i \ge \overline{z}^0$ and go to Step 2.

**Step 7.** (Termination): End. If $\overline{z}^0 = \infty$ then there is no feasible region. If $\overline{z}^0 < \infty$ then the feasible solution is optimal [2].

The following example illustrates the application of the presented algorithm:

*Example 5.1*

Maximize $z = 3x_1 - x_2$

$$\text{Subject to} \quad \begin{array}{l} 7x_1 - 2x_2 \leq 15 \\ x_2 \leq 4 \\ 2x_1 - 2x_2 \leq 3 \\ X \in Z_t^2 \end{array}$$

*Bounding.* We need to obtain the first upper bound. In that purpose we add the slack variables $x_3, x_4, x_5$. The result of this gives the optimal basis, we was obtained by solving the corresponding LP-relaxation problem with the integrality constraints' requirement that had been dropped. The optimal basis representation from Table 5.1 is $x_1 = \frac{24}{10}, x_2 = \frac{9}{10}, x_4 = \frac{31}{10}, z = \frac{63}{10}$.

Table 5.1 shows that we obtained an upper bound $z = \frac{63}{10}$. The nonintegral solution is $\left(\overline{x_1}, x_2\right) = \left(\frac{24}{10}, \frac{9}{10}\right)$. Since the feasible solution wasn't obtained, we assumed that the lower bound $\underline{z} = -\infty$ by convention.

*Branching.* Since $\underline{z} < \overline{z}$, we need to partition or branch. We mean by partition to split the feasible region. The branching idea is to choose the respective variable that is basic and fractional in the LP-relaxation solution, and as soon as such variable is chosen, we need to divide the problem into two subproblems. Since $x_j = \overline{x_j} \notin Z^1$, we can take the partition (5.3). It is obvious that $S = S_1 \cup S_2$ and $S_1 \cap S_2 = \varnothing$. Moreover, the solution $\overline{x}$ of LP(S) is not feasible for both LP(S₁) and LP(S₂). That implies $\max\left\{\overline{z_1}, \overline{z_2}\right\} < \overline{z}$ so the upper bound strictly decreases.

In Figure 5.1 there is a partial Branch-and-Bound tree 1.

Since the upper bound decreases and by partitioning the problem S into S₁ and S₂, we take the first fractional variable $\overline{x_1} = \frac{24}{10} \notin Z^1$. Then we are partitioning as $S_1 = S \cap \{x : x_1 \leq 2\}$ and $S_2 = S \cap \{x : x_1 \geq 3\}$.

**Table 5.1 The Simplex Tableau Representation for Example 5.1**

| | $x_1$ | $x_2$ | $x_3$ | $x_4$ | $x_5$ | |
|---|---|---|---|---|---|---|
| | 0 | 1 | $\frac{1}{5}$ | 0 | $-\frac{7}{10}$ | $\frac{9}{10}$ |
| | 0 | 0 | $-\frac{1}{5}$ | 1 | $\frac{7}{10}$ | $\frac{31}{10}$ |
| | 1 | 0 | $\frac{1}{5}$ | 0 | $-\frac{1}{5}$ | $\frac{24}{10}$ |
| | 0 | 0 | $\frac{2}{5}$ | 0 | $\frac{1}{10}$ | $\frac{63}{10}$ |

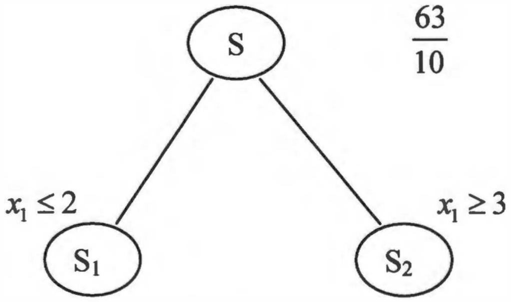

**Figure 5.1 Partial Branch-and-Bound Tree 1**

Note that the subproblems, or further the nodes of the branching tree, that must be examined, are said to be active nodes.

*Choosing a Node*. Now we have two active nodes S$_1$ and S$_2$ in the list of the active nodes.

*Reoptimizing*. Since our previous optimal basis remains dual feasible, it is reasonable to reoptimize the last basis using the dual simplex algorithm. Applying the dual algorithm to the LP(S$_1$) we can add the new constraint $x_1 \le 2$, or by adding the slack variables we obtain the equality constraint in the following form:

$$x_1 + s = 2$$

The final optimal basis for LP(S) is the following:

Maximize $\overline{z_1} = \frac{63}{10} - \frac{2}{5}x_3 - \frac{1}{10}x_5$

Subject to
$$x_1 + \frac{1}{5}x_3 - \frac{1}{5}x_5 = \frac{24}{10}$$
$$x_2 + \frac{1}{5}x_3 - \frac{7}{10}x_5 = \frac{9}{10}$$
$$-\frac{1}{5}x_3 + x_4 + \frac{7}{10}x_5 = \frac{31}{10}$$
$$x_1, x_2, x_3, x_4, x_5, s \ge 0$$

Since $x_1 = -\frac{1}{5}x_3 + \frac{1}{5}x_5 + \frac{24}{10}$ we obtain the new constraint in terms of the nonbasic variables as it follows:

$$-\frac{1}{5}x_3 + \frac{1}{5}x_5 + \frac{24}{10} + s = 2$$

or

$$-\frac{1}{5}x_3 + \frac{1}{5}x_5 + s = -\frac{4}{10}$$

Thus, we have the following dual feasible representation:

Maximize $\overline{z_1} = \frac{63}{10} - \frac{2}{5}x_3 - \frac{1}{10}x_5$

Subject to

$$x_1+\frac{1}{5}x_3-\frac{1}{5}x_5=\frac{24}{10}$$
$$x_2+\frac{1}{5}x_3-\frac{7}{10}x_5=\frac{9}{10}$$
$$-\frac{1}{5}x_3+x_4+\frac{7}{10}x_5=\frac{31}{10}$$
$$-\frac{1}{5}x_3+\frac{1}{5}x_5+s=-\frac{4}{10}$$
$$x_1,x_2,x_3,x_4,x_5,s\geq 0$$

1) The first constraint as mentioned above is the following

$$x_1+s=2$$

2) After some algebraic transformations were performed we obtained other constraints in the following form:

$$x_2+x_4=4$$
$$x_2-\frac{1}{2}x_5+s=\frac{1}{2}$$
$$x_3-x_5-5s=11$$
$$x_4+\frac{1}{2}x_5-s=\frac{35}{10}$$

The new reoptimized linear program is shown below:

Maximize $\overline{z}=\frac{11}{2}-\frac{1}{2}x_5-2s$

Subject to

$$x_1+s=2$$
$$x_2-\frac{1}{2}x_5+s=\frac{1}{2}$$
$$x_3-x_5-5s=11$$
$$x_4+\frac{1}{2}x_5-s=\frac{35}{10}$$

With $\overline{z_1}=\frac{11}{2}$ and $\left(\overline{x_1^1},x_2^1\right)=\left(2,\frac{1}{2}\right)$

*Branching.* S cannot be pruned, so use the previous branching tool. It allows us to create two new nodes $S_{11} = S_1 \cap \{x : x_2 = 0\}$ and $S_{12} = S_1 \cap \{x : x_2 \geq 1\}$ and add them to the list of active nodes. The current tree is shown in Figure 5.2.

*Choosing a Node.* Now the list of active nodes contains $S_2$, $S_{11}$, $S_{12}$. Let us choose, arbitrarily the node $S_2$ and examine it in more detailed fashion.

*Reoptimizing.* By solving LP($S_2$) we have the same approach as it was described previously by using the dual simplex algorithm. The constraint $x_1 \geq 3$ gives us the equality as it follows for t ≥ 0:

$$x_1 - t = 3$$

Expressing $x_1$ in terms of the nonbasic variables we obtain:

$$x_1 + \frac{1}{5}x_3 - \frac{1}{5}x_5 = \frac{24}{10}$$

or

$$x_1 = -\frac{1}{5}x_3 + \frac{1}{5}x_5 + \frac{24}{10}$$

Since $x_1 - t = 3$ we have

$$3 + t + \frac{1}{5}x_3 - \frac{1}{5}x_5 = \frac{24}{10}$$

or

$$\frac{1}{5}x_3 - \frac{1}{5}x_5 + t = -\frac{6}{10}$$

Reviewing the constraint, we obtain the resulting linear program, which appeared to be infeasible, since $\overline{z_2} = -\infty$ and, hence, the node $S_2$ is pruned by infeasibility:

$$\text{Maximize } \overline{z_2} = \frac{63}{10} - \frac{2}{5}x_3 - \frac{1}{10}x_5$$

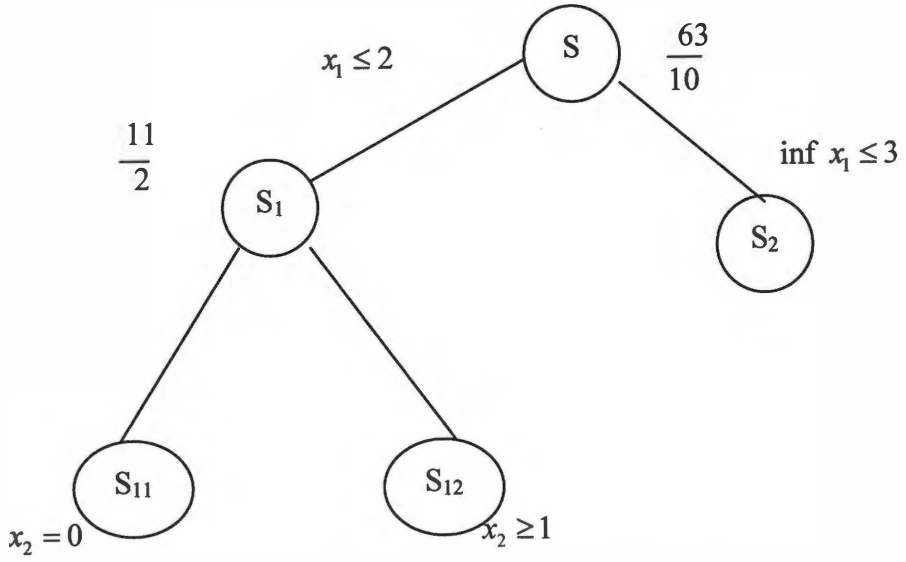

**Figure 5.2 Partial Branch-and-Bound Tree 2**

$$x_1+\frac{1}{5}x_3-\frac{1}{5}x_5=\frac{24}{10}$$

$$x_2+\frac{1}{5}x_3-\frac{7}{10}x_5=\frac{9}{10}$$

Subject to $-\frac{1}{5}x_3+x_4+\frac{7}{10}x_5=\frac{31}{10}$

$$\frac{1}{5}x_3-\frac{1}{5}x_5+t=-\frac{6}{10}$$

$$x_1,x_2,x_3,x_4,x_5,t\geq 0$$

*Choosing a Node.* The node list now consists of $S_{11}$, $S_{12}$. Arbitrarily choosing node $S_{12}$ we remove it from the list.

*Reoptimizing.* $S_{12}=S\cap\{x:x_1\leq 2,x_2\geq 1\}$. The optimal solution is $\overline{x^{12}}=(2,1)$ with the value 5. So, as $\overline{x^{12}}$ is an integer, $z^{12}=5$.

*Updating the Incumbent.* Since LP($S_{12}$) solution is integral, we update the best feasible solution as max {$\underline{z}$, 5} → $\underline{z}$ and store the corresponding solution (2,1). So, now $S_{12}$ is pruned by optimality.

*Choosing a Node.* The node list now consists of $S_{11}$.

*Reoptimizing.* $S_{11}=S\cap\{x:x_1\leq 2,x_2\leq 0\}$. The resulting linear program has the optimal solution $\overline{x^{11}}=\left(\frac{3}{2},0\right)$ with the value 4.5. As $\underline{z}=5>\overline{z_{11}}=4.5$, the node is pruned by bound.

*Choosing a Node*: As the list is empty, the algorithm terminates. The incumbent solution x = (2,1) with value 5 as optimal. Figure 5.3 shows the complete Branch-and-Bound tree.

The figure 5.4 shows the division of the feasible region by using the Branch-and-Bound strategy.

### 5.3 The Application of the Branch-and-Bound Method

As mentioned above we have the following four cases by solving the corresponding LP-relaxation problem:

**Case 1.** The LP has no feasible solution. As a result, there is no feasible solution for the current IP respectively.

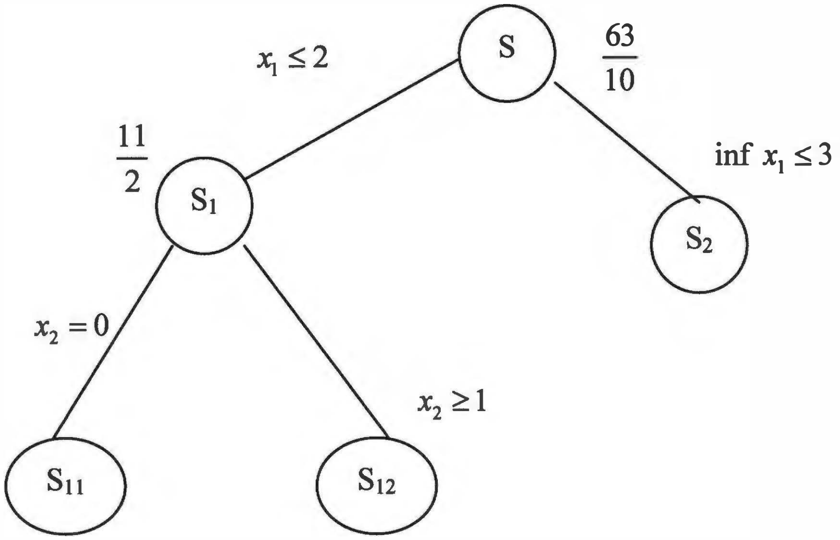

**Figure 5.3 Complete Branch-and-Bound Tree**

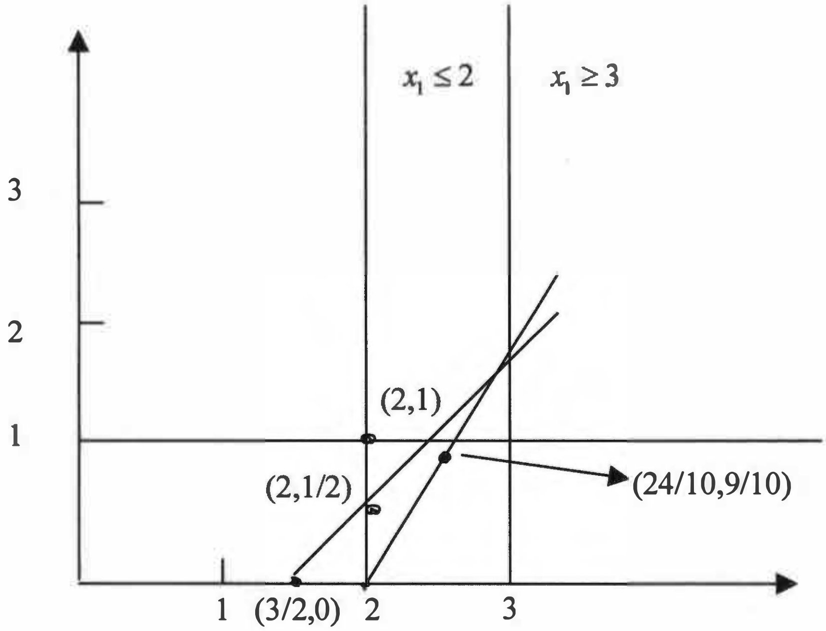

**Figure 5.4 Division of the Feasible Region**

**Case 2.** The LP relaxation has an optimal solution $z_{lp} \geq z^*$. The current IP solution $z_{ip} \geq z_{lp} \geq z^*$, and, therefore, we cannot improve over the incumbent variable.

**Case 3.** The LP-solution is an integral and feasible, and it yields $z_{lp} \leq z^*$. Therefore, the solution is optimal for the current IP, and $z^*$ is reset to $z_{lp}$.

**Case 4.** None of the above cases occur; the optimal LP-solution exists, and $z_{lp} < z^*$, but is not an integer. Therefore, the LP-solution is not feasible for the current IP.

The first three cases provide the condition that the current node is fathomed or implicitly enumerated. The Case 1 means that the node is fathomed by infeasibility. The Case 2 refers to the node to be fathomed by bound, and the Case 3 relates to the node to be fathomed optimality. If a problem is fathomed by Case 3 then we obtain the useful information for the incumbent to be updated. If the problem is not fathomed, and it winds up, then we need further application of the branching. Since the node is not fathomed with the corresponding constraint set that has not been partitioned, then such node is called to be active, as was mentioned previously. If there no active remaining nodes, then the enumeration ends []. Next example illustrates all above-mentioned cases with further exploration of the Branch-and-Bound method.

*Example 5.2* Solve the following problem in terms of integer using Branch-and-Bound algorithm.

Minimize $z = x_1 + 8x_2$

$$\begin{aligned} & 2x_1 + 3x_2 \geq 6 \\ & -3x_1 + x_2 \leq 1 \\ \text{Subject to } & x_1 + x_2 \leq 7 \\ & x_1 - 3x_2 \leq 1 \\ & x_1, x_2 \geq 0 \, \text{and integer} \end{aligned}$$

1. Solve the corresponding LP: Step 4 of the algorithm Branch-and-Bound. The optimal solution is $z_{lp}^0 = \frac{53}{9}$. Set $\underline{z_0} = 6$. The decision vector $x^0\left(\frac{7}{3}, \frac{4}{9}\right)$ is not all integer.

2. Go to Step 3 of the Branch-and-Bound algorithm: Choose $x_1$ for partitioning. Node 1 is associated with $x_1 \leq 2$ and node 2 is associated with $x_1 \geq 3$

3. Solving the augmented LP at node 2 gives

$$z_{lp}^2 = \frac{25}{3} \text{ and}$$

$$x^2 = \left(3, \frac{2}{3}\right)$$

There is no improvement in the lower bound of the objective function $\underline{z}^2 = \left\lceil z_{lp}^2 \right\rceil = 9$.

4. Choose $x_2$ for partitioning: Node 3 is associated with $x_2 \leq 0$ and

node 4 is associated with $x_2 \geq 1$.

5. Solving the LP at node 4 gives all-integer $x^4 = (3,1)$ with $z_{lp}^4 = 11$.

6. Set $\overline{z^0} = 11$ at step 5 of the Branch-and-Bound algorithm: Fathom the node.

7. Branch the node 3; the LP is infeasible. Fathom the node 3.

8. Node 1 is active. Go to step 4. Solving the LP gives

$$z_{lp}^1 = \frac{22}{3}$$

$$x^1 = \left(2, \frac{2}{3}\right)$$

9. Branching on x2 = 2/3 gives node 5 and node 6. Node 5 gives an infeasible solution and is fathomed.

10. Go to step 4 and solve the LP relaxation for node 6. We get

$$z_{lp}^6 = \frac{19}{2}$$

$$x^6 = \left(\frac{3}{2}, 1\right)$$

11. Branching on $x_1 = \frac{3}{2}$ gives node 7 and node 8. Node 7 is associated with $x_1 \leq 1$ and

node 8 is associated with $x_1 \geq 2$.

12. Solve node 7. Solving the LP gives

$$z_{lp}^7 = \frac{35}{3}$$

$$x^7 = \left(1, \frac{4}{3}\right)$$

$\underline{z}^7 = \lceil z_{lp}^7 \rceil = 12 \Rightarrow \underline{z}^7 > \overline{z}^0 \Rightarrow$ node 7 is fathomed.

13. Branching on the $x_1 \geq 2$ constraint gives an all-integer solution

$$x^8 = (2,1)$$

$$z_{lp}^8 = \overline{z}^8 = 10$$

14. At step 5 we have $\overline{z}^0 = \min\{11,10\} = 10$. Node 8 is fathomed.

15. There are no actives nodes remaining. We got the optimal solution:

<u>Answer:</u> $z^* = 10, x^* = (2,1)$

See Figure 5.5 for a complete solution tree.

## <u>The Algorithm</u>

**Step 1.** Start at node i = 0

$\underline{z}^0 = -\infty, \overline{z}^0 = \infty$. Go to Step 2.

**Step 2.** If there are no active nodes, then go to Step 7. Otherwise, choose an active node i. If the LP is solved, go to Step 3; if not, go to Step 4.

**Step 3.** Choose $x_k \left(x_k = \lfloor x_k \rfloor + f_k, 0 < f_k < 1\right)$.

Partition $\begin{array}{l} S^i \cap \{x : x_k \leq \lfloor x_k \rfloor\} \\ S^i \cap \{x : x_k \geq \lceil x_k \rceil\} \end{array}$ Go to Step 2.

**Step 4.** Solve the LP relaxation. If the LP is not feasible, fathom the node and go to Step 2. If the LP has an optimal solution $x^i \Rightarrow \underline{z}^i = \lceil z_{lp}^i \rceil$., go to Step 5.

**Step 5.** If $x^i$ is not all-integer, go to Step 6; otherwise, $\overline{z}^i = z^i$ and fathom the node i. Then $\overline{z}^0 = \min\{\overline{z}^0, \overline{z}^i\}$ and go to Step 6.

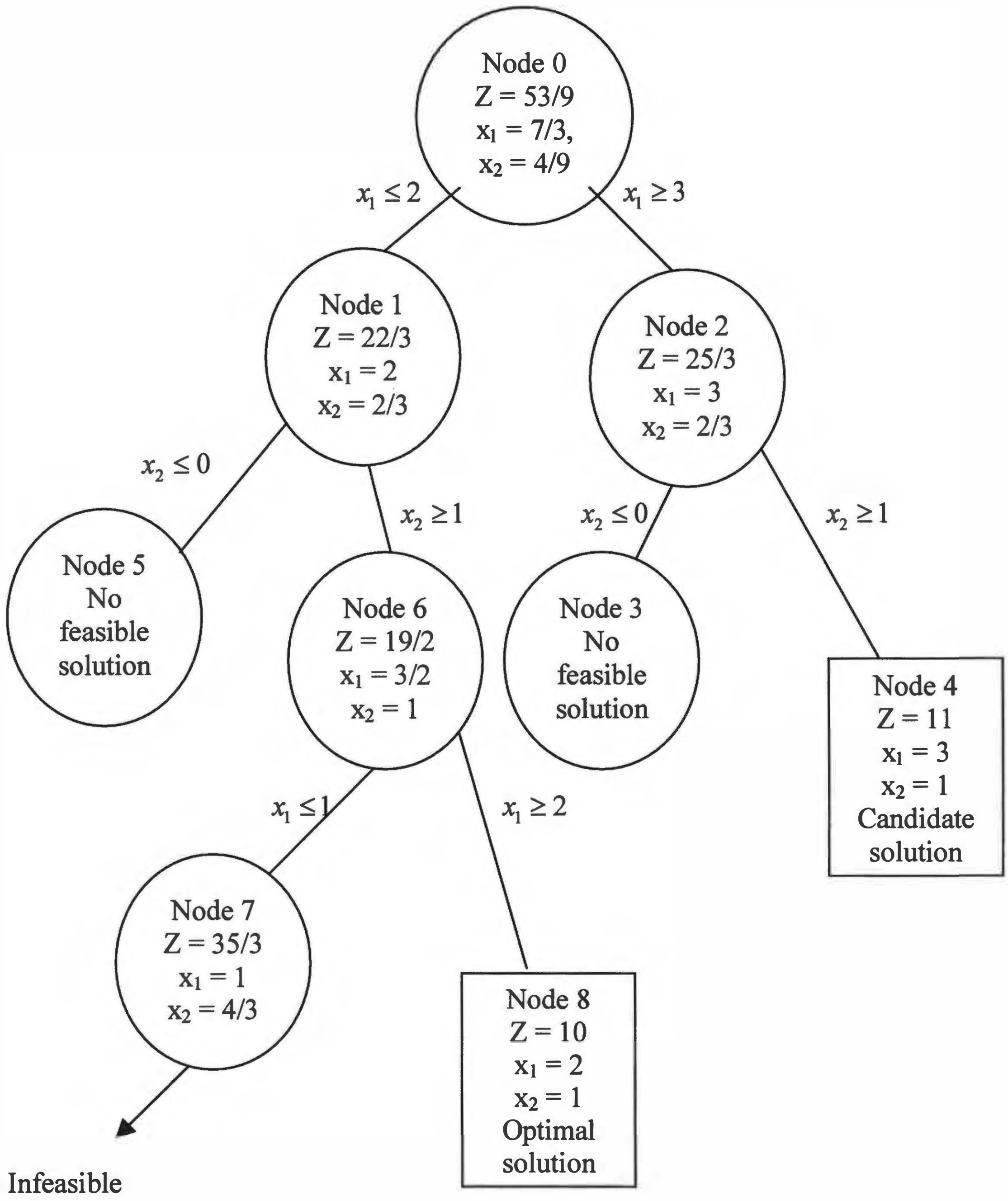

**Figure 5.5 The Solution Tree**

**Step 6.** Fathom only node i, s.t. $\underline{z}^i \ge \overline{z^0}$ and go to Step 2.

**Step 7.** Termination. If $\overline{z^0} = \infty$, then there is no feasible solution. If $\overline{z^0} < \infty \Rightarrow$ the feasible solution gives $\overline{z^0}$ which is optimal.

Example 5.2 solved using the LINDO

```
Optimum Found at Step 2
Objective Function Value
1)        5.888889

Variable        Slack or Surplus        Reduced Cost
X               2.333333                0.000000
Y               0.444444                0.000000

Row             Slack or Surplus        Dual Prices
2)              0.000000                -1.222222
3)              7.555555                0.000000
4)              4.222222                0.000000
5)              0.000000                1.444444

No. Iterations =  2

The Tableau

Row             Basis        X          Y          Slk 2      Slk 3      Slk 4
1               ART          1.000      8.000      0.000      0.000      0.000
2               SLK 2        -2.000     -3.000     1.000      0.000      0.000
3               SLK 3        -3.000     1.000      0.000      1.000      0.000
4               SLK 4        1.000      1.000      0.000      0.000      1.000
5               SLK 5        1.000      -3.000     0.000      0.000      0.000
ART             ART          -2.000     -3.000     0.000      0.000      0.000


Row             SLK          5
1               0.000        0.000
2               0.000        -6.000
3               0.000        1.000
4               0.000        7.000
5               1.000        1.000
ART             0.000        -6.000
```

```
X Enters at Value   1.0000    in Row    5 Obj. Value = -1.0000
Y Enters at Value   0.4444    in Row    2 Obj. Value = -5.8889

Objective Function Value
1)        5.888889


Variable        Value            Reduced Cost
X               2.333333         0.000000
Y               0.444444         0.000000


Row             Slack or Surplus              Dual Prices
2)              0.000000                      -9.000000
3)              7.555555                      -8.000000
4)              4.222222                      4.000000
5)              0.000000                      -3.000000

No. Iterations =  2

LP Optimum Found at Step  2


Objective Function Value
1)        5.888889


Variable        Value          Reduced Cost
X               2.333333       0.000000
Y               0.444444       0.000000


Row             Slack or Surplus              Dual Prices
2)              0.000000                      -1.222222
3)              7.555555                      0.000000
4)              4.222222                      0.000000
5)              0.000000                      1.444444
```

**Solution**

Min z = 5.888889

$x_1 = 2.333333 = 2.\overline{3}$

$x_2 = 0.444444 = 0.\overline{4}$

or

$$x_1 = 2\frac{1}{3} = \frac{7}{3}$$

$$x_2 = \frac{4}{9}$$

$$z_{lp}^0 = \frac{53}{9}$$

$$x^0 = \left(\frac{7}{3}, \frac{4}{9}\right)$$

# CHAPTER 6

# CUTTING PLANE ALGORITHM

## 6.1 The Fundamental Concepts of the Cutting Method

In Chapter 5 we introduced the basic idea of the Branch-and-Bound-methods. The objective function of those methods behaved in such manner as to more parallel to itself in the direction where its value is deterioration from the continuous extreme point. These shifts are planned in such a way that it is impossible to avoid the superior extreme point. The cutting methods have different approach restructuring of the feasible space by adding some special (valid) constraints in order to find the required optimum feasible point of the modified solution space. The basic idea of these cutting methods is that these valid inequalities cut off parts of the solution space in a way to include all feasible points. The Figure 6.1 illustrates the application of the cutting methods.

The lattice points of the continuous convex solution space are the feasible integer points. The point A is continuous optimum point (not integer). The cutting planes I and I I are restructuring the continuous solution space in such a way to obtain the extreme point with the integrality conditions.

There might be applied several cuts until we find the required extreme point. We need to add that the cutting methods are converged methods so the application of different cuts is finite.

In his early works Dantzig et al (1954) [7] and later Markowitz and Manne [19] were the first to introduce a cut for solving a linear programming problem in terms of integers.

The first finite cutting algorithms were developed by Gomory (1958) [13]. For the pure IP Gomory presented how to construct a cut in a systematical way from the simplex tableau. In 1960 Gomory [14] introduced the cutting algorithm for the mixed IP. The cutting algorithm was the dual type and the solution to the problem is not available until the algorithm terminates.

The applications of the cutting method require considering the LP-relaxation problems by using the corresponding simplex algorithm. If we cannot obtain the integer solutions, then the restructuring cutting constraints must be added. The application of the dual algorithm restructures the feasibility and yield of the optimal answer. If this answer is integer, the process terminates. If the answer is not an integer, the new constraints must be added and the algorithm is repeated.

The fundamental concept of the cutting methods is the valid inequality and such cut is restructuring the convex polyhedron of the LP-relaxation problem.

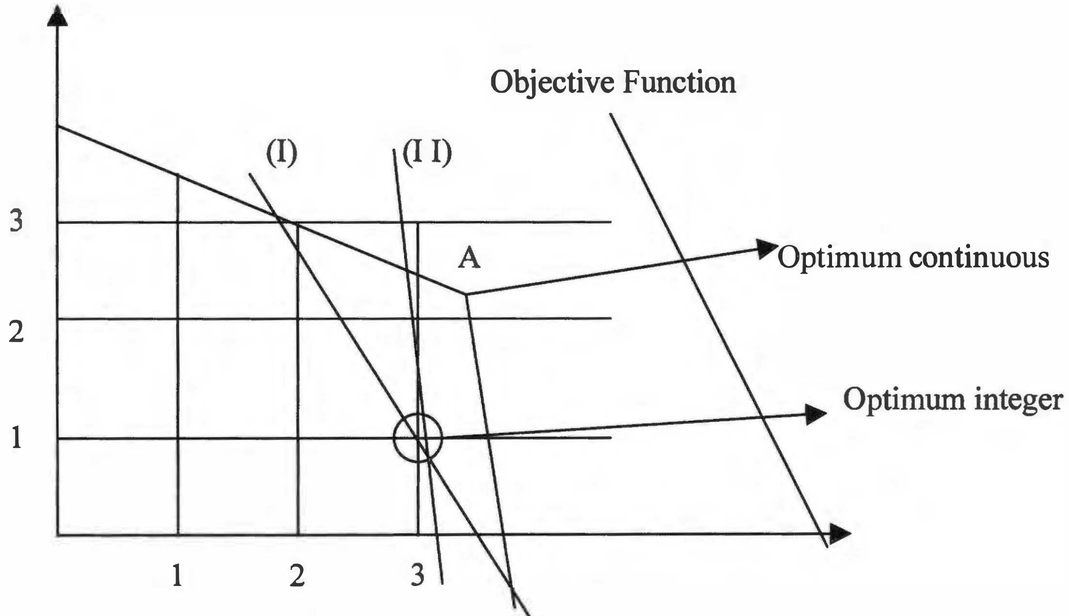

**Figure 6.1 The Cutting Planes I and II**

### 6.2 Valid Inequalities

Let me introduce some theoretical excerpts for the general IP problem. Let us consider the general integer problem:

$$\text{(IP)} \qquad \max\{cx : x \in X\},$$

$$\text{where } X = \{x : Ax \le b, x \in Z_t^n\}.$$

The following proposition tells us that we can solve an IP problem in terms of the linear program.

*Proposition 6.1* $\quad \operatorname{conv}(X) = \{x : \tilde{A}x \le \tilde{b}, x \ge 0\}$ is a polyhedron.

This proposition says that for any value of c, an optimal extreme point solution of LP is simultaneously an optimal solution of an IP. The same results we have for mixed integer program, where $X = \{(x, y) \in R_t^n \times Z_t^n : Ax + Gy \le b\}$,and A, G, b are rational.

In connection with the given proposition let us give the definition of polyhedron, and, subsequently, the definitions of a formulation and a convex hull.

*Definition 6.1* A subset of R$^n$ and a finite set of linear constraints given over this subset is a polyhedron: $P = \{x \in R^n : Ax \le b\}$.

*Definition 6.2* A polyhedron $P \subseteq R^{n+p}$ is called a formulation for $x \subseteq Z^n \times R^n$ if $x = P \cap \{Z^n \times R^n\}$

*Definition 6.3* For the given set $x \subseteq R^n$, the convex hull is denoted by $\operatorname{conv}(X) = \left\{x : x = \sum_{i=1}^{n} \lambda_i x^i\right\}$, where $\sum_{i=1}^{t} \lambda_i = 1$, $\lambda_i \ge 0$ for i = 1,…,t over all finite subsets $\{x^1, \cdots, x^t\}$ of X.

We have the following important proposition:

*Proposition 6.2* $\quad$ The extreme points of conv(X) all lie in X.

These two propositions shed the light for the question of the better formulation that is important for the reduction to the linear program.

Since $X \subseteq \operatorname{conv}(X) \subseteq P$ for all formulations P, this suggests the next definition.

*Definition 6.4* For the given set $X \subseteq R^n$ and two formulations P$_1$ and P$_2$ in X, P$_1$ is better than P$_2$ if $P_1 \subset P_2$.

As it was said, the result of this proposition tells us thet we can restructure an IP problem as the linear program

$$\text{(LP)} \quad \max\{cx : \tilde{A}x \le \tilde{b}, x \ge 0\},$$

and, then for any value of c an optimal extreme point solution of LP is an optimal solution to the respective IP.

The important concept for the IP problems is the concept of a valid inequality.

*Definition 6.5* The inequality $\pi x \le \pi_0$ is said to be valid for $X \subseteq R^n$ if this inequality holds for all x ∈ X.

Suppose we have $X = \{x \in Z^n : Ax \le b\}$ and $conv(X) = \{x \in R^n : \tilde{A}x \le \tilde{b}\}$. Then, the constraints $a^i x \le b^i$ and $\tilde{a}^i x \le \tilde{b}^i$ are obviously valid inequalities for X.

The right questions would be asked appropriately:

i. Which of the valid inequalities is more useful?

ii. How to use the “good” valid inequality to solve a particular problem?

### 6.2.1 Some Valid Inequalities

Let us consider the following example of the valid inequalities:

*Example 6.1* For a pure 0-1 let us consider the Knapsack problem.

$$X = \{x \in B^5 : 2x_1 - 3x_2 + 3x_3 - 3x_4 + x_5 \le -3\}$$

If $x_2 = x_4 = 0$, then the left-hand side $= 2x_1 + 3x_3 + x_5 \ge 0$. The right-hand side = -3, which is impossible. So, we obtain the valid inequality $x_2 + x_4 \ge 1$. If $x_1 = 1$ and $x_2 = 0$, (lhs) $= 2 + 3x_3 - 3x_4 + x_5 \ge 2 - 2 = 0$ and (rhs) = -3, which is impossible. This result suggests us that $x_1 \le x_2$ is also a valid inequality.

*Example 6.2* For a mixed 0-1 set let us consider $X = \{(x, y) : x \le 999y, 0 \le x \le 6, y \in B^1\}$, where $B^1$ is the set of one-dimensional 0-1 vectors.

We can check that the inequality $x \le 6y$ is valid caused by $X = \{(0,0), (x,1)\}$ with $0 \le x \le 6$.

*Example 6.3* Let us consider the set $X = \{(x, y): x \le 8y, 0 \le x \le 12, y \in Z_t^1\}$. The inequality $x \le 6 + 4y$ or $x \le 14 - 4(2 - y)$ is valid. On Figure 6.2 we describe the valid inequality graphically and see that the added inequality gives the convex hull of X.

In general case, when we cannot divide c by b and for the given $X = \{(x, y): x \le cy, 0 \le x \le b, y \in Z_t^1\}$ the valid inequality is as follows: $x \le b - \lambda(k - y)$, where $k = \left\lceil \frac{b}{c} \right\rceil$ and $\lambda = b - \left(\left\lceil \frac{b}{c} \right\rceil - 1\right)c$.

*Example 6.4* This example represents the sample of an integer rounding. Let us have the integer region $X = P \cap Z^4$, where formulation $P = \{x \in R_t^4 : 15x_1 + 21x_2 + 13x_3 + 5x_4 \ge 80\}$.

Suppose we divide by the coefficient 13 of $x_3$. This gives us the valid inequality for P

$$\frac{15}{13}x_1 + \frac{21}{13}x_2 + x_3 + \frac{5}{13}x_4 \ge 6\frac{2}{13}$$

Since x ≥ 0 we can round up the coefficients on the left and right up to the integers.

$$2x_1 + 2x_2 + x_3 + x_4 \ge 7$$

*Example 6.5* Mixed Integer Rounding.

Let us consider the formulation P of the Example 6.4

$$P = \{x \in R_t^4 : 15x_1 + 21x_2 + 13x_3 + 5x_4 \ge 80\}$$

In terms of a general transportation problem there are four types of truck available in the formulation P, accordingly. Again, dividing by 13 yields a valid inequality

$$\frac{15}{13}x_1 + \frac{21}{13}x_2 + x_3 + \frac{5}{13}x_4 \ge \frac{80}{13}$$, and so we get

$$2x_1 + 2x_2 + x_3 + x_4 + \beta s \ge 7 \text{ for some } \beta \quad (*)$$

Analyzing the term $\frac{80 - s}{13}$ we can conclude that (rhs) $\left\lceil \frac{80 - s}{13} \right\rceil$ decreases from 7 to 6 at the critical value s = 2. This suggests that the value of $\beta = \frac{1}{2}$. The inequality (*) is valid for the values of $\beta \ge \frac{1}{2}$, and it turns out to be the following inequality:

$$\frac{3}{2}x_1 + 2x_2 + x_3 + x_4 + \frac{1}{2}s \ge 7$$

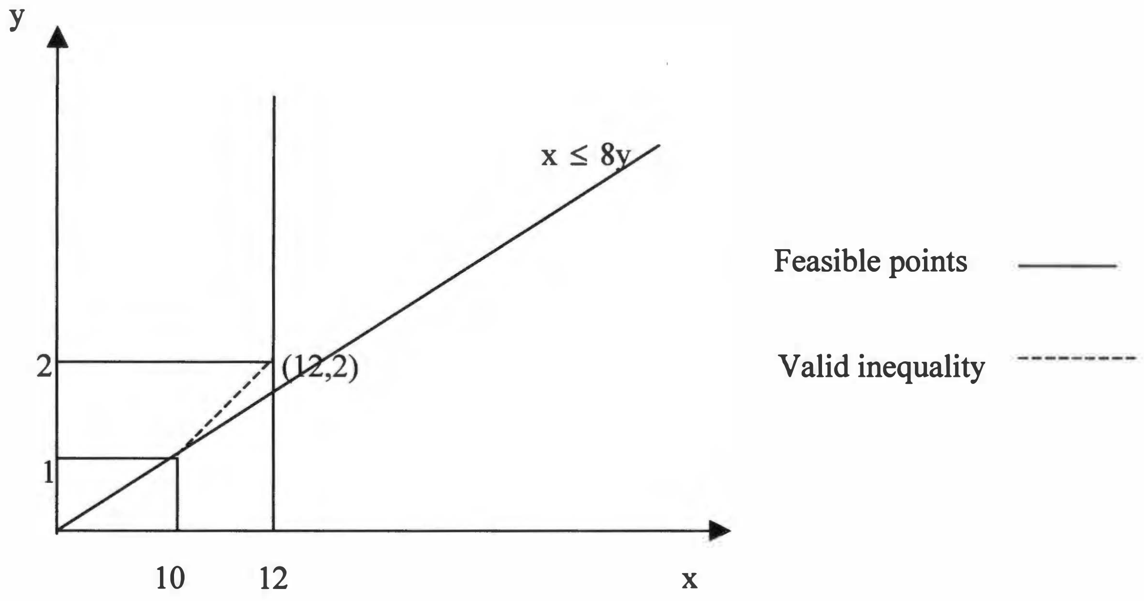

**Figure 6.2 The Convex Hull of X and the Valid Inequality**

### 6.3 Valid Inequalities for a Linear Programming Problem

In order to construct the valid inequalities for an IP problem, first of all, we have to discuss the valid inequalities for the LP problem:

*Proposition 6.3* The inequality $\pi x \leq \pi_0$ of the LP is valid for $P = \{x : Ax \leq b, x \geq 0\} \neq 0$ for $uA - v = \pi$, we have $ub \leq \pi_0$, or alternatively there exists $u \geq 0$ such that $uA \geq \pi$ and $ub \leq \pi_0$.

This proposition tells us that by linear programming duality $\max\{\pi x : x \in P\} \leq \pi_0$ if and only if $\max\{ub : uA - v = \pi, u \geq 0, v \geq 0\} \leq \pi_0$.

### 6.3.1 Valid Inequalities for an IP Problem

Let us consider the feasible region of an IP, and, then find the corresponding valid inequality. The following proposition tells us how to construct the valid inequality of an IP.

*Proposition 6.4* If $x = \{x \in Z^1 : x \leq b\}$, then the valid inequality for x is x $\leq \lfloor b \rfloor$

Next example is a vivid illustration of this proposition.

*Example 6.6* Valid Inequalities of an IP.

This example shows the approach to be used to construct valid inequalities for any IP problems. Suppose the IP region is given by $x = P \cap Z^n$, where the formulations are the following

$$\begin{aligned} &8x_1 - 2x_2 \leq 15 \\ &x_2 \leq 5 \\ &3x_1 - 2x_2 \leq 5 \\ &x_1, x_2 \geq 0 \end{aligned}$$

Multiplying the first constraint by nonnegative weight 3/8 yields the valid inequality for an IP

$$3x_1 - \frac{6}{8}x_2 \leq \frac{45}{8}$$

Reducing the coefficients on the left-hand side to the nearest integer yields the valid inequality for the formulation P

$$3x_1 - 0 \times x_2 \leq \frac{45}{8}$$

Rounding the coefficients on the right-hand side gives us

$$3x_1 \le \left\lfloor \frac{45}{8} \right\rfloor = 5$$

So, we obtain the valid inequality $3x_1 \le 5$.

Observe that if we use a weight of 1/3 on the third constraint, we could obtain the tighter inequality

$$x_1 - \frac{2}{3}x_2 \le \frac{5}{3}$$

$$x_1 - 0x_2 \le \left\lfloor \frac{5}{3} \right\rfloor = 1$$

$$or$$

$$x_1 \le 1$$

Now let us introduce the general procedure to construct the valid inequalities for any IP problem. This procedure is called Chvatal-Gomory procedure to construct the valid inequality for the set $X = P \cap Z^n$, where the formulation $P = \{x \in R_t^n : Ax \le b\}$, A is an $m \times n$ matrix with columns $\{a_1, a_2, \cdots, a_n\}$ and $u \in R_t^m$:

i. The inequality

$\sum ua_j x_j \le ub$ is valid for any $u \in R_t^m$, $u \ge 0$.

ii. The inequality

$\sum \lfloor ua_j \rfloor x_j \le ub$ is valid for the given formulation P for $x \ge 0$.

iii. The inequality

$\sum_{j=1}^{n} \lfloor ua_j \rfloor x_j \le \lfloor ub \rfloor$ is valid for x is integer and the $\sum_{j=1}^{n} \lfloor ua_j \rfloor x_j$ is integer.

This three-step procedure is used to construct valid inequalities for any integer program. The idea behind it is to examine, first, the initial formulation $P = \{x : Ax \le b, x \ge 0\}$ with $X = P \cap Z^n$ and then find a set of valid inequalities $Qx \le q$ for x. Next, we need to add a new set of inequalities to formulation P obtaining a new formulation $P^1 = \{x : Ax \le b, Qx \le q, x \ge 0\}$ with $X = P^1 \cap Z^n$. Then we can apply the Branch-and-Bound algorithm, or whatsoever, to formulation P$^1$.

### 6.4 Gomory's Fractional Cutting Plane Algorithm

Let us consider the integer program

max {cx: Ax = b, x ≥ 0 and integer}

First, we have to solve the corresponding LP-relaxation problem and find an optimal basis. Next, we have to choose the basic variable that is not an integer, and then apply the Chvatal-Gomory procedure to construct a valid inequality. That means, we cut off the continuous linear programming solution.

Generally, suppose we have a problem as it follows:

$$\max \overline{a_{00}} + \sum_{j\in NB} \overline{a_{0j}} x_j$$

$$x_{Bu} + \sum_{j\in NB} \overline{a_{uj}} x_j = \overline{a_{u0}} \text{ for } u = 1,\cdots,m$$

x ≥ 0 and integer, where $\overline{a_{0j}} \leq 0$ for $j \in NB$ and $\overline{a_{u0}} \geq 0$ for $u = 1,\cdots,m$

NB is the set of non-basic variables.

Suppose that the basic optimal solution x* is not integer. Then, there exists some row u with $\overline{a_{u0}} \notin Z^1$. The Chvatal-Gomory cut for row u is

$$x_{Bu} + \sum_{j\in NB} \lfloor \overline{a_{uj}} \rfloor x_j \leq \lfloor \overline{a_{u0}} \rfloor$$

If we eliminate $X_{Bu}$ we obtain

$$\sum \left(\overline{a_{uj}} - \lfloor \overline{a_{uj}} \rfloor\right) x_j \geq \overline{a_{u0}} - \lfloor \overline{a_{u0}} \rfloor$$

If we denote by $f_{uj} = \overline{a_{uj}} - \lfloor \overline{a_{uj}} \rfloor \mathit{for}\ j \in NB$ and $f_{u0} = \overline{a_{u0}} - \lfloor \overline{a_{u0}} \rfloor$ then we can rewrite the last inequality as $\sum_{j\in NB} f_{uj} x_j \geq f_{u0}$

By the definitions we have $0 \leq f_{uj} < 1$ and $0 < f_{u0} < 1$. Since $x_j^* = 0$ for all non-basic variables j ∈ NB in the LP-relaxation solution, this inequality allows us to cut off x*. Let us now present a basic cutting plane algorithm for an IP problem

max {cx: x ∈ X}

### 6.4.1 Cutting Plane Algorithm

**Step 1.** Initialization. Set t = 0 and $P^0 = P$.

**Step 2.** Solve the LP-relaxation problem

$$\overline{z^t} = \max\{cx : x \in P^t\}$$

Find the optimal solution $x^t$.

**Step 3.** Check of integrality. If $x^t \in Z^n$, algorithm terminates. If $x^t \notin Z^n$, find the inequality $(\pi^t, \pi_0^t) \in X = P \cap Z^n$ with $\pi^t x^\tau > \pi_0^t$, so that it cuts off $x^t$.

**Step 4.** Set $P^{t+1} = P^t \cap \{x : \pi^t x \le \pi_0^t\}$, augment t. Go to Step 2, otherwise, go to Step 5.

**Step 5.** Termination.

We must observe that if an algorithm terminates without finding an integral solution, then the formulation $P^t$ from Step 4 is an improved formulation for a Branch-and-Bound method. This formulation is given as follows

$$P^t = P \cap \{x : \pi^i x \le \pi_0^i, i = 1, \cdots, t\}$$

The following example will illustrate the presented cutting plane algorithm. This example was already presented for the Branch-and-Bound method in Chapter 5.

*Example 6.7* Let us consider an IP problem

Maximize $z = 3x_1 - x_2$

Subject to
$$\begin{aligned} &7x_1 - 2x_2 \le 15 \\ &x_2 \le 4 \\ &2x_1 - 2x_2 \le 3 \\ &x_1, x_2 \ge 0 \text{ and integer} \end{aligned}$$

Adding slack variables $x_3, x_4, x_5$ we must note that since all constraints are integers, the slack variables are also integers. Solving the LP problem gives us:

Maximize $z = \frac{63}{10} - \frac{2}{5}x_3 - \frac{1}{10}x_5$

Subject to
$$\begin{aligned}
&x_1 + \frac{1}{5}x_3 - \frac{1}{5}x_5 = \frac{24}{10}\\
&x_2 + \frac{1}{5}x_3 - \frac{7}{10}x_5 = \frac{9}{10}\\
&-\frac{1}{5}x_3 + x_4 + \frac{7}{10}x_5 = \frac{31}{10}\\
&x_i \ge 0 \text{ and integer}, i = 1,\cdots,5
\end{aligned}$$

The optimal LP solution is $\left(\frac{24}{10},\frac{9}{10},0,\frac{31}{10},0\right)$. We use the first row where the basic variable $x_1$ is fractional to construct the cut:

$$s = -\frac{4}{10} + \frac{1}{5}x_3 - \frac{1}{5}x_5, \text{ where } s, x_3, x_5 \ge 0 \text{ and integer.}$$

We have to add this cut and reoptimize. We will get a new optimal tableau

Maximize $z = \frac{11}{2} - \frac{1}{2}x_5 - 2s$

Subject to
$$\begin{aligned}
&x_1 + s = 2\\
&x_2 - \frac{1}{2}x_5 + s = \frac{1}{2}\\
&x_3 - x_5 - 5s = 11\\
&x_4 + \frac{1}{2}x_5 - s = \frac{35}{10}\\
&x_i, s \ge 0 \text{ and integer}, i = 1,\cdots,5
\end{aligned}$$

Now the new optimal LP solution is $\left(2,\frac{1}{2},11,\frac{35}{10},0\right)$ which is still not integer. The Gomory fractional cut for row, where $x_2$ is basic is the following cut:

$$\frac{1}{2}x_5 \ge \frac{1}{2} \text{ or } -\frac{1}{2}x_5 + t = -\frac{1}{2}, t \ge 0 \text{ and integer.}$$

Adding this constraint and reoptimizing, we get

Maximize $z = 5 - 2s - t$

$$\text{Subject to}\quad \begin{aligned} &x_1 + s = 2 \\ &x_2 + s - t = 1 \\ &x_3 - 5s - 2t = 12 \\ &x_4 - s + t = 3 \\ &x_1, s, t \geq 0 \,\text{and intieger}, i = 1, \cdots, 5 \end{aligned}$$

Now the LP solution is integral and optimal for the corresponding IP, and thus $(x_1, x_2) = (2,1)$ solves the IP problem.

## 6.5 Mixed Integer Cuts

Now we discuss how to generate the valid inequalities for mixed integer programs. We have the proposition:

*Proposition 6.5* Let $X = \{(x, y) \in R_t^1 \times Z^1 : x + y \geq b\}$, and $F = b - \lfloor b \rfloor > 0$. Then the inequality $x \geq F(\lceil b \rceil - y)$x or $\frac{x}{F} + y \geq \lceil b \rceil$ is valid for x.

Indeed, if $y \geq \lceil b \rceil$, then $x \geq 0 \geq F(\lceil b \rceil - y)$. If $y < \lceil b \rceil$, then

$$x \geq b - y = F + (\lfloor b \rfloor - y) \geq F + F(\lfloor b \rfloor - y) = F(\lceil b \rceil - y)$$

as $\lfloor b \rfloor - y \geq 0$ and $F < 1$.

### 6.5.1 The Mixed Integer Rounding Inequality (MIR)

Let us consider the MIR set

$$X^{MIR} = \{(x, y) \in R_t^1 \times Z_t^2 : a_1 y_1 + a_2 y_2 \leq b + x\}$$

in which $a_1, a_2, b$ are scalars, $b \notin Z^1$.

Next proposition gives us a tool to construct the valid MIR inequality.

*Proposition 6.6* If $F = b - \lfloor b \rfloor$ and $F_i = a_i - \lfloor a_i \rfloor, i = 1,2$ and if $F_1 \leq F \leq F_2$ then $\lfloor a_1 \rfloor y_1 + \left( \lfloor a_2 \rfloor + \frac{F_2 - F}{1 - F} \right) y_2 \leq \lfloor b \rfloor + \frac{x}{1 - F}$ is a valid MIR inequality.

Indeed, $(x,y)\in X^{MIR}$(x,y) and satisfies $\lfloor a_1\rfloor y_1+\lceil a_2\rceil y_2\le b+x+(1-F_2)y_2$ as $y_1\ge 0$ and $a_2=\lceil a_2\rceil-(1-F_2)$. The Proposition 6.5 gives as

$$\lfloor a_1\rfloor y_1+\lceil a_2\rceil y_2\le\lfloor b\rfloor+\frac{x+(1-F_2)y_2}{1-F},$$

which is the desired inequality.

Next example is the illustration of the Proposition 6.6

*Example 6.8* Let us consider the set

$$X=\left\{(x,y)\in Z_t^3\times R_t^1:\frac{11}{2}y_1+y_2+\frac{13}{4}y_3\le\frac{23}{2}+x\right\}.$$

Using Proposition 6.6, we obtain the valid MIR inequality with $F=\frac{1}{2}$, $F_1=\frac{1}{2}$, $F_2=0$, $F_3=\frac{1}{4}$.

The MIR inequality is obtained in the form as it follows:

$$5y_1+y_2+\frac{11}{4}y_3\le 11+2x$$

### 6.5.2 Gomory's Mixed Integer Cut

As mentioned above, if a basic integer variable is fractional in any row of the LP optimal tableau, then exactly such variable must be used to generate a cut. In that case such a row leads to a constraint set of the form

$$x^G=\left\{(y_{Bu},y,x)\in Z^1\times Z_t^{n_1}\times R_t^{n_2}:y_{Bu}+\sum_{j\in n_1}\overline{a_{uj}}y_j+\sum_{j\in n_2}\overline{a_{uj}}x_j=\overline{a_{u0}}\right\},$$

where $n_i=|N_i|$ for $i=1{,}2$.

There is the following proposition for the Gomory mixed integer cut:

*Proposition 6.7* If $\overline{a_{u0}}\notin Z^1,F_j=\overline{a_{uj}}-\lfloor\overline{a_{uj}}\rfloor$ for $j\in N_1\cup N_2$ and $F_0=\overline{a_{u0}}-\lfloor\overline{a_{u0}}\rfloor$, then the Gomory's mixed integer cut is valid for $x^G$ and is off the form

$$\sum_{F_j\le F_0}F_jy_j+\sum_{F_j>F_0}\frac{F_0(1-F_j)}{1-F_0}y_j+\sum_{a_{uj}<0}\overline{a_{uj}}x_j+\sum_{a_{uj}<0}\frac{F_0}{1-F_0}\overline{a_{uj}}x_j\ge F_0$$

The Proposition 6.5 gives us the valid inequality for $X^{MIR}$ and this inequality is given as

$$\lfloor a_1 \rfloor y_1 + \left( \lfloor a_2 \rfloor + \frac{F_2 - F}{1-F} \right) y_2 \le \lfloor b \rfloor + \frac{x}{1-F}$$

The mixed integer rounding inequality for $x^G$ is

$$y_{Bu} + \sum_{F_j \le F_0} \lfloor \overline{a_{uj}} \rfloor y_j + \sum_{F_j > F_0} \left( \lfloor \overline{a_{uj}} \rfloor + \frac{F_j - F_0}{1-F_0} \right) y_j + \sum_{a_{uj}<0} \frac{\overline{a_{uj}}}{1-F_0} x_j \le \lfloor \overline{a_{u0}} \rfloor$$

If we substitute for $y_{Bu}$, then this will prove this proposition. Next example will show the use of the above–mentioned proposition. Example 6.9 is the further application of the Gomory Mixed Integer Cut and this example is the same as the Example.7

*Example 6.9* Solving the problem of the Example.7 as a linear program gives:

Maximize $z = \frac{63}{10} - \frac{2}{5}x_3 - \frac{1}{10}x_5$

Subject to
$$\begin{aligned} x_1 + \frac{1}{5}x_3 - \frac{1}{5}x_5 &= \frac{24}{10} \\ x_2 + \frac{1}{5}x_3 - \frac{7}{10}x_5 &= \frac{9}{10} \\ -\frac{1}{5}x_3 + x_4 + \frac{7}{10}x_5 &= \frac{31}{10} \\ x_1 \in Z_t^1, x_i \ge 0, i &= 2, \cdots, 5 \end{aligned}$$

The basic variable $x_1$ is fractional, so the first row cut gives the MIR as $x_1 \le 2$. If we eliminate $x_1$, then the Gomory Mixed Integer Cut is

$$\frac{1}{5}x_3 - \frac{1}{5}x_5 \ge \frac{4}{10} \quad \text{or}$$

$$\frac{1}{5}x_3 - \frac{1}{5}x_5 \ge \frac{2}{5}$$

Adding this cut and again reoptimizing gets the solution $x = \left(2, \frac{1}{2}\right)$. This solution is feasible and hence optimal for the mixed integer program.

# CHAPTER 7

# GOMORY METHOD and LP RELAXATION SIMPLEX ALGORITHM

## 7.1 Introduction

At the early stage of using Gomory method we are required to solve the LP relaxation by the basic simplex algorithm. If the extreme point solution is not integer, then a cut must be introduced. Such cut, however, preserves only the optimality state of the primal problem. Further application of the dual algorithm preserves both feasibility and yields an optimal extreme solution. If the solution is integer-valued, the process terminates. Otherwise, a new cut is introduced and the algorithm iterates until an integer solution yields. This method was overwhelmed by a more sufficient procedure in which the problem is transformed by a simplex algorithm, which maintained the integer-value characteristic of the tableau during all iteration [14].

Suppose that the starting simplex tableau is all-integer and that we begin with a feasible solution for the dual simplex algorithm but not the primal [10]. Let $P_1, P_2, \ldots, P_m$ be a feasible basis for the dual problem. Note that the initial Table 7.1 is given for a minimization problem. In Table 7.1 $x_{00}$ is the value of the objective function. Some $x_{i0} < 0$ for $i \geq 1$ when the solution is not primal feasible. $x_{0j} = z_j - c_j \leq 0$ for $j \geq 1$ when the solution is dual feasible and an additional set of $n - m$ constraints, $x_j^1 - x_j = 0$ for $j = m+1, \cdots, n$ with $x_j^1 \geq 0$ and $c_j^1 = c_j$ enhance the tableau, and now we are induced to find a basic feasible solution that contains *n* nonnegative variables instead of the usual *m*. Now we must control a simplex algorithm in such a way to move toward the interior optimal integer-valued point.

We now have to preserve the integer characteristic of the tableau and move toward the convex set defined by Equations (7.1). It can be done by finding cutting planes defined as equalities in terms of the non-basic and slack variables. Eliminating the new slack vector from the basis with a pivot –1 will move the solution towards the optimal integer point. After that, the finite number of the pivoting will bring us to the optimal integer solution.

## 7.2 Gomory Method for Pure IP Problem

To find the new cutting constraint we must consider any equation of the Table 7.1 with respective $x_{i0} < 0$, for instance, the $1^{\text{th}}$ equation

$$x_{l0} = x_l + x_{l,m+1}x_{m+1} + x_{l,m+2}x_{m+2} + \ldots + x_{ln}x_n \qquad (7.1)$$

in which $x_{m+1}, \ldots, x_n$ are non-basic variables.

**Table 7.1 Initial Tableau**

| | | | | $c_1$ | . | $c_l$ | . | $c_m$ | $c_{m+1}$ | . | $c_n$ | $c_{m+1}$ | . | $c_n$ | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| i | Basis | $c$ | $P_0$ | $P_1$ | . | $P_l$ | . | $P_m$ | $P_{m+1}$ | . | $P_n$ | $P^1_{m+1}$ | . | $P^1_n$ | $P_{n+s}$ |
| 0 | --- | - | $x_{00}$ | 0 | . | 0 | . | 0 | $x_{0,m+1}$ | . | $x_{0n}$ | 0 | . | 0 | |
| 1 | $P_1$ | $c_1$ | $x_{10}$ | 1 | . | 0 | . | 0 | $x_{1,m+1}$ | . | $x_{1n}$ | 0 | . | 0 | |
| . | . | . | . | . | . | . | . | . | . | . | . | . | . | . | |
| $l$ | $P_l$ | $c_l$ | $x_{l0}$ | 0 | . | 1 | . | 0 | $x_{l,m+1}$ | . | $x_{ln}$ | 0 | . | 0 | |
| $m$ | $P_m$ | $c_m$ | $x_{m0}$ | 0 | . | 0 | . | 1 | $x_{m,m+1}$ | . | $x_{mn}$ | 0 | . | 0 | |
| $m$+1 | $P^1_{m+1}$ | $c_{m+1}$ | 0 | 0 | . | 0 | . | 0 | -1 | . | 0 | 1 | . | 0 | |
| . | . | . | . | . | . | . | . | . | . | . | . | . | . | . | |
| $n$ | $P^1_n$ | $c_n$ | 0 | 0 | | 0 | | 0 | 0 | | -1 | 0 | | 1 | |
| $n$+1 | | | | | | | | | | | | | | | |

Next, we represent each coefficient of (7.1) as a multiple of an integer and a remainder in the form $b_{lj}\lambda + r_{lj}$, in which $b_{lj}$ is an integer, $r_{lj}$ is a remainder, and $\lambda > 0$ is a number to be found. In such way the coefficients can be represented as

$$x_{lj} = b_{lj}\lambda + r_{lj} = \left[\frac{x_{lj}}{\lambda}\right]\lambda + r_{lj} \quad \text{for all j,} \quad \left[\frac{1}{\lambda}\right]\lambda + r = 1 \qquad (7.2)$$

$$0 \le r_{lj} < \lambda \qquad 0 \le r < \lambda \qquad 0 < \lambda$$

Note that the brackets mean the "integer part of." If $\frac{x_{lj}}{\lambda} < 0$ then $\left[\frac{x_{lj}}{\lambda}\right] = b_{lj} < 0$ since there is (7.2). If $\lambda > 1$ note that $\left[\frac{1}{\lambda}\right] = 0.$

If we substitute (7.2) into (7.1) and combine the corresponding terms, we get

$$\left[\frac{x_{l0}}{\lambda}\right]\lambda + r_{l0} = \left(\left[\frac{1}{\lambda}\right]\lambda + r\right)x_l + \left(\left[\frac{x_{l,m+1}}{\lambda}\right]\lambda + r_{l,m+1}\right)x_{m+1} + \left(\left[\frac{x_{l,m+2}}{\lambda}\right]\lambda + r_{l,m+2}\right)x_{m+2} + \ldots + \left(\left[\frac{x_{ln}}{\lambda}\right]\lambda + r_{ln}\right)x_n$$

or

$$r_{l0} + \lambda\left(\left[\frac{x_{l0}}{\lambda}\right] - \left[\frac{x_{l,m+1}}{\lambda}\right]x_{m+1} - \left[\frac{x_{l,m+2}}{\lambda}\right]x_{m+2} - \left[\frac{x_{ln}}{\lambda}\right]x_n - \left[\frac{1}{\lambda}\right]x_l\right) \qquad (7.3)$$
$$= r_{l,m+1}x_{m+1} + r_{l,m+2}x_{m+2} + \ldots + r_{ln}x_n$$

We can rewrite the term in braces using summation notation

$$\left[\frac{x_{l0}}{\lambda}\right] - \sum_{j \notin B}\left[\frac{x_{lj}}{\lambda}\right]x_j - \left[\frac{1}{\lambda}\right]x_l \qquad (7.4)$$

The expression (7.4) is an integer and nonnegative since all bracketed variables and quantities are integers and the right-hand side of (7.3) is nonnegative.

If $\lambda > 1$, then the last term of (7.4) and (7.2) is zero and we can represent (7.4) as it follows,

$$b_{l0} - \sum_{j \notin B} b_{lj}x_j \ge 0 \qquad (7.5)$$

Note, that (7.5) is a cutting plane expressed by the current non-basic variables. If we subtract out the nonnegative slack variable $x_{n+s}$, then we get

$b_{l0} - \sum_{j \notin B} b_{lj} x_j - x_{n+s} = 0$ or

$$b_{l0} = b_{l,m+1} x_{m+1} + b_{l,m+2} x_{m+2} + \ldots + b_{ln} x_n + x_{n+s} \qquad (7.6)$$

Since $x_{n+s}$ is equal to the left-hand side of (7.5), the variable $x_{n+s}$ is an integer variable. For any $\lambda > 1$, (7.6) is satisfied by any integer solution of the LP problem. Then, nonnegative integer variable $x_{n+s}$ must be presented as a new variable. Since all the non-basic variables $x_{m+1} = \cdots = x_n = x_0$, then (7.6) is infeasible as $b_{l0} < 0$. If we choose $\lambda > 1$ that makes $b_{lj} (j \notin B)$ of (7.6) equal to $-1$, then (7.6) can be used as a cutting plane.

Since $\lambda$ must be chosen to get $b_{lk} = \left[\frac{x_{lk}}{\lambda}\right] = -1$, we have an equation from the dual simplex algorithm as it follows: For the set $x_{lj} < 0$ we must form the ratios $\frac{z_j - c_j}{x_{lj}}$ and

$$\text{let } \theta = \min_{x_{lj} < 0} \frac{z_j - c_j}{x_{lj}} = \frac{z_k - c_k}{x_{lk}} > 0 \qquad (7.7)$$

Thus, from the above equation (7.7) we have

$$\frac{z_k - c_k}{\left[\frac{x_{lk}}{\lambda}\right]} = \frac{z_k - c_k}{-1} \leq \frac{z_j - c_j}{\left[\frac{x_{lj}}{\lambda}\right]} = \frac{z_j - c_j}{b_{lj}} \text{for } x_{lj} < 0 \qquad (7.8)$$

and the new value of the objective function is given by

$x_{00} - \frac{x_{0k}}{-1} b_{l0} = x_{00} + x_{0k} b_{l0} = x_{00} + (z_k - c_k)\left[\frac{x_{l0}}{\lambda}\right]$, where $P_k$ is the pivot column such that $\theta = \min_{x_{lj} < 0} \frac{x_{0j}}{-1} = \frac{z_k - c_k}{-1}$.

Note that $z_k - c_k = 0$ means the degeneracy in the dual problem. (7.8) can be rewritten as $\frac{z_k - c_k}{-1} \leq \frac{z_j - c_j}{b_{lj}}$.

If we let $m_j$ be the largest integer for which $\frac{z_k - c_k}{-1} \leq \frac{z_j - c_j}{-m_j}$, then

$$-b_{lj} = -\left[\frac{x_{lj}}{\lambda}\right] \leq m_j \text{ for } x_{lj} < 0 \qquad (7.9)$$

Therefore, for each j the smallest λ from (7.9) gets $P_k$ to be the pivot column. The pivot element is given by $b_{lk} = -1$, where $\lambda_j = -\left(\frac{x_{lj}}{m_j}\right)$.

We have for $\lambda_{\min} = \max_{x_{lj}<0} \lambda_j$. If $\lambda_{\min} = \lambda_k = -x_{lk} = 1$ then the pivot element is –1, and next, we construct the appropriate cutting constraint that consists of the non-basic variables. Note that λ is not limited to be an integer. So, for the $P_k$ we get

$m_k = 1, \lambda_k = -x_{lk}, \lambda_{\min} \geq \lambda_k$, and, hence $b_{lk} = \left[\frac{x_{lk}}{\lambda_{\min}}\right] = -1$.

The simplex procedure can be summarized in the following steps:

**Step 1.** Determine the dual feasible solution with an all-integer tableau.

**Step 2.** Choose the rows containing a negative constant term. Select a row to be used to construct the cutting constraint (the $1^{\text{th}}$ row). In a case if no such row exists, the optimal all-integer solution is found.

**Step 3.** For $-(z_k - c_k) = \min_{x_{lj}<0}\{-(z_j - c_j)\}$for $j \geq 1$ there exists the corresponding pivot column $P_k$. In a case if there is no $x_{lj} < 0$ then the problem is not feasible.

**Step 4.** For every *j* with $x_{lj} < 0$ define the largest integer $-(z_k - c_k) \leq \frac{-(z_j - c_j)}{m_j}$ and

$\lambda_j = -\frac{x_{lj}}{m_j}$ for $x_{lj} < 0$.

**Step 5.** Choose $\lambda_{\min} = \max \lambda_j$. If $\max \lambda_j = 1$ then set $\lambda_{\min} > 1$.

**Step 6.** Construct the cutting constraint (7.6) and add it to Table 7.1 with the new equation being $(n+1)^{\text{st}}$ and the new basic variable $x_{n+s}$ associated to column $P_{n+s}(s = 1,2,\cdots)$.

**Step 7.** Use $b_{lk} = -1$ as the pivot element and apply simplex algorithm. The result caused $x_{n+s}$ to be a non-basic variable. Since the pivot element is –1, the tableau remains in integers. The new row dropped out and the process continued.

### 7.3 The LP-based Relaxation and Gomory Cuts

We illustrate the given procedure by the following example:

**Example 7.1** Solve the problem in terms of integers:

Minimize $z = x. + 8x_2$

$$
\begin{aligned}
& 2x_1 + 3x_2 \geq 6 \\
& -3x_1 + x_2 \leq 1 \\
\text{Subject to } & x_1 + x_2 \leq 7 \\
& x_1 - 3x_2 \leq 1 \\
& x_1, x_2 \geq 0 \text{and integer}
\end{aligned}
$$

Adding slack variables will give Table 7.2:

Minimize $z = x. + 8x_2 + 0x_3 + 0x_4 + 0x_5 + 0x_6$

$$
\begin{aligned}
& -2x_1 - 3x_2 + x_3 = -6 \\
& -3x_1 + x_2 + x_4 = 1 \\
\text{Subject to } & x_1 + x_2 + x_5 = 7 \\
& x_1 - 3x_2 + x_6 = 1 \\
& x_i \geq 0 \text{and integer,i} = 1, \cdots, 6
\end{aligned}
$$

1) Add additional equation $x_1' = x_1$, $x_2' = x_2$

2) Since $x_{10} < 0$, let L = 1

3) $-(z_1 - c_1) = \min(1,8), \Rightarrow k = 1$

4) $1 \leq \dfrac{1}{m_1} \Rightarrow m_1 = 1, 1 \leq \dfrac{8}{m_2} \Rightarrow m_2 = 8$

5) $\lambda_1 = -\left(\dfrac{-2}{1}\right) = 2$, $\lambda_2 = -\left(\dfrac{-3}{8}\right) = \dfrac{3}{8}$

6) $\lambda = \max\left(2, \dfrac{3}{8}\right) = 2$

**Table 7.2 Initial Step of Example 7.1**

| | | | | 1 | 8 | 0 | 0 | 0 | 0 | 1 | 8 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| i | Basis | $c_b$ | $P_0$ | $P_1$ | $P_2$ | $P_3$ | $P_4$ | $P_5$ | $P_6$ | $P_1'$ | $P_2'$ | $P_7$ |
| 0 | | | 0 | -1 | -8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | $b_3$ | 0 | -6 | -2 | -3 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | $b_4$ | 0 | 1 | -3 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 3 | $b_5$ | 0 | 7 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 4 | $b_6$ | 0 | 1 | 1 | -3 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 5 | $b_1'$ | 1 | 0 | -1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| 6 | $b_2'$ | 8 | 0 | 0 | -1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 7 | | | -3 | -1 | -2 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

7) $\left[\frac{-6}{2}\right]=\left[\frac{-2}{2}\right]x_1+\left[\frac{-3}{2}\right]x_2+x_7 \Rightarrow -3=-x_1-2x_2+x_7$. Add this to Table 7.2

8) Do the simplex transformation to get Table 7.3

If we remove $P_3, P_4, P_5, P_6, P_1$ and place $P_7$ in the column of $P_1$ we'll get Table 7.4. The (n+1)st row is clear for addition of a new constraint in the next iteration:

1) L = 4

2) Since only $x_{42} \le 0 \Rightarrow k = 4$

3) $6 \le \dfrac{6}{m_2} \Rightarrow m_2 = 1$

4) $\lambda_2 = -\left(\dfrac{-5}{1}\right) = 5 \Rightarrow \lambda = 5$

5) $\left[\frac{-2}{5}\right]=\left[\frac{-5}{-5}\right]x_2+\left[\frac{1}{5}\right]x_7+x_8$

6) $-1=-1x_2+0x_7+x_8$ → new constraint

Adding new constraint $-1=-1x_2+0x_7+x_8$ → we get Table 7.5.

Again we find the new constraint $-1=-x_7+x_9$ and place it in Table 7.5. We do the transformation and find $x_{i0} \ge 0, i=1,\cdots,n$. So, we don't need to complete Table 7.6.

The solution is

$$\begin{aligned}
x_1' &= x_1^* = 2 \\
x_2' &= x_2^* = 1 \\
x_3^* &= 1 \\
x_4^* &= 6 \\
x_5^* &= 4 \\
x_6^* &= 2 \\
z^* &= 10
\end{aligned}$$

**Table 7.3 Second Step of Example 7.1**

| | | | | 1 | 8 | 0 | 0 | 0 | 0 | 1 | 8 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| i | Basis | $c_b$ | $P_0$ | $P_1$ | $P_2$ | $P_3$ | $P_4$ | $P_5$ | $P_6$ | $P_1'$ | $P_2'$ | $P_7$ |
| 0 | | | 3 | 0 | -6 | 0 | 0 | 0 | 0 | 0 | 0 | -1 |
| 1 | $b_3$ | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | -2 |
| 2 | $b_4$ | 0 | 10 | 0 | 7 | 0 | 1 | 0 | 0 | 0 | 0 | -3 |
| 3 | $b_5$ | 0 | 4 | 0 | -1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 |
| 4 | $b_6$ | 0 | -2 | 0 | -5 | 0 | 0 | 0 | 1 | 0 | 0 | 1 |
| 5 | $b_1'$ | 1 | 3 | 0 | 2 | 0 | 0 | 0 | 0 | 1 | 0 | -1 |
| 6 | $b_2'$ | 8 | 0 | 0 | -1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 7 | | | | | | | | | | | | |

**Table 7.4 Third Step of Example 7.1**

| i | Basis | $P_0$ | $P_7$ | $P_2$ | $P_8$ |
|---|---|---|---|---|---|
| 0 | | 3 | -1 | -6 | 0 |
| 1 | $b_3$ | 0 | -2 | 1 | 0 |
| 2 | $b_4$ | 10 | -3 | 7 | 0 |
| 3 | $b_5$ | 4 | 1 | -1 | 0 |
| 4 | $b_6$ | -2 | 1 | -5 | 0 |
| 5 | $b_1'$ | 3 | -1 | 2 | 0 |
| 6 | $b_2'$ | 0 | 0 | -1 | 0 |
| 7 | | -1 | 0 | -1 | 1 |

Table 7.5 Fourth Step of Example 7.1

| i | Basis | $P_0$ | $P_7$ | $P_8$ | $P_9$ |
|---|---|---|---|---|---|
| 0 | | 9 | -1 | -6 | 0 |
| 1 | $b_3$ | -1 | -2 | 1 | 0 |
| 2 | $b_4$ | 3 | -3 | 7 | 0 |
| 3 | $b_5$ | 5 | 1 | -1 | 0 |
| 4 | $b_6$ | 3 | 1 | -5 | 0 |
| 5 | $b_1'$ | 1 | -1 | 2 | 0 |
| 6 | $b_2'$ | 1 | 0 | -1 | 0 |
| 7 | | -1 | -1 | 0 | 1 |

Table 7.6 Fifth Step of Example 7.1

| i | Basis | $P_0$ | $P_9$ | $P_8$ | $P_{10}$ |
|---|---|---|---|---|---|
| 0 | | 10 | | | |
| 1 | $b_3$ | 1 | | | |
| 2 | $b_4$ | 6 | | | |
| 3 | $b_5$ | 4 | | | |
| 4 | $b_6$ | 2 | | | |
| 5 | $b_1'$ | 2 | | | |
| 6 | $b_2'$ | 1 | | | |
| 7 | | | | | |

# BIBLIOGRAPHY

[1] Balas,E. *A Note on the Branch-and-Bound Principle*. Oper.Res.16,442-445, errata Oper.Res.16,886 (1968)

[2] Jonathan F.Bard, *Practical Bilevel Optimization; Algorithms and Applications*, Kluwer Academic Publishers (1998)

[3] Bertier,P.,and Roy,B. *Une Procedure de Resolution Pour Une Classe de Problemes Pouvant Avoir un Charactere Combinatoire.* ICC (Int.Comp.Cent.). Bull.4,19-28 (Trans. By W.S.Jewell,ORC Rep.67-34,Univ. of California,Berkeley (1965)

[4] Charnes,A.W., Cooper,W. and Henderson,A. *Introduction to Linear Programming*, John Wiley&Sons, Inc., New York (1953)

[5] L.Cooper and S.Steinberg. *Methods and Applications of Linear Programming*, Saunders (1974)

[6] Dantzig,G.B. *Maximization of a Linear Function of Variables Subject to Linear Inequalities*, Chap.21 in Koopmans [16]

[7] Dantzig,G.B.,Fulkerson,D.R., and Johnson,S.M. *Solution of a Large Scale Traveling Salesman Problem.* Oper.Res. 2, 393-410

[8] Dantzig,G.B., and Orden, A. *Duality Theorems*, RAND Report RM-1265, The RAND Corporation, Santa Monica, CA, October (1953)

[9] *Directorate of Management Analysis, Symposium on Linear Inequalities and Programming,* A.Orden and L.Goldstein (eds.), DCS/Comptroller, Headquarters U.S.Air Force, Washington, D.C., April (1952)

[10] Saul Gass, *Linear Programming: Methods and Applicatrions*, Fifth Edition, McGraw-Hill Book Company (1985)

[11] Arthur M. Geoffrion, *Integer Programming by Implicit Enumeration and Balas' Method*, SIAMReview, Vol 9, Issue 2, 178-190 (1967)

[12] Glover, F., *Stronger Cuts in Integer Programming*, Oper.Res. 15,1174-1176 (1967)

[13] Gomory,R.E., *Outline of an Algorithm for Integer Solutions to Linear Programs*, Bull. Amer.Math.Soc. 64, 275-278, (1958)

[14] Gomory, R.E., *An Algorithm for the Mixed Integer Program*, RMM-2597. RAND Corp. Santa Monica, California

[15] Gomory,R.E., *All-Integer Programming Algorithm*, pp. 193-206 in J.F. Muth and G.L. Thompson (eds), 'Industrial Scheduling", Prentice Hall Inc., Englewood Cliffs, NJ, (1963)

[16] Koopmans,T.C., (ed.), *Activity Analysis of Production and Allocation*, Cowles Commission Monograph 13, John Wiley & Sons, Inc., New York, (1951)

[17] A.H.Land and A.G. Doig, *An Automatic Method for Solving Discrete Programming Problems*, Econometrica 28, 497-520, (1960)

[18] J.D.C. Little, K.G. Murty, D.W. Sweeney and C. Karel, *An Algorithm for the Traveling salesman problem*, Oper.Res. 11,972-989, (1963)

[19] Markowitz, H.M. and Manne, A.S., *On the Solution of Discrete Programming Problems*, Econometrica 25, 84-110, (1957)

[20] Mitten, L.G., *Branch-and-Bound Methods: General Formulation and Properties,* Oper. Res. 18, 24-34, (1970)

[21] Orden,A., *Application of the Simplex method to a Variety of Matrix Problems*, pp.28-50 of Directorate of Management Analysis [6]

[22] Linus Schrage, *Optimization Modeling with LINDO*, Fifth Edition, Duxbury Press, (1997)

[23] Wayne L.Winston, *Operation Research: Applications and Algorithm*, Duxbury Press, (1994)

[24] Laurence A.Wolsey, *Integer Programming*, A Wiley-Interscience Publication, John Wiley and Sons, Inc., (1998)

# APPENDIX

# LINDO COMPUTER PACKAGE FOR BOTH A LINEAR PROGRAMMING AND AN INTEGER PROGRAMMING

## A.1 LINDO for a Linear Programming

The material of this appendix is based substantially on the book of Linus Schrage [22] since the text of this book has been updated to reflect the availability of the Windows version of LINDO. Note, The acronym of LINDO stands for Linear, INteractive, Discrete Optimizer. The main purpose of LINDO is to use quickly input of a Linear Programming information and solve it.

Linear programming is a mathematical procedure for determining the optimal allocation of resources that are in short supply within some criteria of constraints. Resources, that are in short supply, may well range from raw materials such as ore, timber, foodstuffs, inventory control, transportation, human resources, available housing, capita, etc. The list goes on and on.

Accordingly, linear programming basically starts when you as a user want to know something. You want to know the answer to something as simple as the following objective function:

MAX $2x + 5y$ Maximize a simple functional relationship

LINDO is user friendly!

Every model in LINDO must start either MAX or MIN. Either of these two words, when entered at the command line, indicates to LINDO that a new model is being entered. Complete the objective function with the command ST, S.T or an explicit SUBJECT TO.

Note: The MAX or max and MIN or min commands erase any current model in memory and any current solution. The new model you enter becomes the current model. Note the use of the lower case letters in the following example:

An example using MAX command to input a small model flows:

max $20x + 30y$

st $x \le 50$

$y \le 60$

$x + 2y \le 120$

end

```
LP optimum found at step 2

Objective function value = 1)   2050.000

Variable        Value           Reduced Cost

    X           50.000000             0.000000

    Y           35.000000             0.000000

Row             Slack or Surplus            Dual prices

2)              0.000000                    5.000000

3)              25.000000                   0.000000

4)              0.000000                    15.000000

No. iterations = 2
```

**Model Syntax**

Fortunately, the list of rules for LINDO syntax is rather short and very easy to learn. There are only 9 rules that illustrate the basic syntax of the LINDO modeling language:

- Objective Function Syntax: Start all models with MAX or MIN or max or min.
- Variable Names: Limited to 8 characters. This is an unfortunate DOS artifact!
- Constraint Names: Terminated with a parenthesis. To name a constraint you must name your constraint with its name terminated with a right parenthesis. After the right parenthesis, you enter the constraint as before. An example of the following is given as: ABOUND) x < 10

  XBOUND) $x_1 + x_2 + x_3 \geq 15$

- If you wish, the end of the objective function and the beginning of the constraints is signified with any of the following commands will work:

  SUBJECT TO

  S.T.

  ST

  s.t.

st

Then type the variable constraints as they are modeled:

| | |
|---|---|
| Constraint 1 | $4x + 3y <10$ |
| * | * |
| * | * |
| * | * |
| Constraint N | $3x + y < 12$ |

Followed by an End

End

- Recognized Operators (+, -, >, <, =)
- Order of Precedence: **Parentheses not recognized**
- Adding Comment: Start with an exclamation mark!
- Splitting lines in a model: Permitted in LINDO
- Case Sensitivity: LINDO has none
- Right-hand Side Syntax: **Only constant values**
- Left-hand Side Syntax: **Only variables and their coefficients**

**Windows File Menu**

This menu contains commands, which generally pertain to the movement of the files and data, program input and file generation output of LINDO. For more information, please click on a command on the graphic or on a corresponding link below.

Windows Menu Commands in brief:

| **Command** | **Description** |
|---|---|
| **File** | Opens the Windows File Menu |
| **New** | Creates a new, blank Model Window |
| **Open** | Opens a model from a disk file |

| | |
|---|---|
| **View** | Opens a “View Only” model from disk |
| **Save** | Saves the active Model Window to disk |
| **Save as** | Saves a Model Window to disk and prompts for a new name |
| **Close** | Closes the active window |
| **Print** | Prints the contents of the active window |
| **Printer Setup** | Used to configure your printer |
| **Log Output** | Opens or closes a log file used for recording the results of your session |
| **Take Commands** | Runs a LINDO script file (*.ltx) |
| **Basis Read** | Loads a solution from disk into the active model |
| **Basis Save** | Saves the active model’s current solution to disk |
| **Title** | Displays the title of the active model |
| **Date** | Displays the date |
| **Elapsed Time** | Displays the time that has elapsed since the start of the session |
| **License** | Allows input of a new password to upgrade your copy of LINDO |
| **Exit** | Exits LINDO |
| **Edit** | Opens the commands that, in general, support the full screen editing options |
| **Undo** | Undoes the last edit operation |
| **Cut** | Removes selected text from window and places it in clipboard |
| **Copy** | Copies selected text to clipboard |
| **Paste** | Pastes clipboard’s contents into an active window |
| **Clear** | Removes selected text from window |
| **Find/Replace** | Finds a given string of text in active window and optionally allows for replacement |
| **Options** | Used to configure LINDO options |

**Go to Line** Jumps to a given line number in active window

**Paste Symbol** An editing tool that allows you to paste variable names and reserved symbols into active window

**Select All** Selects all the text in active window

**Clear All** Clears all the text in active window

**Choose New Front** Resets font in active window

**Solve** Opens the commands to invoke the core LINDO solver. Solves model in active window

**Compile Model** Compiles model in active window

**Debug** Debugs model in active window if it is infeasible or unbounded

**Pivot** Performs a simplex iteration, or pivot, on the model in active window

**Preemptive Goal** Uses Lexico optimization (a form of goal programming) on active window

**Reports** Opens the commands to generate reports related to your model

**Solution** Creates a solution report for active window

**Range** Creates a range (sensitivity) analysis report for the active window

**Parametrics** Performs parametric analysis on the right-hand side of a constraint

**Statistics** Displays statistics for the model in active window

**Peruse** Used to generate text reports on and/or graphic of selected items of the active model

**Picture** Creates a text and/or graphics "picture" of the nonzero structure of the current model

**Basis Picture** Creates a text-based "picture" of the basis matrix for active model

**Tableau** Displays the simplex tableau for the active model

**Formulation** Displays the active model

**Show Column** Displays a column/variable of the active model

**Positive Definite** Determines if the constraint matrix of a quadratic programming model is positive definite.

### The Model Formulation Process

- Understand the real problem
- Formulate a model for the problem
- Gather and generate the input data for the model, e.g., per unit cost basis, etc
- Solve or run the model, use graphical analysis when possible
- Check and test, use iteration to develop an appropriate model
- Implement the solution

### The Good Model Formulation Process

*Your success depends onto a large extent upon being very familiar with the organization involved.* Your input data is critical. For example, it helps to know who knows what the real production rate is on the punch press machine, etc.

Formulating good linear programming models is considered an art boarding on a science according to the author Linus Scharage. It is an art because it always involves an approximation of the real world.

### Approaches to Model Formulation

1. Constructive approach
2. Template approach

The constructive approach is more fundamental and general. The template approach utilizes an existing template of a model formulation in which you might make a modest change.

Basically, the constructive approach follows a three-step model formulation process:

- Identify and define the decision variables. Defining a decision variable includes specifying the units of which the variable is measured, e.g., tons, meters, hours, etc.
- Specify and define the objective function, including the units in which it is measured.
- Specify the constraints, including the units in which the variable is measured.

Some helpful questions (or hints) to ask yourself in defining the decision variables:

- What should be the format of a report, which gives a solution to the problem? What numbers constitute the answer, amounts of ingredients to use, number of people hired for various shifts, amount to produce such as cars, engines or polymer?
- What is the objective? Among all the usable solutions, **ask yourself how I would** measure a preference or minimize a loss or maximize a profit?
- What are the constraints? A way the author suggests is to think about constraints as: Given the purported solution to a problem, **what numeric checks would you** perform to test the correctness of your solution? There are a number of poits to note here:

- A common problem with model formulation is to overlook some constraints or variables and the entire formulation process should be regarded as an iterative one (iterating back and forth between variables/constraints/objective until we are satisfied).
- The mathematical problem given above has the form
  - All variables are continuous (i.e. can take fractional values)
  - A singe objective (maximize or minimize)
  - The objective and constraints are linear, i.e. any term is either a constant or a constant multiplied by an unknown (e.g. 127, 20x, 3y are linear terms but **xy** is a non-linear term).
  - Any model formulation, which satisfies these three conditions, as stated above, is called a linear program (LP).
- It is implicitly assumed that it is permissible to work in fractions of: days, hours and production units – there are problems where this is not permissible and variables must take integer values will be dealt with under *integer programming* (IP).
- Frequently, the decision variables should be integer, but for reasons of simplicity we let them be fractional. This is especially relevant in problems where the values of the decision variables are large because any fractional part can then usually be ignored (note that often the data (numbers) that we use in formulating the LP will be inaccurate anyway).

- The way the complete mathematical representation of the problem is set out above is the standard way (with the objective first, then the constraints and finally the reminder that all variables ≥ 0).

**Problem Definition Analysis**

*A Characteristic Problem*

A problem faced by railroads is that of assembling engine sets for particular trains. There are three important characteristics associated with each engine type, namely, operating cost per hour, horsepower and tractive power. Associated with each train, e.g. the Super Chief run from Chicago to Los Angeles is a required horsepower and a required tractive power. The horsepower depends largely upon the speed required by the run, whereas the tractive power required depends largely upon the weight of the train and the steepness of the grades encountered on the run. For a particular train the problem is to find the right combination of engines that satisfies the horsepower and tractive power requirements at lower cost.

In particular, consider the Cimarron Special that runs from Omaha, NE to Santa Fe, NM. This train requires 12,000 horsepower and 5,000 tractive power units. Two engine types are available, the GM-I and the GM-II, for pulling the train. The GM-I has 2,000 horsepower and 10,000 tractive power units, and its variable operating costs are $150 per hour. The GM-II has 3,000 horsepower and 10,000 tractive units, and its variable operating costs are $180 per hour. The engine set maybe mixed, e.g. two GM-I's and three GM-II's.

The Cimarron Special that runs from Omaha, NE to Santa Fe, NM. The train requires 12,000 horsepower and 50,000 tractive power units.

| | Engine type GM-I | Engine type GM-II |
|---|---|---|
| Horsepower | 2,000 | 3,000 |
| Tractive power | 10,000 | 10,000 |
| Costs/hour | $150 | $180 |

A linear program mode could be written as, LP-formulation of the program could be:

$$\text{MIN} \qquad Z = \$150(GMI) + \$180(GMII)$$

ST

$$2{,}000(GMI) + 3000(GMII) = 12{,}000$$

```
                          10,000(GMII) + 10,000(GMII) = 50,000

          END
```

Typical program output of LINDO follows:

```
        Objective Function Value   1) $810.0000

    Variable          Value         Reduced Cost

        GMI           3.000000      0.000000

        GMII          2.000000      0.000000

    Row               Slack or Surplus            Dual Prices

        2)            0.000000                    -0.030000

        3)            0.000000                    -0.009000

    No. Iterations = 0
```

Ranges in Which the Basis is Unchanged:

```
                            Obj Coefficient Ranges

    Variable          Current            Allowable            Allowable

                        Coef               Increase             Decrease

        GMI           150.000000         Infinity      Infinity

        GMII          180.000000         Infinity      Infinity

                      Right-hand Side Ranges

    Row               Current            Allowable            Allowable

                        RHS                Increase             Decrease

        2             12000.000000 2999.999756  1999.999878

        3             50000.000000 10000.000000 10000.000000
```

The solution is:

- The Cimarron Special, that runs from Omaha, NE to Santa Fe, NM.
- This train requires 12,000 horsepower and 50,000 tractive power units.
- The Cimarron Special should be made up of 3 GM-I engines and 2 GM-II engines.
- The horsepower and tractive power requirements at lowest expense; fee \$810/hour.

*Another Characteristic Problem*

The surgical unit of a small hospital is becoming more concerned about finances. The hospital cannot control or set many of the important factors that determine its financial health. For example, the length of stay in hospital for a given type of surgery is determined in large part by government regulation. The amount that can be charged for the given type of surgical procedure is controlled largely by the combination of the market and government regulations. Most of the hospital's surgical procedures are elective, so the hospital has considerable control over which patients and associated procedures are attached to the hospital. The surgical unit effectively has two scarce resources, the hospital beds are available to it (70 in a typical week), and the surgical suite hours available (165 hours in a typical week). Patients admitted to the surgical unit can be classified into three categories:

| Patient Type | Days of Stay | Surgical Suite Hours Needed | Financial Contribution |
|---|---|---|---|
| A | 3 | 2 | \$240 |
| B | 5 | 1.5 | \$225 |
| C | 6 | 3 | \$425 |

For example, each type B-patient admitted will use 5 days of the $7 \times 70 = 490$ bed days available each week, and 1.5 hours of the 165 Surgical Suite hours available each week. One doctor has argued that the surgical unit should try and admit more type A-patients. Her argument is that "in terms of dollars per days of stay type A-patient is clearly the best, whereas in terms of \$/(Surgical Suite hours), it is much more worse for patients of type B and C." Is the doctor correct?

Suppose the surgical unit can in fact control the number of each type of each patient each week; i.e., they are decision variables. How many of each type should be admitted each week?

```
MAX  240 A + 225 B + 425 C

ST     3 A + 5 B + 6 C ≤ 490

         2 A + 1.5 B + 3 C ≤ 165

END

GIN 3                          General integer format for first three variables
```

```
Objective Function Value  1)  $24,375.00

Variable                 Value           Reduced Cost

A                         0.000000       -240.000000

B                       80.000000        -225.000000

C                       15.000000        -425.000000

Row          Slack or Surplus               Dual Prices

2)           0.000000                       0.000000

3)           0.000000                       0.000000
```

The solution is:

- The Surgical Unit should try not to admit any type A-patients, admit only 80-type B and 15-type patients/week.
- The maximum monetary profits to be reduced at $24,375/week.

## INTEGER PROGRAMMING –IP

Two kinds of integer variables (INT) are recognized by LINDO: zero/one variables (binary) and general integer variables. Zero/one variables are restricted to the values 0 or 1. They are useful for representing go/no-go type decisions. General integer variables (GIN) are restricted to the non-negative integer values (0,1,2,…). **GIN** and **INT** statements are used respectively to identify general and binary integer variables. The statements should appear after the END statement in your model.

Variables that are restricted to the values 0 or 1 are identified using the (INT) Integer statement. It is used in one of three forms:

INTEGER <Any Variable Name>

or INTEGER <N> or INT <N>

The first (and recommended) form identifies variable <Variable Name> as being 0/1. The second form identifies the first N-variables in the current formulation as being 0/1. The third is a syntax contraction of the second form. The order of the variables is determined by their order encountered in the model. The order can be verified by observing the order of variables in the solution report. The second form of this command is more powerful because it allows the user to identify several integer variables with one line, but it requires the user to be aware of the exact order of the variables. This may be confusing if not all variables appear in the objective. If there are several variables, but they are scattered throughout the internal representation of the model, the variables would have to be identified as integers on separate lines.

General integer variables are identified with the GIN statement. Otherwise, the GIN is used exactly as the INT command. For example, GIN 4 makes the first 4-variable general integer. GIN TONIC makes the variable TOBIC a general integer variable. The solution method used on an integer program (IP) is the branch-and-bound search process. It will typically find a sequence of better and better solutions. If LINDO is in the default (verbose) output mode, then a log of the search process will be displayed. If you have placed LINDO into terse output mode from either the Options command in Windows versions ot eh Terse command in command-line versions, then only a short message is printed as each better solution is found in the branch-and-bound search.

Once the branch-and-bound enumeration process is completed, the best solution found is reinstalled. Thus, report-generating commands can be used to examine the best solution. Note, however, that the reduced costs and dual prices resulting from an integer model are essentially meaningless to the casual user and, therefore, should be disregarded. General integer variables are represented directly. LINDO does not use the so-called binary expansion representation of a general integer variable as a sum of 0/1 variables. For most practical problems, the binary expansion method increases solution times prohibitively.

Integer programming method may be used when one or more variables can take only integral values. Examples are the number of trucks in the fleet, the number of generators in powerhouse, batch sizing, job sequencing, and so on. Approximate solutions can be obtained without using integer-programming models, but the approximation generally becomes poorer as the numbers become smaller. Integer programming is often a way to attack many discrete optimization problems, i.e., optimization problems in which some of the variables are constrained to lie in a discrete set.

### Tightening Loose IP Formulations

Because IP problems tend to be difficult to solve, it is important that they are "tight." Loosely speaking, this means when the formulation is solved as an LP, the solution should look a lot like the IP solution (e.g., many of the IP variables are naturally integer and the LP objective value is approximately equals to the IP objective value).

LINDO has a command-line command, TITAN, which will do some of this tightening. It will do two things:

1) Deduce tighter the upper and lower bounds for continuous variables and then using this information

2) Reduce the coefficients of integer variables where justified.

The net effect of 2) is that when the problem is solved as an LP, the non-zero IP variables will be closer to 1 than they would otherwise have been. For example, if you have the constraint: $7x + 4y + 2z \leq 5$, where all variables have a lower bound of zero and X is declared to be an integer variable, then TITAN will transform this constraint to: $5x + 4y + 2z \leq 5$.

It is important to note that the student version unfortunately will not allow a partial demonstration using the TITAN command. The TITAN command is a regular part of the LINDO command set.

### Optimization Modeling – Integer Programming

We want to decide how to place 7 songs on a record album so as to maximize the number of songs on the "short" side of the album. The short side must contain no more than half the total music time. The times in minutes by songs are:\

SONG: 1 2 3 4 5 6 7

TIME: 2 5 2 3 7 2 2 minutes

Y<i> = 1 if song <i> is assigned to short side, else 1

MAX Y1 + Y2 + Y3 + Y4+ Y5 + Y6 + Y7

SUBJECT TO

$$2\ Y1 + 5\ Y2 + 2\ Y3 + 2\ Y4 + 7\ Y5 + 2\ Y6 + 2\ Y7 \leq 11$$

END

The Y's must be 0/1:

INT 7 Integer command format the first seven variables as being a 0/1.

In consecutive **Tableau** and **Pivot** form:

```
The Tableau
Row  (Basis)   Y1      Y2      Y3       Y4     Y5      Y6      Y7    SLK  2
   1   ART     -1.000  -1.000  -1.000  -1.000 -1.000 -1.000 -1.000 0.000  0.000
   2   SLK  2  2.000    5.000   2.000   2.000  7.000  2.000  2.000 1.000 11.000
ART     ART  -1.000  -1.000 -1.000   -1.000 -1.000 -1.000 -1.000 0.000  0.000
Pivot:  Y1 Enters at Value 1.0000  in Row -1 Obj. Value = 1.0000
The Tableau
Row  (Basis)   Y1      Y2      Y3       Y4     Y5      Y6      Y7    SLK  2
   1   ART      1.000  -1.000 -1.000  -1.000 -1.000 -1.000  -1.000 0.000 1.000
   2   SLK 2 -2.000   5.000   2.000   7.000  2.000  2.000   1.000  9.000 7.000
Pivot:  Y3 Enters at Value  1.0000  in Row  -1 Obj. Value = 2.0000
The Tableau
Row  (Basis)   Y1      Y2      Y3       Y4     Y5      Y6      Y7    SLK   2
   1  ART    1.000  -1.000  1.000  -1.000 -1.000  -1.000  -1.000   0.000 2.000
   2  SLK 2  -2.000   5.000  -2.000   2.000  7.000  2.000   2.000  1.000 7.000
Pivot:  Y4 Enters at Value  1.0000  in Row  -1 Obj. Value = 3.0000
The Tableau
Row  (Basis)   Y1      Y2      Y3       Y4     Y5      Y6      Y7    SLK   2
   1  ART    1.000  -1.000  1.000   1.000 -1.000  -1.000  -1.000   0.000 3.000
   2  SLK 2  -2.000   5.000  -2.000  -2.000  7.000  2.000  2.000  1.000  5.000
Pivot:  Y6 Enters at Value  1.0000  in Row  -1 Obj. Value = 4.0000
```

```
The Tableau

Row  (Basis)   Y1      Y2      Y3       Y4     Y5       Y6      Y7    SLK   2

     1  ART   1.000  -1.000  1.000   1.000  -1.000   1.000  -1.000   0.000  4.000

     2  SLK 2 -2.000   5.000  -2.000  -2.000  7.000  -2.000   2.000   1.000  3.000

Pivot: Y7 Enters at Value  1.0000  in Row  -1 Obj. Value = 5.0000

The Tableau

Row  (Basis)   Y1      Y2      Y3       Y4     Y5       Y6      Y7    SLK   2

     1  ART   1.000  -1.000  1.000   1.000  -1.000   1.000   1.000   0.000  5.000

     2  SLK 2 -2.000   5.000  -2.000  -2.000  7.000  -2.000  -2.000   1.000  1.000

Pivot: Y2 Enters at Value  0.20000  in Row  2 Obj. Value = 5.2000

The Tableau

Row  (Basis)   Y1      Y2      Y3      Y4      Y5      Y6      Y7    SLK    2

1  ART           0.600  0.000   0.600    0.600   0.400    0.600  0.600   0.200    5.200

2        Y2     -0.400 1.000   -0.400   -0.400  1.400   -0.400  -0.400  0.200    0.200

LP Optimum Found at Step 6

Objective Value = 5.1999981

New Integer Solution Of   5.00000000            at Branch  0      Pivot  6

Objective Function Value  1) 5.000000

Variable     Value           Reduced Cost          Row    Slack or Surplus    Dual Prices

   Y1         1.000000   -1.000000              2)        1.000000                0.000000

   Y2         0.000000   -1.000000

   Y3        1.000000    -1.000000              Number of Iterations = 6

   Y4        1.000000    -1.000000              Branches = 0

   Y5        0.000000    -1.000000              Determ. = 1.000E 0
```

```
Y6        1.000000   -1.000000

Y7        1.000000   -1.000000
```

**Extreme Programming Examples**

*Example of* ***Feasible*** *Model as Follows*:

Results show an Optimal Solution **output** as:

```
LP Optimum Found at Step 1 Objective Function Value  1) 16.66667

Variable        Value           Reduced Cost

  X             0.000000        4.666667

  Y             3.333333        0.000000

Row             Slack or Surplus                Dual Prices

2)              0.000000                        1.666667

3)              8.666667                        0.000000

No. Iterations = 1

Ranges in which the basis is unchanged:

                        Obj Coefficient Ranges

Variable                Current         Allowable       Allowable

                         Coef            Increase        Decrease

  X                     2.000000        4.666667        Infinity

  Y                     5.000000        Infinity        3.500000

                        Right-Hand Side Ranges

Row                     Current         Allowable       Allowable

                         RHS            Increase         Decrease

2                       10.000000       26.000000       9.999999

3                       12.000000       Infinity        8.666667
```

# VITA

Djavanshir Djebrail Gadjiev attended the Azerbaijan S. M. Kirov State University named after M. E. Rasulzade (formerly USSR), where he got the degrees of the Bachelor of Arts in Mathematics Education and  Master of Science in Mathematics. Upon graduating from the University in 1976 he had a job teaching high school math in Baku, capital of the Azerbaijan Republic (formerly USSR). After several years of teaching, he decided to further his education by attending the Graduate School of the Academy of Sciences of the USSR to extend the degree in Mathematics. He graduated from the Graduate School in 1983 and upon his research work in the area of Optimization and Differential Equations he defended a dissertation in 1988 at the Academy of Sciences of the USSR, Moscow (present Russia).

In 2000-2002 he taught the College Algebra as a lecturer at the Mathematics Department of the University of Tennessee, Knoxville. In 2002 he decided to attend the University of Tennessee, Knoxville by accepting a teaching associate position.

He graduated on December, 2003 from the University of Tennessee, Knoxville with a Master of Science degree in Mathematics.