To the Graduate Council:

I am submitting a thesis written by Jayosree Mukherjee entitled "Transportation Problems and Assignment Problems." I have examined the final 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:

Suzanne Lenhart

Accepted for the Council:

Associate Vice Chancellor and
Dean of the Graduate School

# TRANSPORTATION PROBLEMS
# AND
# ASSIGNMENT PROBLEMS

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

**Jayosree Mukherjee**
**May, 2000**

# ACKNOWLEDGEMENT

I take this opportunity to thank my committee chair Dr. Y. Kuo for her patience, advice and guidance to complete this thesis. Without her encouragement, it would not have been possible to complete this thesis. As I was working in Atlanta for the last few months, she kept on seeing me in her office on almost every weekend. No words will be sufficient to express my gratitude to Dr. Kuo. I also thank Dr. S. Lenhart and Dr. X. Feng for spending their valuable time to read my thesis. Last but not the least; I thank my family who has been very supportive to me during this endeavor.

# Abstract

Special computation techniques have been developed for certain types of linear programming problems that are widely used in mathematical modeling of real life situations. One of the most useful of these special-purpose algorithms is transportation method. In a classical transportation problem the cost is minimized by selecting the optimal path for transporting goods from a set of origins to a set of destinations. Assignment method is a special case of more general transportation problem. A discussion of linear programming in general, theory supporting transportation method, methods of finding optimal solution, applications of transportation and assignment problems are presented. Transportation and assignment methods are found to offer significant computational efficiency over the simplex method for certain types of linear problems.

# TABLE OF CONTENTS


| | | PAGE |
|---|---|---|
| CHAPTER 1 | INTRODUCTION .. . . . .... . . . .... . ... . . . . | 1 |
| CHAPTER 2 | THE LINEAR PROGRAMMING PROBLEM AND THE SIMPLEX METHOD.. . . . ..... . .. ... . .. . | 13 |
| 2.1 | EXAMPLE OF A LINEAR PROGRAMMING PROBLEM. . ... . . .... ... . .... ..... ... . . | .....13 |
| 2 2 | SIMPLEX METHOD . . .... . . . . . . ... ... . | 14 |
| 2.3 | THEORETICAL PROOF OF SIMPLEX METHOD... . . . | 16 |
| 2 4 | EXAMPLE OF SIMPLEX METHOD . ... .. .. .. . . | ...20 |
| 2.5 | THE TRANSPORTATION PROBLEM AS A LINEAR PROGRAMMING PROBLEM. . . . . | 24 |
| CHAPTER 3 | THE INITIAL FEASIBLE SOLUTION FOR BALANCED TRANSPORTATION PROBLEM.. . . . . .. . | 27 |
| 3 1 | THE NORTHWEST CORNER METHOD . . ..... . ... | 36 |
| 3.2 | HERTLEY'S CURRENT LINE METHOD . . . | 39 |
| 3.3 | THE MINIMUM ENTRY OR SHORTCUT METHOD. .. | 41 |
| 3 4 | VOGEL'S ADVANCED START METHOD. . . . ... | 44 |
| 3 5 | RUSSEL'S APPROXIMATION METHOD... .. ... .. .. | 47 |
| CHAPTER 4 | AN OPTIMAL SOLUTION TO BALANCED TRANSPORTATION PROBLEM . . . . . .... ... . . | 52 |
| 4 1 | THE TRANSPORTATION PROBLEM HAS AN OPTIMAL SOLUTION. ... .. . . . .. ..... . ... . . | 53 |
| 4 2 | THE STEPPING STONE METHOD . . . ... . . ..... | 65 |

4 3 THE U-V METHOD . . . . . 80

CHAPTER 5 ASSIGNMENTPROBLEMS. . . . . 84

5.1 TRANSPORTATION ALGORITHM. . . . . 86

5.2 THE ASSIGNMENT PROBLEM AND THE HUNGARIAN METHOD . . . . . 99

5.3 CONCLUSION . . . . . 108

REFERENCES . . . . . 111

APPENDIX . . . . . 114

VITA . . . . . 128

# CHAPTER 1
# INTRODUCTION

In this paper we have attempted to make a detailed study of a specific problem in linear programming called transportation problem. This particular topic has been selected since transportation problems are widely used for developing mathematical models for many real life problems in the field of engineering, science and business. At the beginning of the paper, we shall examine the theoretical support and method for solving the classical linear problem. We shall then proceed to discuss various aspects of transportation problem Before discussing the various methods of finding a basic feasible solution of a transportation problem, we shall establish with theoretical support that a feasible solution exists for a transportation problem The different methods are described in details with example problems, and compared to determine the most suitable method for finding the initial basic feasible solution We shall continue with giving theoretical support that indicates there are two techniques to find out the optimal solution In the next chapter, we discuss a specific type of transportation problem, called assignment problem and some additional applications The

paper is concluded with my observations

An important class of linear programming problem can be formulated in terms of a network composed of points connected by routes, over which various kinds of transportation take place The classical transportation problem arises when we must determine an optimal schedule of shipment with following conditions.

1 Shipments originate at sources where fixed stockpiles of a commodity are available,

2 Shipments are sent directly to their final destinations where various fixed amounts are required,

3 The stockpiles will be exhausted and the demand will be fulfilled.

4 Finally the cost must satisfy a linear objective function, that is, the cost of each shipment is proportional to the amount shipped, and the total cost is the sum of the individual costs [2]

L V Kantorovich in 1939 showed that a class of problems closely related to the classical transportation case has a remarkable variety of applications concerned typically with the allotment of tasks to machines whose cost and rates of production vary by task and machine type. He gave us useful but

incomplete algorithm for solving such problems In 1948, he authored an application study, jointly with Gavurin, on the capacitated transportation problem. Frank L Hitchcock was the first man who described the original form of transportation problem T.C. Koopsmans in 1949 developed the transportation problem with more details George B. Dantzig first stated the linear programming formation of the transportation problem in 1951. Several variations of solving the transportation problem have been developed since. Among these are the northwest-corner method, the short-cut method, and Russell's approximation [3] Based on this investigation, Kuhn developed an efficient algorithmic method for solving assignment problems in 1955

During World War II, T C Koopman was the member of the Combined Shipping Board. He used the solutions of the transportation problem to help reduce over-all shipping times, for the shortage of cargo ships constituted a critical bottleneck The solution of transportation problem is used in many diverse areas, such as finance, nutrition, economics, and waste disposal. The main source of above background information is from Dantzig [2].

The following notational conventions will be used throughout this

paper· All matrıces will be denoted by capıtal letters and vectors will be denoted by small and bold face. The symbols I and 0 represent the identity matrix and the matrıx wıth all zero entrıes respectıvely The ith component of a vector x will be denoted $x_i$ throughout the paper, we will use vector as a row when I appears on the left of a multiplication, and as a column when it appears on the rıght of the multiplicatıon. We named an equatıon or system by use of parenthesis and numbered, i.e.(1.1)

**The general transportation problem can be stated as follows:**

There are some special techniques, which have been developed to solve some certaın kınd of lınear programmıng problems. The transportatıon algorıthm ıs one of such specıal techniques It is to minimize the transportatıon cost between sources to dıfferent destınatıons A homogeneous product ıs to be shıpped in the amount $a_1$, $a_2$, . $a_m$, respectıvely, from each of m shıppıng orıgins and received in amounts $b_1$, $b_2$, .. $b_n$, respectively, by each of n shıppıng destinations. The cost of shipping an unıt amount from the ıth orıgın to jth destınatıon ıs $c_{ij}$ and ıs known for all combinatıons (ı,j) The problem ıs to determıne the amounts $x_{ij}$ shipped over all routes (ı,j) so as to mınımıze the total cost of transportatıon. The total amount shipped form orıgın i ıs $a_i \geq 0$ and the total amount receıved by

destination j is $b_j \geq 0$ Find the values for the variable $x_{ij}$ which minimize the total cost

$$\sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij} \qquad (1\ 1)$$

subject to the constraints

$$\sum_{j=1}^{n} x_{ij} = a_i \qquad (1.2)$$

$$\sum_{i=1}^{m} x_{ij} = b_j \qquad (1\ 3)$$

and

$$x_{ij} \geq 0 \qquad (1.4)$$

In order for equations (1.2) and (1 3) consistent, we must have the sum of equations (1 2) equal to the sum of equations (1 3), that is

$$\sum_{i=1}^{m}\sum_{j=1}^{n} x_{ij} = \sum_{j=1}^{n}\sum_{i=1}^{m} x_{ij} = \sum_{i} a_i = \sum_{j} b_j = A$$

We note that the system of equations (1 1) to (1.4) is a linear programming problem with m+n equations in mn variables.

For m=3 and n=5, if we write out the equations that correspond to (1.2) and (1 3), we shall obtain the 8(that is, m+n) equations in 15 (that is mn) unknowns. If we subtract the second and third equations from the sum of the

equations fourth through eighth, the result is equation one. Equation one of the eight equations is redundant and need not be included in the system Generalizing , we note that one equation from the System (1.2) or (1 3) can be eliminated, and the transportation problem reduces to m+n-1 independent equations in mn variables. For the preceding equations we have the following matrix for the reduced system Here we have eliminated the Equation (1.2)·

$$A = \begin{array}{c} \begin{matrix} P_{11} & P_{12} & P_{13} & P_{14} & P_{15} & P_{21} & P_{22} & P_{23} & P_{24} & P_{25} & P_{31} & P_{32} & P_{33} & P_{34} & P_{35} \end{matrix} \\ \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \end{array} \qquad (1\ 5)$$

$$P_0 = \begin{pmatrix} a_2 \\ a_3 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \end{pmatrix}$$

The vector $\mathbf{P}_{ij}$ corresponds to the variable $x_{ij}$.

$X = (x_{11}, \ldots . \quad x_{ij}, \ldots, x_{mn})$ be a column vector with mn variables and

$\mathbf{c} = (c_{11}, \ldots . \quad c_{ij}, \ldots, c_{mn})$ be a row vector of cost coefficients, the reduced transportation problem can be represented as follows ·

minimize

$$z = \mathbf{c}X$$

subject to

$$AX = P_0$$

$$X \geq 0 \qquad (1.6)$$

**Definition 1.1** An optimal solution is a solution that satisfies both the conditions of the problem and given objective

**Definition 1.2** A slack variable is an unknown nonnegative number that is used to transform an inequality to an equation.

**Definitions 1.3** A linear programming problem (LP) is an optimization problem for which we do the following

1 We attempt to maximize (or minimize) a linear function of the decision variables The function that is to be maximized or minimized is called the Objective Function.

2 The value of the decision variables must satisfy a set of constraints. Each

constraint must be a linear equation or linear inequality

3. A sign restriction is associated with each variable For any variable $x_i$ the sign restriction specifies either that $x_i$ must be non-negative ($x_i >= 0$) or that $x_i$ may be unrestricted in sign .

For the actual linear programming problem the constraints may be mixed in the sense that some or all of them may be in the form of inequalities and some or all of them may hold as equations. Nevertheless, we shall always write a linear programming problem in what we shall call the " Standard Forms". The three standard forms are as follows

1 $$\text{Min } z = \sum_{j=1}^{n} c_j x_j$$

Subject to following constraint

$$\sum_{j=1}^{n} a_{ij} x_j = b_i, \quad i = 1,2,. \ldots m \qquad (1\ 7)$$

$$x_j \geq 0 \ , \quad j = 1,2, \ldots n \qquad b_i \geq 0$$

or equivalently , in matrix notation as

2 $$\text{min} \quad z = \mathbf{c}\, X$$

where $\mathbf{c} = (c_1, c_2 \ldots\ldots c_n)$ $\quad X = \begin{bmatrix} x_1 \\ x_2 \\ . \\ x_n \end{bmatrix}$

$$AX = b$$

Where $\quad A = \begin{bmatrix} a_{11} & a_{12} & . & a_{1j} & . & a_{1n} \\ a_{21} & a_{22} & .. & a_{2j} & . & a_{2n} \\ . & . & .. & . . & . & \\ a_{i1} & a_{i2} & & a_{ij} & . & a_{in} \\ & \ldots & . & .. & . & \\ a_{m1} & a_{m2} & & a_{mj} & & a_{mn} \end{bmatrix}$ and $b = \begin{bmatrix} b_1 \\ b_2 \\ . \\ b_j \\ \\ b_m \end{bmatrix}$

$$x \geq 0 \qquad (1.8)$$

3 Minimize

$$z = \mathbf{c}X$$

subject to the following constrains

$$x_1P_1 + x_2P_2 + x_3P_3 + \quad . \quad x_nP_n = P_0 \qquad (1.9)$$

where $P_j$ is the jth column of the A matrix (1.5) and $P_0 = b$

and

$$x \geq 0$$

In order to do this, it is apparent that we have somehow converted those constraints of the original problem that were stated as inequalities into some sort of equivalent equality form

Consider a typical inequality

$$\sum_{j=1}^{n} a_{ij}x_j \leq b_i \qquad (1\ 10)$$

If we add an additional non-negative variable $x_{p+1}$ to the left-hand side of the equation there we have

$$\sum_{j=1}^{n} a_{ij}x_j + x_{p+1} = b_i \qquad (1\ 11)$$

$$x_{p+1} \geq 0$$

The variable $x_{p+1}$ is referred to as a slack variable In the same way a constraint of the form

$$\sum_{j=1}^{p} a_{ij}x_j \geq b_i$$

We can write

$$\sum_{j=1}^{p} a_{ij}x_j - x_{p+1} = b_i \qquad (1\ 12)$$

In this case the variable $x_{p+1}$ is called a surplus variable.

**Definition 1.4** A convex combination of the points $U_1, U_2, \ldots, U_k$ is a point $U = \alpha_1 U_1 + \alpha_2 U_2 + \ldots \ldots + \alpha_k U_k$ where the $\alpha_j$ are scalars, $\alpha_j \geq 0$ and

$$\sum_j \alpha_j = 1$$

**Definition 1.5** A subset C of $E_n$ is convex if for all pairs of points $U_1$ and $U_2$ in C, any convex combination $U = \alpha_1 U_1 + \alpha_2 U_2$ is also in C

**Definition 1.6** A feasible solution to the linear programming is a vector $X = (x_1, x_2, \ldots \ldots x_n)^T$ which satisfies all (1.2), (1.3), (2 4) the constraints of the problem.

**Definition 1.7** A basic solution to the linear programming problem is a feasible solution with no more than m positive $x_j$ and such that the vectors $a_j$ associated with these latter $x_j$ are a linearly independent set of vectors

**Definition 1.8** A non-degenerate basic feasible solution is a basic feasible solution with exactly m positive $x_j$ Hence a degenerate basic feasible solution has less than m positive $x_j$

**Definition 1.9** A minimum feasible solution is a feasible solution which also minimizes (1.7), $z = \mathbf{c}X$

**Definition 1.10** an optimal basic feasible solution is a basic solution that satisfies conditions (1 7), (1.8), and (1 9).

**Definition 1.11** Balanced Transportation Problem

The transportation problem, as we have defined earlier was formulated to find the minimum-cost plan for shipping a homogeneous product from a number of supply points (sources) to a number of points demand (destinations). When the total supply is exactly equal to the total demand, the problem is called the balanced transportation problem. Mathematically we can say where $a_1 \quad a_m, b_1 .. \ b_n$ represent amounts shipped and received respectively from each of m origins and n destinations i and j represent various origins and destinations

$$\sum_{i=1}^{m} a_i = \sum_{j=1}^{n} b_j$$

$a_i$ and $b_j$ are non-negative by definition.

**Definition 1.12** Unbalanced Transportation Problem

When the total supply is more than the total demand or the total demand is more than the total supply, the problem is called unbalanced transportation problem

# CHAPTER 2

# THE LINEAR PROGRAMMING AND THE SIMPLEX METHOD

Before covering the transportation problem, it is necessary to discuss the linear programming problem and simplex method. Simplex method is the most popular method for solving the linear programming problem. The simplex method involves using one basic feasible solution to the linear programming problem, and systematically moves towards the optimal basic feasible solution This begin with one extreme point and move to another extreme point, until the optimal solution is found But this method is very lengthy method to solve the transportation problems

In this chapter also all matrices will be denoted by capital letters, small letters with bold face will denote vectors and other notations as defined in previous chapter The theorems and proofs stated shall be from or variations from Gass[3] and Cooper[1]

## Section 2.1 Example of a Linear Programming Problem

**Example 2.1**

$$\text{Min } z = -2x_2 + 3x_4$$

Subject to

$$- x_1 + 2x_2 \quad + 4x_4 \quad = 12$$

$$x_2 + x_3 + 3x_4 \quad = 18$$

$$x_2 \quad - x_4 + x_5 = 4$$

A feasible solution is $X = (x_1,x_2,x_3, x_4,x_5) = (6, 3, 6, 3, 4)$; a basic solution is $X = (-12, 0, 18, 0, 4)$, a nondegenerate basic feasible solution is $X = (0, 0, 9, 3, 7)$; A minimum feasible solution is $X = (0, 14/3, 34/3, 2/3, 0)$ The minimum value of the objective function is $f(x) = f(0, 14/3,34/3,2/3,0) = -2(14/3) + 0(34/3) + 2(2/3) = - 8$. The minimum feasible solution is also an optimal basic feasible solution. If a problem has an optimal feasible solution, it has an optimal basic feasible solution.

**Section 2.2 The Simplex Method**

The standard form which is mentioned in Chapter 1, for the central problem of linear programming, consists in finding values for a set of nonnegative variables that satisfies a system of linear equations and minimizes a linear form z. We distinguish between the simplex method which starts with a linear program in standard form and the simplex algorithm which starts with a canonical form, consists of pivot operations, and forms the main subroutine of the simplex method.

First we have to consider the set of m linearly independent vectors $P_1$ $P_2, \ldots \ldots .P_m$, of matrix A of (1.6)and denote the mxm matrix by B =[ $P_1, P_2,$ . $P_m$]. The matrix B is an admissible basis The vector X compute the corresponding solution and representation of the other vectors in terms of the basis. We will now redefine $P_j$ to be jth column vector, j = 1,2, n with the last column begin $P_0$. Originally we assumed all the elements of $P_0$ to be nonnegative, we have the initial basic feasible solution $X_0 = P_0$ and $X_j = P_j$ where $X_0 = (x_{10}, x_{20}, \ldots, x_{m0}) \geq 0$ and $X_j = (x_{1j}, x_{2j}, \ldots, x_{mj})$ We also need to define $z_0 = \sum_{i=1}^{m} c_i x_{j0}$ and $z_j = \sum_{i=1}^{m} c_i x_{ij}$, j = 1,2, . n

Now we can proceed to search for the optimal solution from our initial tableau

1 From the tableau, view the bottom row $(z_j - c_j) \leq 0$ for j = 1,2,3 . .n, then the solution is optimal. If $z_j - c_j > 0$, proceed to Step 2

2, Select the largest element in the bottom row, maximum$(z_j - c_j)$. This is our pivot column $P_j$.

3 The selection of the vector to be removed from the basis to ensure feasibility of the new solution The vector with min $(x_{i0}/x_{ij})$for those $x_{ij} > 0$, where j corresponds to the vector selected in Step2. If all $x_{ij} \leq 0$ then

the solution is unbounded.

The transformation of the tableau is done by the complete elimination procedure to obtain the new solution and associated elements. Each such iteration produces a new basic feasible solution.

**Section 2.3 Theoretical Proof of the Simplex Method**

Now we will give some theoretical support for the simplex method, which is taken from Gass[3] and Dantzig[2]. We shall cover a couple of examples involving the simplex method. For the theorems in this chapter we will assume that the linear- programming problem is feasible, that every basic feasible solution is non-degenerate, and that we are given a basic feasible solution. Unless otherwise stated the theorems and proofs are close variation of Gass[3]

These assumptions, as will be discussed are made without any loss in generality. Let the given solution be $X_0 = (x_{10}, x_{20}, \ldots, x_{m0})$ and the linearly independent vectors be $P_1, P_2 \ldots P_m$. Now we can write

$$x_{10}P_1 + x_{20}P_2 + \ldots x_{m0}P_m = P_0 \qquad (2\ 4)$$

and $$x_{10}c_1 + x_{20}c_2 + \ldots \ldots x_{m0}c_m = z_0 \qquad (2.5)$$

where all $x_{i0} > 0$, the $c_i$ are the cost coefficients of the objective function These are form of the linear programming problem. Now the set $P_1, P_2$ . $P_m$ are linearly independent vectors and thus form a basis Let $P_j$ be given by

$$x_{1j}P_1 + x_{2j}P_2 + \ldots\ldots . x_{mj}P_m = P_j \qquad (2.6)$$

and $$x_{1j}c_1 + x_{2j}c_2 + \ldots\ldots . x_{mj}c_m = z_j \qquad (2.7)$$

**Theorem 2.1** If for any fixed j, the condition $z_j - c_j > 0$ holds, then a set of feasible solutions can be constructed such that $z < z_0$ for any member of the set, where the lower bound of z is either finite or infinite.

Proof. Multiply(2 6) by some number $\theta$ and subtract from (2.4) Similarly multiply (2 7) by the same $\theta$ and subtract from (2 5) we get

$$(x_{10} - \theta x_{1j})P_1 + (x_{20} - \theta x_{2j})P_2 + + (x_{m0} - \theta x_{mj})P_m + \theta P_j = P_0 \qquad (2.8)$$

$$(x_{10} - \theta x_{1j})c_1 + (x_{20} - \theta x_{2j})c_2 + + (x_{m0} - \theta x_{mj})c_m + \theta c_j = z_0 - \theta(z_0 - c_j) \qquad (2\ 9)$$

where $\theta c_j$ has been added to the both sides of (2.9) If all the coefficients of the vectors $P_1, P_2 \ldots . P_m, P_j$ in (2.8)are nonnegative, then we have determined a new feasible solution whose value of the objective function is , by (2.9) $z = z_0 - \theta(z_0 - c_j)$ Since the variables in (2.8) are all positive, it is clear , that there is a value of $\theta > 0$ (finite or infinite) for which the

coefficients of the vectors in (2.8) remain positive. From the assumption that, for a fixed j, $z_j - c_j > 0$, we have $z = z_0 - \theta(z_0 - c_j) > z_0$ for $\theta > 0$ Now here is two cases· case I at least one $x_{ij} > 0$ in (2 6)for i = 1,2,. ,m, or case2, $x_{ij} \leq 0$ for all i = 1,2,.. ..m for the fixed j.

Case I If, for the fixed j, at least one $x_{ij} > 0$ in (2.6) for i = 1,2, . .m, the largest value of $\theta$ for which all coefficients of (2.8) remain nonnegative is given by

$$\theta_0 = \min_j \frac{x_{i0}}{x_{ij}} > 0 \quad \text{for } x_{ij} > 0 \qquad (2.10)$$

Since we assumed that the problem is nondegenerate, i.e., that all feasible solutions have m positive elements, the minimum in (2 10) will be obtained for a unique i If $\theta_0$ is substituted for $\theta$ in (2 8) and (2.9), the coefficient corresponding to this unique i will disappear We have then constructed a new basic feasible solution consisting of $P_j$ and m-1 vectors of the original basis. This new basis can be used as the previous one . If a new $z_j - c_j > 0$ and a corresponding $x_{ij} > 0$, another solution can be obtained which has a smaller value of the objective function This process will

continue either until all $z_j - c_j \leq 0$, or until, for some $z_j - c_j > 0$, all $x_{ij} \leq 0$. If all $z_j - c_j \leq 0$, the process ends.

Case II If we have for some j, for some $z_j - c_j > 0$, and all $x_{ij} \leq 0$ then there is no upper bound to $\theta$ and objective function has a lower bound of $-\infty$. We see for this case, for an $\theta > 0$, all the coefficients of (2.8) are positive. We then have a feasible solution consisting of m+1 positive elements. Hence, by taking $\theta$ large enough, the corresponding value of the objective function given by the right hand side of (2.9) can be made arbitrarily small. This completes the proof.

**Theorem 2.2** If for any basic feasible solution $X = (x_{10}, x_{20}, \ldots, x_{m0})$ the conditions $z_j - c_j \leq$ holds for all $j = 1, 2, \ldots n$ then (2.4) and (2.5) constitute a minimum feasible solution.

Proof: Let

$$y_{10}P_1 + y_{20}P_2 + \ldots y_{n0}P_n = P_0 \qquad (2.11)$$

and

$$y_{10}c_1 + y_{20}c_2 + \ldots\ldots y_{n0}c_n = z^* \qquad (2.12)$$

be any other feasible solution with $z^*$ the corresponding value of the objective function. We will show that $z_0 \leq z^*$. By assumption, $z_j - c_j \leq$ holds for all j, so that replacing $c_j$ by $z_j$ in (2.12) yields

$$y_{10}z_1 + y_{20}z_2 + \ldots \ldots \; y_{n0}z_n \leq z^* \qquad (2\ 13)$$

For each j we substitute the expression for $P_j$ given by (2 6) into (2.11) and we get

$$y_{10}\left(\sum_{i=1}^{m} x_{i1}P_i\right) + y_{20}\left(\sum_{i=1}^{m} x_{i2}P_i\right) + \ldots y_{n0}\left(\sum_{i=1}^{m} x_{im}P_i\right) = P_0 \text{ or}$$

$$\left(\sum_{j=1}^{n} y_{j0}x_{1j}\right)P_1 + \left(\sum_{j=1}^{n} y_{j0}x_{2j}\right)P_2 + \ldots \left(\sum_{j=1}^{n} y_{j0}x_{mj}\right)P_m = P_0 \qquad (2\ 14)$$

For each j we substitute the expression for $z_j$ given by (2.7) and (2.13) we have

$$\left(\sum_{j=1}^{n} y_{j0}x_{1j}\right)c_1 + \left(\sum_{j=1}^{n} y_{j0}x_{2j}\right)c_2 + \qquad \left(\sum_{j=1}^{n} y_{j0}x_{mj}\right)c_m \leq z^* \qquad (2\ 15)$$

Here sets of vectors $P_1, P_2 \ldots P_m$, is linearly independent, therefore

$$y_{10}z_1 + y_{20}z_2 + \ldots \ldots \; y_{n0}z_n \leq z^* \text{ and } z_0 \leq z^*$$

This proofs the theorem and verifies the simplex method.

## Section 2.4 Example of Simplex Method

Now we will work out some examples using the simplex method

**Example 2.2** Maximize

$$z = 2x_1 + 4x_2 + 2x_3$$

subject to the following constraints

$$x_1 + 2x_2 - 3x_3 \leq 5$$

$$2x_1 + 3x_2 + x_3 \leq 10$$

$$x_1 + 2x_2 + x_3 \leq 20$$

and $x_1, x_2, x_3 \geq 0$

This above problem has been defined as a maximization problem of the linear programming problem This problem can be easily changed into minimization problem by multiplying z by –1 The $z_j$ elements are obtain by taking the inner product of $P_j$ with the cost column c We can also transfer inequalities into equations by use of slack variables ($x_4, x_5$, and $x_6$) The initial basic solution generally are these slack variables. With these changes the problem becomes

Minimize

$$z = -2x_1 - 4x_2 - 2x_3$$

subject to the following constraints

$$x_1 + 2x_2 - 3x_3 + x_4 = 5$$

$$2x_1 + 3x_2 + x_3 + x_5 = 10$$

$$x_1 + 2x_2 + x_3 + x_6 = 20$$

and

$$x_i \geq 0 \text{ for } i = 1,2, \ldots 6$$

Referring to table 2 1 below , $P_4$ , $P_5$ , $P_6$ are in our initial basis We need to

augment the system with full artificial basis of three vectors. We now go through our simplex procedure There are three elements on the bottom row, which are positive, therefore we do not have an optimal solution $P_2$ becomes our pivot column with $x_{22} = 3$ becoming our pivot element R is representing the respective rows, we pivot. The matrix operation are as follows, in order· 1/3 $R_2 \Rightarrow R_2$, $-2R_2+R_1 \Rightarrow R_1$, $-2R_2+R_3 \Rightarrow R_3$,$-4R_2+R_4 \Rightarrow R_4$ We will get the result from Table 2 1

**Table 2.1 Initial Tableau of Example2.2**

| | Basis | c | $P_0$ | -2 | -4 | -2 | 0 | 0 | 0 | θ |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | $P_1$ | $P_2$ | $P_3$ | $P_4$ | $P_5$ | $P_6$ | |
| 1 | $P_4$ | 0 | 5 | 1 | 2 | -3 | 1 | 0 | 0 | |
| 2 | $P_5$ | 0 | 10 | 2 | 3* | 1 | 0 | 1 | 0 | 10/3 |
| 3 | $P_6$ | 0 | 20 | 1 | 2 | 1 | 0 | 0 | 1 | 10 |
| 4 | | | 0 | 2 | 4 | 2 | 0 | 0 | 0 | |

Pivot Column

**Table 2.2 tableau 2 of example 2.2**

| I | Basis | c | $P_0$ | -3 | -4 | -2 | 0 | 0 | 0 | θ |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | $P_1$ | $P_2$ | $P_3$ | $P_4$ | $P_5$ | $P_6$ | |
| 1 | $P_4$ | 0 | -5/3 | -1/3 | 0 | -11/3 | 1 | -2/3 | 0 | 5/11 |
| 2 | $P_5$ | -4 | 10/3 | 2/3 | 1 | 1/3 | 0 | 1/3 | 0 | |
| 3 | $P_6$ | 0 | 40/3 | -1/3 | 0 | 1 /3* | 0 | -2/3 | 1 | 40 |
| 4 | | | -40/3 | -2/3 | 0 | -2/3 | 0 | -4/3 | 0 | |

Pivot column

**Table 2.3 Optimal tableau of example 2.2**

| I | Basis | c | $P_0$ | -3 | -4 | -2 | 0 | 0 | 0 | θ |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | $P_1$ | $P_2$ | $P_3$ | $P_4$ | $P_5$ | $P_6$ | |
| 1 | $P_4$ | 0 | 145 | -4 | 0 | 0 | 1 | -8 | 11 | |
| 2 | $P_5$ | -4 | -10 | 1 | 1 | 0 | 0 | 1 | -1 | |
| 3 | $P_6$ | -2 | 40 | -1 | 0 | 3 | $P_6$ | -2 | 3 | |
| 4 | | | -40 | 0 | 0 | 0 | 0 | 0 | -2 | |

Since in the Table 2.4 there are no positive numbers on bottom row, $z_j - c_j \leq 0$, for all $j = 1,2,.. .,6$ we have optimal solution with values $x_1 = 0$, $x_2 = -10$, $x_3 = 40$, $z = -40$ and $-z = 40$ The values of the slack variables are not important , but $x_4 = 145$ , $x_5 = 0$, $x_6 = 0$,

**Section 2.5 The transportation Problem as a Linear- Programming Problem**

Now we will set up a transportation problem as a linear- programming problem Recall from Chapter1, the transportation problem is as follows·

Minimize

$$z = \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} x_{ij} \qquad (1.1)$$

Subject to the constraints

$$\sum_{j=1}^{n} x_{ij} = a_i \qquad i = 1,2, \quad , m \qquad (1\ 2)$$

$$\sum_{i=1}^{m} x_{ij} = b_j \qquad j = 1,2, \quad , n \qquad (1.3)$$

and

$$x_{ij} > 0 \qquad (1\ 4)$$

where $a_i$, $b_j$, and $c_{ij}$ are given constants

This can also be written as follows

Minimize

$$z = c_{11}x_{11} + c_{12}x_{12} + \ldots + c_{1n}x_{1n} + c_{21}x_2 + c_{22}x_{22} + \ldots + c_{2n}x_{2n} + \ldots + c_{ij}x_{ij}$$

$$+ c_{m1}x_{m1} + c_{m2}x_{m2} + \ldots\ldots + c_{mn}x_{mn},$$

subject to the following constraints

$$x_{11} + x_{12} + \ldots + x_{1n} = a_1$$

$$x_{21} + x_{22} + \ldots x_{2n} = a_2$$

$$\ldots \quad \ldots \quad \ldots \quad \ldots \quad \ldots \quad \ldots \quad \ldots$$

$$x_{i1} + x_{i2} + \ldots + x_{ij} = a_i$$

$$\ldots \quad \ldots \quad \ldots \quad \ldots \quad \ldots \quad \ldots \quad \ldots$$

$$x_{m1} + x_{m2} + \ldots x_{mn} = a_n$$

$$x_{11} + x_{21} + \ldots + x_{i1} + \ldots + x_{m1} = b_1$$

$$x_{12} + x_{22} + \ldots + x_{i2} + \ldots + x_{m2} = b_2$$

$$\ldots \quad \ldots \quad \ldots \quad \ldots \quad \ldots \quad \ldots \quad \ldots$$

$$x_{ij} + x_{2j} + \ldots + x_{ij} + \ldots + x_{mj} = b_j$$

$$\ldots \quad \ldots \quad \ldots \quad \ldots \quad \ldots \quad \ldots \quad \ldots$$

$$x_{1n} + x_{2n} + \ldots + x_{in} + \ldots + x_{mn} = b_n$$

From this we can see that the transportation problem does have an optimal solution if the constants are bounded below. However, for reasonable size values of m origins and n destinations the dimensions of resultant problem of m+n-1 equations and mn variables become rather large. Using the simplex algorithm to solve the transportation problem therefore becomes time consuming and tedious.

# CHAPTER 3

# THE INITIAL FEASIBLE SOLUTION FOR BALANCED TRANSPORTATION PROBLEM

In this chapter we are going to discus about the balanced transportation problem. We will also define what makes a balanced transportation problem, and give some theoretical support to verify that the transportation problem has a initial feasible solution. There are different methods to solve the initial feasible solution for the balance transportation problem.

Consider the transportation problem stated as follow:

Find values for the variables $x_{ij}$, which

Minimize

$$z = \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} x_{ij} \qquad (3\ 1)$$

Subject to the constraints

$$\sum_{j=1}^{n} x_{ij} = a_i \qquad i = 1,2,\ldots, m \qquad (3.2)$$

$$\sum_{i=1}^{m} x_{ij} = b_j \qquad j = 1,2, \ldots, n \qquad (3.3)$$

and

$$x_{ij} > 0 \qquad (3\ 4)$$

where $a_i$, $b_j$, and $c_{ij}$ are given constants (this will be an integer if the $a_i$ and $b_j$ were originally non negative integers)

**Definition 3.1** A balanced transportation problem is a transportation problem in which $\sum_{i=1}^{m} a_i = \sum_{j=1}^{n} b_j$

Here (mxn) decision variables and (m+n-1) side constraints.

In general the cost matrix C can be arranged

$$C = \begin{bmatrix} c_{11} & c_{12} & .. & c_{1j} & . & c_{1n} \\ c_{21} & c_{22} & . & c_{2j} & .. & c_{2n} \\ . & . & & & ... & . \\ c_{i1} & c_{i2} & . & c_{ij} & .. & c_{in} \\ . & .. & & ... & ... & .. \\ c_{m1} & c_{m2} & .. & c_{mj} & ... & c_{mn} \end{bmatrix} .. . . \quad ...(3\ 5)$$

**Theorem 3.1** Transportation problem has a feasible solution

**Proof :** By the definition of the balance transportation problem

$$\sum_{1=1}^{m} a_i = \sum_{j=1}^{n} b_j$$

and by definition of the transportation we know $a_i$ and $b_j \geq 0$

Since

$$\sum_{i=1}^{m} a_i = \sum_{j=1}^{n} b_j = D$$

and $x_{ij} = \frac{a_i b_j}{D}$ for all (i,j)

Therefore $x_{ij} \geq 0$ for all (i,j)

By the definition

$$\sum_{j=1}^{n} x_{ij} = a_i \text{ (for } i = 1,2,\ldots,m)$$

$$\sum_{i=1}^{m} x_{ij} = b_j \text{ (for } j = 1,2,\ldots,n)$$

Therefore it is satisfied since

$$\sum_{j=1}^{n} x_{ij} = \frac{\sum_{j=1}^{n} a_i b_j}{D} = \frac{a_i \sum_{j=1}^{n} b_j}{D} = a_i$$

and

$$\sum_{i=1}^{m} x_{ij} = = \frac{\sum_{j=1}^{m} a_i b_j}{D} = \frac{b_j \sum_{j=1}^{m} a_i}{D} = = b_j$$

Therefore we have proved that the transportation problem has feasible solution

**Example 3.1** One fan manufacturing company has plants in Atlanta, New York and Houston. Company has 10 fans in Atlanta, 25 in New York and 30 in Houston The company has to ship 15 to Chicago, 20 to L A, and 30 to Florida. The transportation costs per unit between cities are given in Table 3 1

$$C = \begin{bmatrix} \mathbf{4} & \mathbf{6} & \mathbf{11} \\ \mathbf{10} & \mathbf{12} & \mathbf{18} \\ \mathbf{8} & \mathbf{3} & \mathbf{2} \end{bmatrix}$$

Demand · $b_1 = 15$ ; $b_2 = 20$ ; $b_3 = 30$

Supply · $a_1 = 10$ ; $a_2 = 25$ , $a_3 = 30$

Mathematically this problem can be stated as

Minimize $$z = 4x_{11} + 6x_{12} + 11x_{13} + 10x_{21} + 12x_{22} +$$

$$18x_{23} + 8x_{31} + 3x_{32} + 2x_{33}$$

Subject to

$$x_{11} + x_{12} + x_{13} = 10$$

$$x_{21} + x_{22} + x_{23} = 25$$

$$x_{31} + x_{32} + x_{33} = 30$$

$$x_{11} + x_{21} + x_{31} = 10$$

$$x_{12} + x_{22} + x_{32} = 20$$

$$x_{13} + x_{23} + x_{33} = 30$$

$$x_{11}, x_{12}, \ldots\ldots, x_{33} \geq 0$$

This is a learner programming problem with (3x3) decision variable and (3+3-1) side constraint

**Table 3.1 Cost to Transport Fans for example 3.1**

| To / From | Chicago | L.A | Florida | Supply |
|---|---|---|---|---|
| Atlanta | 4 | 6 | 11 | 10 |
| New York | 10 | 12 | 18 | 25 |
| Houston | 8 | 3 | 2 | 30 |
| Demand | 15 | 20 | 30 | |

**Theorem 3.2**[3]

Construction of a basic feasible solution. A solution of at most m+n-1 positive $x_{ij}$'s exists

**Proof:** We shall consider the following 3x3 tableau.

| | | | |
|---|---|---|---|
| $x_{11}$ | $x_{12}$ | $x_{13}$ | $a_1$ |
| $x_{21}$ | $x_{22}$ | $x_{23}$ | $a_2$ |
| $x_{31}$ | $x_{32}$ | $x_{33}$ | $a_3$ |
| $b_1$ | $b_2$ | $b_3$ | |

If $a_1 \leq b_1$, then $x_{11} = \min(a_1, b_1)$ which we first determine for the Northwest- corner variable In this case all $x_{ij} = 0$ for $j = 2,3$. Now delete the first row and reduce the value of $b_j = (b_j - a_i)$

If $a \geq b_1$, then $x_{11} = b_1$ and all $x_{i1} = 0$ for $i = 2,3$ Now the first column is to deleted and $a_i = (a_i - b_j)$

After completing each step move rightward or downward, the rule will select as many variables for the basic set as there rows plus columns minus one, m+n-1. in the last step there will remain only one row and one column, both will be dropped at end. This will satisfy that there are at most m+n-1

$x_{ij}$'s that are positive, other $x_{ij}$ are equal to zero.

Step 1: $b_1 > a_1$

| | | | |
|---|---|---|---|
| $x_{11} = a_1$ | $x_{12} = 0$ | $x_{13} = 0$ | 0 |
| $x_{21}$ | $x_{22}$ | $x_{23}$ | $a_2$ |
| $x_{31}$ | $x_{32}$ | $x_{33}$ | $a_3$ |
| $b_1 - a_1$ | $b_2$ | $b_3$ | |

Step 2. $a_2 > b_1 - a_2$

| | | | |
|---|---|---|---|
| $x_{11} = a_1$ | 0 | 0 | 0 |
| $x_{21} = b_1 - a_1$ | $x_{22}$ | $x_{23}$ | $a_2 - b_1 + a_1$ |
| 0 | $x_{32}$ | $x_{33}$ | $a_3$ |
| 0 | $b_2$ | $b_3$ | |

Step 3. $a_2 - b_1 + a_1 < b_2$

| | | | |
|---|---|---|---|
| $x_{11} = a_1$ | 0 | 0 | 0 |
| $x_{21} = b_1 - a_1$ | $x_{22} = a_2 - b_1 + a_1$ | $x_{23}$ | 0 |
| 0 | $x_{32}$ | $x_{33}$ | $a_3$ |
| 0 | $b_2 - a_2 + b_1 - a_1$ | $b_3$ | |

Step 4. $b_2 - a_2 + b_1 - a_1 < a_3$

| | | | |
|---|---|---|---|
| $x_{11} = a_1$ | 0 | 0 | 0 |
| $x_{21} = b_1 - a_1$ | $x_{22} = a_2 - b_1 + a_1$ | $x_{23}$ | 0 |
| 0 | $x_{32} = b_2 - a_2 + b_1 - a_1$ | $x_{33}$ | $a_3 - b_2 + a_2 - b_1 + a_1$ |
| 0 | 0 | $b_3$ | |

Step 5 : $b_3 < a_3 - b_2 + a_2 - b_1 + a_1$

| | | | |
|---|---|---|---|
| $x_{11}=a_1$ | 0 | 0 | 0 |
| $x_{21}=b_1-a_1$ | $x_{22}=a_2-b_1+a_1$ | $x_{23}$ | 0 |
| 0 | $x_{32}=b_2-a_2+b_1-a_1$ | $x_{33}=b_3$ | $a_3-b_2+a_2-b_1+a_1-b_3$ |
| 0 | 0 | 0 | |

$x_{11}=a_1, x_{21}=b_1-a_1, x_{22}=a_2-b_1+a_1, x_{32}=b_2-a_2+b_1-a_1, x_{33}=b_3$

The above example shows that feasible solution can have at most m+n-1 positive $x_{ij}$'s A set of variables $x_{ij}$, constitutes a basic set of its values satisfying all m+n equations are given uniquely as linear combination of any m+n-1 of the m+n marginals $a_1, a_2, \ldots a_n, b_1, b_2, \ldots, b_n$ when the remaining values are set equal to zero. this shows the values are uniquely determined by some particular set of m+n-1 marginals.

$\sum_{i=1}^{m} a_i = \sum_{j=1}^{n} b_j$ These may be expressed as a linear combination of the others

Also $b_n = \sum^{m} a_i - \sum^{n-1} b_j, \quad x_{ij} \geq 0$

$\sum_{j=1}^{n} x_{ij} = a_i \quad$ and $\quad \sum_{i=1}^{m} x_{ij} = b_j$

Therefore this is a basic feasible solution

Now we are going to discuss different methods to solve the transportation problem to find out the initial basic feasible solution

## Section 3.1 The Northwest Corner Method

The Northwest Corner method is one of the systematic methods available to obtain the initial feasible solution. Many authors, including Cooper[1] and Kwak[7] also discussed the Northwest corner method The Northwest Corner Method is the simplest method to find out the initial basic feasible solution

The steps are as follows

1. Let $x_{11} = \min[a_1, b_1]$
2. If $a_1 < b_1$ all the other variables in the first row will be zero value

   $b_1$ will replace by $(b_1 - a_1)$, $x_{21} = \min(a_1, b_1)$
3. If $a_1 > b_1$, then first column will deleted and $a_1$ will replace by

   $(a_1 - b_1)$, $x_{12} = \min(a_1, b_2)$
4. $a_1 = b_1$ then delete the row or column, not both

5. Now move downward, rightward, or downward or rightward (as per Step 4)

6. Continue the procedure setting $x_{ij} = \min [a_i, b_j]$ and applying steps 2,3, 4 as we apply.

This process will be continued until all supplies and demands are satisfied.

Note: We need number of basic variables m+n-1. The blank cells are non-basic and the associate variables have zero values.

**Example 3.2** Solving example 3.1 by Northwest Corner method for balance problem, C is a cost matrix.

$$C = \begin{array}{cccc} \mathbf{4} & \mathbf{6} & \mathbf{11} & \mathbf{10} \\ \mathbf{10} & \mathbf{12} & \mathbf{18} & \mathbf{25} \\ \mathbf{8} & \mathbf{3} & \mathbf{2} & \mathbf{30} \\ \hline \mathbf{15} & \mathbf{20} & \mathbf{30} & \end{array}$$

← Supply

↓

Demand

**Table 3.2 Initial Transportation Tableau of example 3.1**

| To / From | Chicago | L.A | Florida | $a_i$ |
|---|---|---|---|---|
| Atlanta | 10 | 0 | 0 | 10 |
| New York | 5 | 20 | 0 | 25 |
| Houston | 0 | 0 | 30 | 30 |
| $b_j$ | 15 | 20 | 30 | 390 |

By this example we will get (m+n-1) = 5 number of basic variable.

This tableau gives us $x_{11} = 10$, $x_{21} = 5$, $x_{22} = 20$, $x_{23} = 0$, $x_{33} = 30$.

The total cost is $= 10x4 + 5x10 + 20x12 + 0x3 + 2x30 = \$390$.

The northwest corner method is very easy method and this method easily we can program , but not likely to produce a solution near to the optimal solution.

## Section 3.2 Hertley's current line method

This method is from Hartley[6].The steps are as follows·

1 The current line is defined to be row 1 Choose the smallest cost in row 1 (the current line) and assign min ($a_i$ , $b_j$) to $x_{ij}$ corresponding to the cost Adjust $a_i$ and $b_j$ exhausted.

2. If current line is not exhausted then the current line is unchanged Choose the smallest unused cost in the current line and assigned min($a_i$ , $b_j$ )

3. If current line exhausted then cross out the current line and current line will change as follows· Call the most recent cost used $c_{ij}$ If the current line was row i, then the current line change to column j. Conversely, if the current line was column j, the current line now change to row i This will continue until all supplies have been depleted, and all demands filled.

**Example3.3**

Here C is the cost matrix with supply and demand requirement.

$$C = \begin{array}{cccc} \mathbf{4} & \mathbf{6} & \mathbf{11} & \mathbf{10} \\ \mathbf{10} & \mathbf{12} & \mathbf{18} & \mathbf{25} \\ \mathbf{8} & \mathbf{3} & \mathbf{2} & \mathbf{30} \\ \hline \mathbf{15} & \mathbf{20} & \mathbf{30} & \end{array}$$

Supply

demand

Step 1 Current line is row 1. Set $x_{11} = 10$ Cross out row 1 Step 2 · Current line is column 1. Set $x_{31} = 5$. Step 3· current line change to row 3 Cross out column 1. $X_{33} = 25$ Step 4· current line change to column 3 Cross out the row 3. $X_{23} = 5$ Cross out column 3.Step 5: current line change to row 2.$x_{22}$ = 20 Cross out row 2

The solution is 4x10 + 12x20 + 18x5 + 8x5 + 2x25 = $460

**Table 3.3 Initial Transportation Tableau of example 3.1 using Hartley's Current Line Method**

| To / From | Chicago | L.A | Florida | $a_i$ |
|---|---|---|---|---|
| Atlanta | 10 | 0 | 0 | 10 |
| New York | 0 | 20 | 5 | 25 |
| Houston | 5 | 0 | 25 | 30 |
| $b_j$ | 15 | 20 | 30 | $460 |

Hartley's current line method is an algorithm that will generate a basic feasible solution, which is only somewhat likely to generate a solution that close to the optimal solution.

**Section 3.3 The Minimum Entry Method or Short Cut Method.**

This method is from Kwak[7]. This method is more effective, because the allocation according to the minimum cost criterion reduces the computation loop as well as time to determine the optimal solution. It begins with the

smallest unit transportation cost

Table 3 4 we can see the cost matrix associated with the following example

**Example 3.4**

$$C = \left[\begin{array}{ccc|c} \mathbf{4} & \mathbf{6} & \mathbf{11} & \mathbf{10} \\ \mathbf{10} & \mathbf{12} & \mathbf{18} & \mathbf{25} \\ \mathbf{8} & \mathbf{3} & \mathbf{2} & \mathbf{30} \\ \hline \mathbf{15} & \mathbf{20} & \mathbf{30} & \end{array}\right]$$

1 Begin with the minimum cost in C

2 Let the minimum cost be in the (i,j) position in C Assign to $x_{ij}$ the value min $(a_i, b_j)$

3 Adjust $a_i$ and $b_j$ to reflect the amount assigned. Cross out the row or column Where the supply or demand has gone to zero If both $a_i$ and $b_j$ have been gone to zero, cross out remaining of each row and column

4 Find the minimum cost from the remaining entries in C, following the same process from Step 2 , and continue until all supplies and demands are exhausted.

When we apply this method to Example 3.2 we get the initial feasible solution which is often the optimal solution Therefore this method is very effective to find out the initial basic feasible solution.

The initial values of $x_{ij}$ as follows . $x_{11} = 10,\ x_{12} = 0,\ x_{21} = 5,\ x_{22} = 20,\ x_{31}$ = The total cost is = 4x10 + 6x0 + 10x5 + 12x20 + 2x30= $390

**Table 3.4 Example 3.2 cost Table**

| To / From | Chicago | L.A | Florida | Supply |
|---|---|---|---|---|
| Atlanta | 4 | 6 | 11 | 10 |
| New York | 10 | 12 | 18 | 25 |
| Houston | 8 | 3 | 2 | 30 |
| Demand | 15 | 20 | 30 | |

**Table 3.5 Initial Tableau of Example 3.2 Using Short Cut Method**

| To / From | Chicago | L.A | Florida | $a_i$ |
|---|---|---|---|---|
| Atlanta | 10 | 0 | 0 | 10 |
| New York | 5 | 20 | 0 | 25 |
| Houston | 0 | 0 | 30 | 30 |
| $b_j$ | 15 | 20 | 30 | $390 |

## Section 3.4 Vogel's Advanced Start Method.

This method is named after W R Vogel This procedure is taken from Strayer[7].

1 C is the cost matrix We have to compute the difference between two smallest cost for each row and column Column difference will go up of the every column and row difference will go to the left hand side of the every row.

2 Find out the largest difference , let say it occurs in column j Take the smallest cost in column j. Then cross out the row or column where the

supply and demand has gone to zero. If there is a tie for the largest difference choose the difference that always utilization of the smallest cost.

3. If there is only one entry in the column then the difference of that column is the entry in that column; similarly one entry in row.
4. How long all supply and demand have been satisfied we should go back and continue Step 2.

This method uses cost matrix to get an advanced start in finding the optimal solution This method is quite effective to find an initial basic feasible solution, which is closer to the optimal solution.

**Example 3.5**

$$C = \begin{array}{ccc|c} \mathbf{4} & \mathbf{6} & \mathbf{11} & \mathbf{10} \\ \mathbf{10} & \mathbf{12} & \mathbf{18} & \mathbf{25} \\ \mathbf{8} & \mathbf{3} & \mathbf{2} & \mathbf{30} \\ \hline \mathbf{15} & \mathbf{20} & \mathbf{30} & \end{array}$$

**Table 3.6 Example 3.2 Using VAM**

| Difference | 4 | 3 | 9 | $a_i$ |
|---|---|---|---|---|
| 2 | 4 | 6 | 11 | 10 |
| 2 | 10 | 12 | 18 | 25 |
| 1 | 8 | 3 | 2 | 30 |
| $b_j$ | 15 | 20 | 30 | |

**Table 3.7 Initial Tableau of Example 3.2 Using VAM**

| To / From | Chicago | L.A | Florida | $a_i$ |
|---|---|---|---|---|
| Atlanta | 10 | 0 | 0 | 10 |
| New York | 5 | 20 | 0 | 25 |
| Houston | 0 | 0 | 30 | 30 |
| $b_j$ | 15 | 20 | 30 | $390 |

The largest difference is in the column 3 The smallest entry in column 3 is in the (3,3) position. So $x_{33} = 30$. Cross out the column 3 and row 3 also, because $a_3$ and $b_3$ both go to zero. Next largest difference is in column 1

The smallest entry in column 4 is in the (1,1) position. So $x_{11} = 10$ this if we do all the row and column we will get $x_{32} = 0$, $x_{21} = 5$, $x_{22} = 20$

The total cost is = 4x10 + 10x5 + 12x20 + 3x0 + 2x30 = \$390 Hillier comment that the advantage of VAM is, it is relatively easy method and it does take the unit costs into account

**Section 3.5 Russell's Approximation Method.**

Shortcut method on the minimum cost criterion is more efficient than the Northwest Corner method. Russell's approximation method [7] is almost as efficient as the shortcut method, because the basic feasible solution obtained is either optimal or close to an optimal solution.

1 Select the maximum unit cost of each row and each column.

2 From the cost matrix form a new cost matrix In the (i,j) position put the value = (max cost in row i ) + (max cost in column j) - $c_{ij}$

3 Choose the largest entry in C* Adjust $a_i$ and $b_j$ accordingly and cross out corresponding row or column, if both are going to zero at once. We can not choose both If there is a tie choose arbitrarily

4 Continue with Step 3 until all supply has been exhausted.

## Example 3.6

$$C = \begin{array}{ccc|c} \mathbf{4} & \mathbf{6} & \mathbf{11} & \mathbf{10} \\ \mathbf{10} & \mathbf{12} & \mathbf{18} & \mathbf{25} \\ \mathbf{8} & \mathbf{3} & \mathbf{2} & \mathbf{30} \\ \hline \mathbf{15} & \mathbf{20} & \mathbf{30} & \end{array}$$

$$C^* = \begin{array}{ccc|c} \mathbf{17} & \mathbf{17} & \mathbf{18} & \mathbf{10} \\ \mathbf{18} & \mathbf{18} & \mathbf{18} & \mathbf{25} \\ \mathbf{10} & \mathbf{17} & \mathbf{24} & \mathbf{30} \\ \hline \mathbf{15} & \mathbf{20} & \mathbf{30} & \end{array}$$

First choose the maximum cost for row 1 and set $x_{13}= 10$. Cross out the row

1 Choose the max value for column 1 and set $x_{21}= 15$ Cross out the column

1 Continue with this process until all supply has been exhausted

In this way we will get $x_{23}= 10$, $x_{32}= 20$, and $x_{33}= 10$

The total cost is 10x11+15x10 + 10x18 + 20x3 + 10x2 = $ 520

**Table 3.8 Example 3.2 Using RAM**

| To / From | Chicago | L.A | Florida | Supply |
|---|---|---|---|---|
| Atlanta | 0 | 0 | 10 | 10 |
| New York | 15 | 0 | 10 | 25 |
| Houston | 0 | 20 | 10 | 30 |
| Demand | 15 | 20 | 30 | $520 |

RAM is not the quick method like VAM, but it is very efficient method to find out the initial basic feasible solution. Because this solution is most of the time optimal or very close to optimal solution

It will now be appropriate to examine which of the methods described above will be most suitable for finding an initial basic feasible solution Each of the methods, we found, does in fact find a basic feasible solution i.e a solution will exactly (m+n-1) allocated variables whose corresponding cells do not form a loop. The fundamental strategy employed in all the methods is basically the same At each step, we allocate as much as possible and delete either a row or column but never both, from further consideration.

This process must lead to (m+n-1) allocations In spite of these similarities, the initial basic feasible solutions obtained by different degree of variations from the optimal solution. The number of iterations and time required by each method also differs significantly.

The Northwest Corner Method is the simplest method but it has the serious drawback of compete divorce from the considerations of the cost. This defect leads to a number of iterations and a very slow convergence to the optimal solution. The solution obtained by this method is not likely to be near to the optimal solution Hartley's Current method is less time consuming and the solution obtained is nearer to optimal solution, but nonetheless less effective and accurate than other three methods e.g. Vogel's Advanced Start, Minimum Entry and Russell's Approximation Method. Vogel's method uses the cost matrix to get an advanced start in finding a solution that is closer to the optimal solution. The iteration process in Vogel's method is considerably less than Northwest Corner Method and the solution found is closer to optimal solution, but still the process is somewhat lengthy The Minimum Entry Method is one of the most efficient methods to find the initial basic feasible solution We have seen that in this method we begin with the smallest unit of cost The allocation based on minimum cost

criterion reduces the computation and enables us to determine the optimal solution in the shortest time. Russell's Approximation method is equally effective as the minimum entry method and initial basic feasible solution obtained is either optimal or very close to an optimal solution. From the above analysis , I find the last two methods, Minimum Entry and Russell's Approximation are superior to other methods for finding the initial basic feasible solution

# CHAPTER 4

# AN OPTIMAL SOLUTION TO THE BALANCED TRANSPROTATION PROBLEM

In this chapter we will present loop and cell in the transportation problem tableau, discuss and verify the concepts of the theorems regarding the algorithm to find an optimal solution. We shall verify that every transportation problem has an optimal solution. The initial feasible solution will be evaluated to test for optimality by using stepping stone method and u-v method.

**Definition 4.1** The "box" in row i and column j of the tableau is called the (i,j) cell.

**Definition 4.2** A loop is a sequence of cells such that .

a) Each adjacent pair if cells lies in either the same row or same column of the tableau;

b) no group of three or more consecutive cells in the sequence lies in the same row or in the same column; and

c) the first and last cells in the sequence are in either the same row or the same column

The loops are closed, there is no "beginning" and no "end." Similarly every column and row of the tableau contain either no cells or and even number of cell in the loop. Thus every loop is described by a sequence of cell of the form (i,j), (p,j), (l,j), . .. , (s,r), (k,r)

**Section 4.1 The Transportation Problem has an Optimal Solution.**

The transportation problem is to minimize

$$z = \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} x_{ij} \qquad (4\ 1)$$

subject to

$$\sum_{j=1}^{n} x_{ij} = a_i \qquad i = 1,2,. \quad , m \qquad (4.2)$$

$$\sum_{i=1}^{m} x_{ij} = b_j \qquad j = 1,2, . \ \ldots, n \qquad (4.3)$$

and

$$x_{ij} > 0 \qquad (4\ 4)$$

The constraints can be written in the following form:

$$x_{11} + x_{12} + \ldots + x_{1n} = a_1$$

$$x_{21} + x_{22} + \ldots + x_{2n} = a_2$$

$$x_{m1} + \ldots + x_{mn} = a_m$$

$$x_{11} \quad x_{21} + \ldots \quad + x_{m1} = b_1$$

$$x_{12} \quad x_{22} + \ldots \quad + x_{m2} = b_2$$

$$x_{1n} + \ldots \quad x_{2n} + \quad + x_{2n} = b_m \qquad (4\,5)$$

This (3.5) notation is from (Cooper and Steinberg [1]). Thus, if we were to write the system(4 5) in matrix notation, we could do so by defining

$$y = [x_{11}, x_{12}, \ldots x_{1n}, \; x_{21}, x_{22}, \ldots x_{2n}, x_{m1}, \ldots, x_{mn}]^T,$$

$$\mathbf{b} = [\, a_1, a_2, \ldots a_m,\ b_1, b_2, \ldots b_n]^T,$$

$$A = [\, \mathbf{a}_{11}, \mathbf{a}_{12}, \ldots \mathbf{a}_{1n},\ \mathbf{a}_{21}, \mathbf{a}_{22}, \ldots \mathbf{a}_{2n}, \ldots, \mathbf{a}_{m1}, \ldots, \mathbf{a}_{mn}],$$

A is a coefficient matrix of (4.5)

where $\mathbf{a}_{ij}$ is the column of the coefficient matrix A corresponding to the variable $x_{ij}$. Then

$$A\mathbf{y} = \mathbf{b}$$

If $\mathbf{e}_k$ denotes the $k_{th}$ column of the identity matrix $I_{(m+n)}$, then it is easy to form (4.5) that

$$\mathbf{a}_{ij} = e_i + e_{m+j} \qquad (4.6)$$

**Theorem 4.1**[1] Every loop contains an even number of cells.

**proof :** Suppose we have a loop containing N cells. Choose any one of the cells as the starting point, and either of its two adjacent cells in the sequence as the second cell Number these cells 1 and 2, and continue numbering the remaining cells in the loop, in order, from 3 to N. Now , cell 1 and 2 either lie in the same row or the same column. Assume they are in the same row Then the step from any cell 2 to cell 3 must be a column change, the step 3 to cell 4 must be a row change, etc. Alternating in this manner we see that the loop from any cell t to cell (t+1) is a column change if t is even, and is a

row change if it odd. But, the step form cell N to cell 1 must be a column change, since the next change is a row change, from cell 1 to cell 2. Hence, N must be even.

**Theorem 4.2**[1] Let T denote any subset of columns $a_{ij}$ of coefficient matrix A of a transportation problem. Then the columns of T are linearly dependent if and only if the corresponding cells, or a subset of them, can be arranged into a sequence S that forms a loop

**Proof:** a) Assume a sequence of cells forms a loop Then, the sequence is of the form (k,p),(k,r),(s,r),(s,q). .....,(u,v),(u,p).

The corresponding columns of T are $a_{kp}$, $a_{kr}$, $a_{sr}$, $a_{sq}$ ..... $a_{uv}$, $a_{up}$

Suppose we alternately add and subtract these columns, forming the linear combination.

$$\mathbf{v} = a_{kp} - a_{kr} + a_{sr} - a_{sq} + \ldots\ldots\ldots a_{uv} - a_{up} \quad (4\ 7)$$

We know

$$\mathbf{a}_{ij} = \mathbf{e}_{i} + \mathbf{e}_{m+j} \quad (4{:}8)$$

Substitute (4:6) to (4:7) we get

$$\mathbf{v} = (e_k + e_{m+p}) - (e_k + e_{m+r}) + (e_s + e_{m+r}) - (e_s + e_{m+q}) + .$$

$$+ (e_u + e_{m+v}) - (e_u + e_{m+p})$$

$$= (e_k - e_k) + (e_{m+p} - e_{m+p}) + (e_{m+r} - e_{m+r}) + (e_s - e_s) +$$

$$(e_{m+v} - e_{m+v}) + \quad . \quad + (e_u - e_u) = 0$$

Since there are even number of terms in the right hand side of (4·8)

a) Nontrival linear combination of the columns $a_{kp}, a_{kr}, \ldots\ldots a_{uv}, a_{up}$

equals the null vector and the set is therefore linearly dependent

b) We must show that any linearly dependent set T of column of A has a corresponding sequence of cells which forms a loop for such a set T, there exist scalar $\alpha_{ij}$ note all zero such that

$$\sum_{a_{ij} \in T} \alpha_{ij} \mathbf{a}_{ij} = \mathbf{0} \qquad (4\ 9)$$

Hence further we shall say that a vector $a_{ij}$ appears in the sum of (4 9) if and only if corresponding scalar $\alpha_{ij} \neq 0$.

Suppose that $a_{pq}$ appears (4 9) hence the term $\alpha_{pq}\, a_{pq} = \alpha_{pq}(e_p + e_{m+j})$ appears

This implies that at least one other $a_{ij}$ corresponding to row p of the tableau

must also appear in (4.9); if this were not the case , then the pth component of the vector $\Sigma\alpha_{ij}\, a_{ij}$ would be equal to $\alpha_{ij} .1 \neq 0$; which contradicts Equation (3 9) Let $\mathbf{a}_{pk}$ be one such vector appearing in the sum of (4.9) , $k \neq q$ Then by the same reason, there must be at least one vector appearing in the sum of (4.9) corresponding to column q and one corresponding to column k, of the tableau. These conditions can be met only in one of two ways.

1 The vectors $\mathbf{a}_{rq}$ and $\mathbf{a}_{rk}$ both appear in the sum (4 9) $r \neq p$ Then a loop is formed by the cells corresponding to the four vectors $a_{pq}$, $a_{pk}$, $a_{rk}$ and $a_{rq}$ .

2 The vectors $a_{rq}$ and $\mathbf{a}_{sk}$ appear in sum of (4.9) $r \neq s$ and $r,s \neq p$ In this case the sum of (3 8) must also contain at least one additions vector corresponding to row r and one corresponding to row s These conditions can also be met only in one of two possible ways either the vectors $a_{rt}$ and $a_{st}$ are in the sum of (4.9), in which case a loop is formed by the cells corresponding to the vectors $a_{pq}$, $a_{pk}$, $a_{sk}$, $a_{st}$, $a_{rt}$, $a_{rq}$, otherwise, two vectors $a_{rt}$ and $a_{sw}$ must appear in sum of (4.9), $t \neq w$. In this case the

same reasoning leads us to the conclusion that is there an eight cell loop is formed by the appearance of at least two more vectors corresponding to columns l and v of tableau.

We can continue the above argument only a finite number of times, because there are only a finite number of vectors that can appear in the sum of (4 9) In each step of this argument we find that the required addition of two vector either forms a loop or, in turn requires the addition of at least two mote vectors Finally, we must reach a stage at which we have either formed a loop or must add the last two vector clearly when all vectors in the tableau and included, a loop is formed.

Hence we have shown that any linearly dependent set of column of A has a corresponding sequence of cells which forms a loop

We already know that in an ordinary simplex tableau, there are three sets of number to calculate

1 The element $y_{kj}$, where $y_{kj}$ is the coefficient of kth basis vector in the representation of the non-basic vector $a_j$ in terms of current vector;

2 The value of the current basis variables,

3 The values of the quantities $z_j - c_j$

Now we shall define two subscripts to denote a column

$b_k$ = kth current basis vector, k = 1,2,.. . (m+n-1)

$y_{ij}^{k}$ = coefficient of $b_k$ in the representation of the non-basic vector

$a_{ij}$ in terms of $b_1$, $b_2$,. ... .$b_{m+n-1}$,

Thus

$$a_{ij} = \sum_{k=1}^{m+n-1} y_{ij}^{k} b_k \qquad (4\ 10)$$

If the loop consist of the cells (i,j), (i,q), (p,q), (p,r), ..(u,v) then as we have seen the proof of theorem 1 we will get

$$a_{ij} - a_{iq} + a_{pq} - a_{pr} + \ldots\ldots\ldots a_{uv} - a_{uj} = 0 \qquad (4\ 11)$$

**Theorem 4.3** Given any set of basis vectors for the transportation problem, the element $y_{ij}^{k}$ of y are equal to either 0, +1 or –1, and are given by equations

$y_{ij}^{k} = \pm 1$, if $b_k$ is one of the vectors appearing in the right hand side of (4 11) ... (4.12)

0, if $b_k$ is not one of the vectors in (4 11).

and

$$y_{ij}^{k} = \begin{cases} 1, \text{ if t is even} \\ -1, \text{ if t is odd} \end{cases} \quad \ldots\ldots\ldots\ldots (4.13)$$

upon comparing equations (4.10) and (4.11).

Now we have determined an initial basic feasible solution. Also let's define $b_k$ = k the current basic vector, k = 1,2,... ...(m+n-1)

In the simplex method we first choose a non basic vector to enter the basic. Suppose we have choose vector $\mathbf{a}_{ij}$, $\mathbf{a}_{ij}$ is only non-basic vector in a loop and basic vector $b_k$ is the t th vector in the list.

**Proof:**

$$a_{ij} = \sum_{k=1}^{m+n-1} y_{ij}^{k} b_k$$

Now the m+n vector $a_{ij}$, $b_1$,$b_2$... . . $b_{m+n-1}$ are linearly dependent, hence, these vector have a corresponding sequence of cells which form a loop Further more the cell (i,j) must be in this loop, since the vectors $b_1$,$b_2$, .. $b_{m+n-1}$, are linearly independent, and their corresponding cells cannot form a loop.

Let this loop consist of the cells (i,j), (i,q), (p,q), (p,r).. ,(u,v), (u,j).Therefore $a_{ip}$, $a_{pq}$, $a_{pr}$, ..... $a_{uv}$, and $a_{uj}$ are basic vectors. Then from (4.11) we can say

$$a_{ij} = a_{iq} - a_{pq} + a_{pr} - . \quad ....-a_{uv} + a_{uj} \qquad \text{... ... .(4 14)}$$

Now , when we compare and fit (4 10) into (4 12) we get (4.14).

Furthermore ,we number the vectors corresponding to the cells in the loop in the sequence, starting with $a_{ij}$ as number 1, the basic vector $b_k$ is the rth vector in this list, than

$$y_{ij}^{k} = \begin{cases} 1, \text{ if t is even} \\ -1, \text{ if t is odd} \end{cases}$$

**The Transportation Algorithm**

0 Given . An initial balanced transportation tableau.

1 Find an initial basic feasible solution

2 Calculate the quantities $(z_{pq} - c_{pq})$ for each non-basic variable. If $z_{pq} - c_{pq} \leq 0$ for all non-basic variables the solution is optimal If $z_{pq} - c_{pq} > 0$, proceed to Step 3

3 Choose the variable $x_{ij}$ to enter the basis, according to

$$z_{ij} - c_{ij} = \text{maximum } \{ z_{pq} - c_{pq} \}$$

4 Determine the loop formed by non-basic cell (i,j) and the basic cells Number the cells in this loop consecutively, starting with (i,j), let $y_{ij}^{k} = 1$ for even-numbered cells, and $y_{ij}^{k} = -1$ for odd-numbered cells.

5. Choose the variable $x_{Br}$ to leave the basis, according to

$$x_{Br} = \min_{k} \{ x_{Bk} \ \forall_{ij} | y_{ij}^{k} = 1 \}$$

6 Compute the new basic feasible solution, by adding the quantity $x_{Br}$ to each basic variable whose cell is odd-numbered in the loop determined in Step 4, and by subtracting the quantity $x_{Br}$ from each basic variable whose cell is even numbered. Also, set $x_{ij} = x_{Br}$.

6 Return to Step 2

**Definition 4.3** A square system will be said to be triangular if upon suitable rearrangement of its rows and variables, all coefficients below the diagonal are zero and all coefficients on the diagonal are non zero.

**Theorem 4.4** [3] Fundamental Theorem for the Transportation Problem: All bases are triangular (see definition 4.3)

**Proof :** Suppose we have a standard transportation array, with m rows and n

columns and with arbitrary constants, $a_i$ and $b_j$ Consider any particular set of basic variables and substitute and value zero for each of the non-basic variables

Assume that no row or column has exactly one basic variable We discount the possibility that some row or column might have no basic variables since this would mean that the left member of the corresponding equation would be zero while the marginal total, presumed arbitrary, could be non zero. Hence all column under our assumption must have two or more basic entries If k is the total number of basic entries in the array, then since there are at least two such entries per column, we must have

$$k \geq 2n$$

Similarly we have $k \geq 2m$

Since there are at least two basic variables per row . Summing , we get

$$k \geq m+n \qquad (4\ 15)$$

Now, there are m +n equations, but since condition (1) renders one of these redundant , the number of basic variables, k , must also satisfy

$$k < m+n$$

In direct contradiction to (4.15) Thus we have proved the theorem

## Section 4.2 The Stepping Stone Method

The stepping stone method is as follows: Suppose an initial solution $x_{ij}$ has been found and it has been determined that variable $x_{ij}$ which is not currently in the solution, will decrease the value of the objective function Assigning a positive value to $x_{ij}$ introduces an imbalance into the system of constraints Given a basic feasible solution, our next task is to determine whether that solution is optimal or to select an entering variable Now

$$z_{ij} - c_{ij} = \mathbf{c}_B \mathbf{y}_{ij} - c_{ij}$$

we have shown how to obtain the components of $y_{ij}$, that is the coefficients of the representation of $\mathbf{a}_{ij}$ in terms of the basic vectors. Since $\mathbf{y}_{ij}$ consists of 1's, and -1's, and 0's, then $\mathbf{c}_B \mathbf{y}_{ij}$ is calculated by simply adding and subtracting the costs of some basic variables. The $z_{ij} - c_{ij}$ value for the non-basic variable $x_{ij}$ is given by

$$z_{ij} - c_{ij} = (c_{uj} + c_{us} + c_{is} - c_{ik} + c_{ik}) - c_{ij}$$

The optimality criterion for transportation problem is given by $z_{ij} - c_{ij} < 0$ for each non-basic $x_{ij}$. A given cell (k,l) is a candidate to enter the basis if $z_{ij} - c_{ij} > 0$

The foregoing procedure for calculating $z_{ij} - c_{ij}$ utilizes the form

$$z_{ij} - c_{ij} = \mathbf{c}_B B^{-1} \mathbf{a}_{ij} - c_{ij} = \mathbf{c}_B \mathbf{y}_{ij} - c_{ij}$$

The vector $\mathbf{y}_{ij}$ is determined by construction the unique cycle through cell (i,j) and some to the basic cells as discussed above Hence this method is sometimes called the cycle method.

The following example [8] worked out with stepping stone method will further help us to understand the method described above

**Example 4.1**

Destination

↑

| | | | |
|---|---|---|---|
| **2.50** | **2 80** | **4.50** | **150** |
| **3 50** | **2 10** | **3.50** | **110** → Factory capacity |
| **2.80** | **2 40** | **3.80** | **120** |
| **70** | **100** | **160** | |

C =

↓

Destination

requirements

Step 1 Set up the transportation tableau

Consider A,B,and C are destinations and S as slack destination. 1, 2, and 3 are the source or origin. $a_i$ and $b_j$ are the factory capacity(supply) and demand

**Table 4.1 Initial transportation tableau of example 4.1**

| To / From | A | B | C | S | $a_i$ |
|---|---|---|---|---|---|
| 1 | 2.50 | 2.80 | 4.50 | 0 | 150 |
| 2 | 3.50 | 2.10 | 3.50 | 0 | 110 |
| 3 | 2.80 | 2 40 | 3.80 | 0 | 120 |
| $b_j$ | 70 | 100 | 160 | 50 | 380 |

The slack destination S has been used to equate requirement and resources and is thus equal to 50 units. The final solution will contain 50 units of unused plant capacity, in other words, there will be total of fifty units available from the factories that will not be shipped to any of the destinations

Step 2. Develop an initial solution by Northwest Corner Method, which has been described in chapter 3 , Section 3.1. Initial solution using Northwest

corner rule is computed to be $x_{11} = 70$, $x_{12} = 80$, $x_{22} = 20$, $x_{23} = 90$, $x_{43} = 70$, and $x_{44} = 50$. Hence the total cost of initial solution is $1022

Step 3. Test the solution for possible improvement by evaluating unused squares, which is also, called stepping method.

**Table 4.2 Evaluation of unused square 2A**

| To / From | A | B | C | S | $a_i$ |
|---|---|---|---|---|---|
| 1 | 2 50 | 2 80 | 4.50 | 0 | 150 |
| | -1 ↑ 70 | → 1 80 | | | |
| 2 | 3.50 | 2 10 | 3 50 | 0 | 110 |
| | 1 ← 0 | -1 ↓ 20 | 90 | | |
| 3 | 2 80 | 2 40 | 3 80 | 0 | 120 |
| | | | 70 | 50 | |
| $b_j$ | 70 | 100 | 160 | 50 | 380 |

Adding one unit to cell 2A will increase cost by $ 3 50 Such an addition

will also cost rim requirement (constraint for destination A to be exceeded, one unit is therefore to be subtracted form cell 1A The last move will cause costs to decrease by $2 50. Now however the factory capacity are not being satisfied , so to correct the problem we must add one unit to cell 1B (increasing cost by $2.80 and subtract one unit from cell 2B ( decreasing cost by $2.10). The net effect on total cost of such a move would be

+$3.50 – $2 50 + $2.80 – $2 10 = +$1.70

Since the improvement index is a positive number for cell 2A it denotes any change in the shipping plan from factory 2 to destination A will result in increasing cost We have to continue this evaluation process for all unused squares If all the the improvement indexes found greater than or equal to zero the solution is an optimal solution. Conversely if any improvement index is negative, an improved solution, one with a smaller total cost is possible.

The procedure for evaluating each unused square is as follows .

1. Select unused square to be evaluated and trace its closed – path through the tableau
2. Beginning with a plus sign (+) in the unused square being

evaluated, assign on an alternating basis (+) and (-) sign at each corner of the closed path This positive and negative sign represent the addition and subtraction of one unit to the square

3. Determine the net effect on total cost by adding the unit cost in each square with a positive sign and subtracting the unit cost of each square with negative sign The sum is the improvement index for that unused square

We continue the evaluation of the remaining unused squares the following tableaus can be produced

**Table 4.3 Evaluation of Unused square 3A**

| To / From | A | B | C | S | $a_i$ |
|---|---|---|---|---|---|
| 1 | 2.50 | 2.80 | 4.50 | 0 | 150 |
| | -1 | 1 | | | |
| | 70 | 80 | | | |
| 2 | 3 50 | 2 10 | 3.50 | 0 | 110 |
| | | -1 | 1 | | |
| | | 20 | 90 | | |
| 3 | 2 80 | 2.40 | 3.80 | 0 | 120 |
| | 1 | | -1 | | |
| | | | 70 | 50 | |
| $b_j$ | 70 | 100 | 160 | 50 | 380 |

Improvement index · $2.80 - 2.50 + 2.80 - 2.10 + 3\ 50 - 3\ 80 = +0.70$

**Table 4.4 Evaluation of unused square 3B**

| To / From | A | B | C | S | $a_i$ |
|---|---|---|---|---|---|
| 1 | 2.50<br>70 | 2 80<br>80 | 4 50 | 0 | 150 |
| 2 | 3.50 | 2 10<br>-1 → 1<br>20 | 3.50<br>90 | 0 | 110 |
| 3 | 2 80 | 2.40<br>1 ← -1 | 3 80<br>70 | 0<br>50 | 120 |
| $b_j$ | 70 | 100 | 160 | 50 | 380 |

Improvement index : 2.40 –2.10 +3 50 – 3 80 = +0

**Table 4.5 Evaluation of unused square 1C**

| To / From | A | B | C | S | $a_i$ |
|---|---|---|---|---|---|
| 1 | 2 50<br>70 | 2 80<br>-1<br>80 | 4.50<br>1 | 0 | 150 |
| 2 | 3 50 | 2 10<br>1<br>20 | 3.50<br>-1<br>90 | 0 | 110 |
| 3 | 2 80 | 2.40 | 3.80<br>70 | 0 | 120 |
| $b_j$ | 70 | 100 | 160 | 50 | 380 |

Improvement index : 4.50 –2.80 +2 10 – 3.50 = +0 30

**Table 4.6 Evaluation of unused square 1S**

| To / From | A | B | C | S | $a_i$ |
|---|---|---|---|---|---|
| 1 | 2.50<br>70 | 2.80<br>-1<br>80 | 4 50 | 0<br>1 | 150 |
| 2 | 3 50 | 2 10<br>1<br>20 | 3.50<br>-1<br>90 | 0 | 110 |
| 3 | 2 80 | 2 40 | 3 80<br>1<br>70 | 0<br>-1 | 120 |
| $b_j$ | 70 | 100 | 160 | 50 | 380 |

Improvement Index

$$+0 - 2\ 80 + 2.10 - 3.50 + 3\ 80 - 0 = -0.40$$

**Table 4.7 Evaluation of unused 2S**

| To / From | A | B | C | S | $a_i$ |
|---|---|---|---|---|---|
| 1 | 2.50<br>70 | 2.80<br>80 | 4 50 | 0 | 150 |
| 2 | 3.50 | 2 10<br>20 | 3 50<br>-1 ← 1<br>90 | 0 | 110 |
| 3 | 2.80 | 2.40 | 3.80<br>1 → -1<br>70 | 0 | 120 |
| $b_j$ | 70 | 100 | 160 | 50 | 380 |

Improvement Index·

+0 - 3.50 + 3 80 - 0 = +0.30

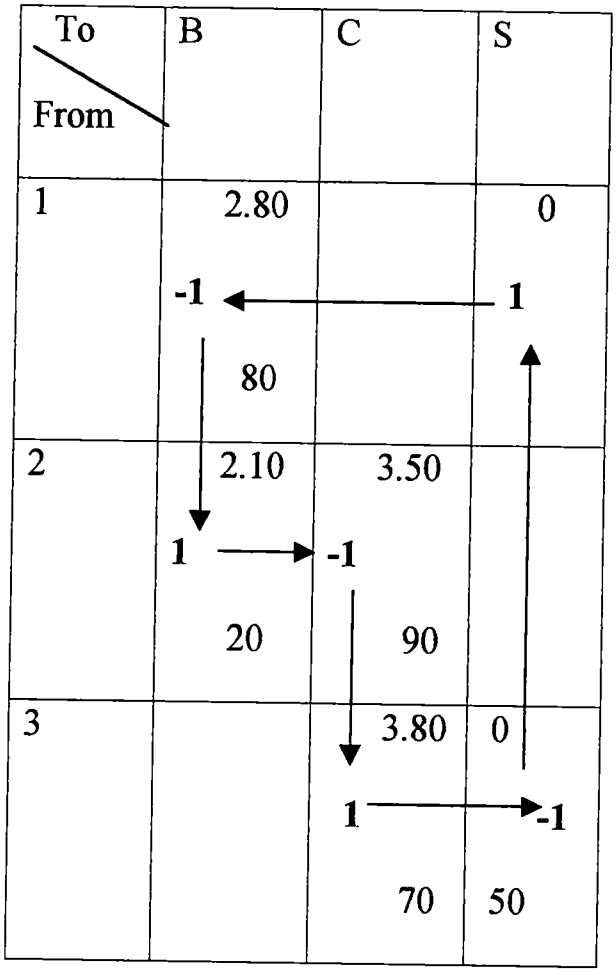
The improvement indexes for unused squares can be summarized as follows

| Unused square | Improvement index |
|---|---|
| **2A** | **1.70** |
| **3A** | **0.70** |
| **1C** | **0.30** |
| **1S** | **- 0.40** |
| **2S** | **0.30** |

Step 4 Develop a new solution.

Since the improvement index 1S is found negative we shall reconstruct the closed path for unused cell 1S and pay particular attention to unused square along this path.

**Table 4.8 Reconstruction of closed path for unused cell 1S**




The maximum quantity that can be shipped from factory 1 to destination S is given by the smallest quantity in a used square with a minus sign on the closed path for this source destination cell From the above table we find it is fifty units, as shown in cell 3S. The new solution is obtained by adding 50 units to all squares on the closed path with plus sign and subtracting 50 units

from all squares with minus sign. The following tableau represent the second solution.

**Table 4.9 Second solution**

| To / From | A | B | C | S | $a_i$ |
|---|---|---|---|---|---|
| 1 | 2.50<br>70 | 2 80<br>30 | 4.50 | 0<br>50 | 150 |
| 2 | 3.50 | 2 10<br>70 | 3 50<br>40 | 0 | 110 |
| 3 | 2 80 | 2.40 | 3 80<br>120 | 0 | 120 |
| $b_j$ | 70 | 100 | 160 | 50 | 380 |

The total cost of this solution is computed as $1002.00

Now we must repeat step 3 in the sequence to evaluate the unused cells in the second solution table 4.9 in order to determine whether any further improvement in the shipping program is possible We compute these improvement indexes.

Square 2A · 2A 1A 1B 2B 2A

+3.50 –2.50+ 2.80 – 2.10 = 1 70

Square 3A · 3A 1A 1B 2B 2C 3C 3A

+2.80 –2.50+ 2.80 – 2.10+3.50-3.80 =0.70

Square 3B : 3B 2B 2C 3C 3B

+2 40 –2 10+ 3 50 – 3.80 = 0

Square 1C 1C 1B 2B 2C 1C

+4.50 –2.80+ 2 10 – 3.50 = 0.30

Square 2S · 2S 1S 1B 2B 2S

+0 –0+ 2 80 – 2 10 = 0 70

Square 3S · 1S 1B 2B 2C 3C

+0 –0+ 2 80 – 2.10 +3.50 -3 80 = 0.40

Because all these indexes are positive, there is no better solution, the second solution of total cost $1002 00 is the least cost shipping program or the optimal solution

## Section 4.3 The UV Method

The UV method is another method frequently used for optimization, and the method is very similar to the stepping stone method described in Section 4.2 The only difference is with determining each cycle value for non-basic cells The advantage of using UV- method is that it does not require finding a loop for each non-basic variable. This method is therefore less tedious and time consuming than stepping stone method. [3,7] The method is called UV as each variable is expressed in terms of variables u and v.

Consider the following The original objective function for the transportation problem may be written as

$$z - \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij} = 0 \quad .. \ldots\ldots(4.16)$$

Suppose we eliminate the basic variables from this expression, by adding a multiple of each of the constraint equations to (4.16), let $u_i$ be the multiple of the constraint $\sum_{j=1}^{n} x_{ij} = a_i$ added to (4.16), and let $v_j$ be the multiple of the constraint $\sum_{i=1}^{m} x_{ij} = b_j$ added to (4.16) Hence, we add in total quantity

$$\sum_{i=1}^{m} u_i \sum_{j=1}^{n} x_{ij} + \sum_{j=1}^{n} v_j \sum_{i=1}^{m} x_{ij} = \sum_{i=1}^{m} u_i a_i + \sum_{j=1}^{n} v_j b_j$$

to equation(4.16) The result is

$$z + \sum_{i=1}^{m}\sum_{j=1}^{n}\{u_i + v_j - c_{ij}\}x_{ij} = \sum_{i=1}^{m} u_i a_i + \sum_{j=1}^{n} v_j b_j \;..\; .(4\ 17)$$

The coefficient of each basic variable $x_{ij}$ in equation (4.17)is zero, by our choice of values for the $u_i$ and $v_j$

Assign a constant $u_i$ for representing each corresponding row and constant $v_j$ for each column Now for each constant $c_{ij}$ corresponding to the basic variable $x_{ij}$, we let $u_i + v_j = c_{ij}$. We assign one of the u-v values to zero and can then solve for the remaining u-v values We then define each $z_{ij} = c_{ij} - (u_i + v_j)$. If any $z_{ij}$ is negative the solution is not a optimal The variable with the maximum negative $z_{ij}$ now to be introduce in the associated cycle and adjust the values of other variables of that cycle similar to stepping stone method Now get a new feasible solution and check it for optimality by repeating the steps described above Using the same problem of stepping stone method we will now find the solution by uv-method. Example is taken from Hughess-Grawoig[8].

**Example 4.2**

This is same problem used in Example 4.1 (stepping stone)

Destination

$$C = \begin{array}{ccc|c} \mathbf{2.50} & \mathbf{2.80} & \mathbf{4.50} & \mathbf{150} \\ \mathbf{3.50} & \mathbf{2.10} & \mathbf{3.50} & \mathbf{110} \\ \mathbf{2.80} & \mathbf{2\ 40} & \mathbf{3.80} & \mathbf{120} \\ \hline \mathbf{70} & \mathbf{100} & \mathbf{160} & \end{array}$$

Factory capacity

Destination

requirements

We found the initial feasible solution from Example 4 1 is

$x_{11}= 70$ , $x_{12}= 80$ , $x_{22}= 20$ , $x_{23}= 90$ , $x_{33}= 70$ , $x_{34}= 50$ . Now we calculate

$u_i$ and $v_j$ by assigning $u_1 = 0$.

$u_1 + v_1 = 2.50$ , $u_1 + v_2 = 2.80$, $u_2 + v_2 = 2\ 10$, $u_2 + v_3 = 3.50$,

$u_3 + v_3 = 3.80$, and $u_3 + v_4 = 0$

Since $u_1 = 0$ , $v_1 = 2.50$, $v_2 = 2.80$, $u_2 = -.70$, $v_3 = 4\ 80$, $u_3 = -1$, and

$v_4 = 1$.

$z_{ij}$ =

| | | | |
|---|---|---|---|
| | | **−.30** | **−1** |
| **1.70** | | | **−.30** |
| **1.30** | | | |

By introducing $x_{14}$ (the maximum negative number) we can compute new solution $x_{11} = 70$, $x_{12} = 30$, $x_{14} = 50$, $x_{22} = 70$, $x_{23} = 40$, and $x_{33} = 120$

Now we calculate the new $u_i$ and $v_j$ by assigning $u_1 = 0$

$u_1 + v_1 = 2\ 50$ , $u_1 + v_2 = 2.80$, $u_2 + v_2 = 2.10$, $u_2 + v_3 = 3.50$,

$u_3 + v_3 = 3.80$, and $u_1 + v_4 = 0$

Since $u_1 = 0$ , $v_1 = 2\ 50$, $v_2 = 2.80$, $u_2 = -.70$, $v_3 = 4\ 20$, $u_3 = -40$, and $v_4 = 0$.

$z_{ij}$ =

| | | | |
|---|---|---|---|
| | | **.30** | |
| **1 70** | | | **.70** |
| **.70** | **0** | | **.40** |

Since there is no negative value in $z_{ij}$, so the new solution is optimal solution which is $70x2\ 50 + 30x2.80 + 50x0 + 70x2\ 10 + 40x3\ 50 + 120x3.80 = 1002$, same as we found by stepping method.

# CHAPTER 5

# ASSIGNMENT PROBLEMS

There is a certain class of transportation problems, called assignment problems, for which simplex method is often very inefficient The assignment problem is a special type of transportation problem Which is widely used for different type of schedule problems There are two kinds of assignment problem one is balanced and other is unbalanced We will consider only balanced assignment problems in this section But in the real world, we occasionally find the situation in which number of available workers may be greater than the number of to be perform at a particular time vice versa For example if five men were to be assignment for four jobs, a dummy column is simply added to transform the assignment problem into square matrix The treatment of unbalanced assignment problems is analogous to the treatment of unbalanced transportation problems

The general balanced assignment problems is the general balanced transportation problem in which m = n and $s_i = d_j = 1$, $i,j = 1,2, \ldots,n$, i e ,

minimize

$$z = \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} x_{ij}$$

subject to $\sum_{j=1}^{n} x_{ij} = s_i,\ i = 1,2, \ldots\ldots n$ ( supply constraints)

(5.1)

$$\sum_{i=1}^{m} x_{ij} = d_j,\ j = 1,2,. \ldots\ldots m \text{ (demand Constraints)}$$

$$x_{ij} \geq 0, \text{ for all } i \text{ and } j$$

Here, total supply is equal to total demand

$$\text{total supply} = \sum_{i=1}^{m} s_i = \sum_{i=1}^{m} (\sum_{j=1}^{n} x_{ij}) = \sum_{i=1}^{m} \sum_{j=1}^{n} x_{ij} = \sum_{j=1}^{n} (\sum_{i=1}^{m} x_{ij}) = = \sum_{i=1}^{n} d_i = \text{total demand}$$

from the above general assignment problem, it can be proven that each $x_{ij}$ is either 0 or 1 in any optimal solution.

In balanced assignment problems, i indexes a set of n persons j indexes a set of n jobs ; $c_{ij}$ is the cost of assigning person i to job j. It is desired to assign each person to exactly one job and each job to exactly one person So total cost of assignment is minimized

$$\text{Min } z = \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij}$$

subject to

$$\sum_{j=1}^{n} x_{ij} = 1,\ i = 1,2,\ldots n \qquad (5\ 2)$$

$$\sum_{i=1}^{m} x_{ij} = 1,\ j = 1,2,\ldots\ldots .m \qquad (5.3)$$

$$x_{ij} = \begin{cases} 1 \text{ if person } i \text{ is assigned to job } j \\ 0 \text{ otherwise} \end{cases} \qquad (5\ 4)$$

Since balanced assignment problem is a special case of the balanced transportation problem, so we can use the transportation algorithm to solve balance assignment problem.

**Section 5.1 : Transportation Algorithm**

(0) Given An initial balanced transportation tableau.

(1) Apply Vogel Advanced –Start Method (VAM) to obtain a feasible solution and a corresponding basis.

(2) Let $b_1 = 0$. Determine $a_1, a_2, \ldots, a_m, b_2, b_3, \ldots, b_n$ uniquely such that $a_i + b_j = c_{ij}$ for all basis cells $c_{ij}$.

(3) Replace each cell $c_{ij}$ by $c_{ij} - a_i - b_j$ , these are the new cells $c_{ij}$

(4) If $c_{ij} \geq 0$ for all i and j , STOP, replace all cells with their original costs from (0); the basic feasible solution given by the current basis cells is optimal. Otherwise, continue.

(5) Choose $c_{ij} < 0$ ( Usually, the most appropriate choice is the most negative $c_{ij}$) . Find the unique cycle C alternately as giver cells (-) and getter cells (+) . Choose the giver cell associated with the smallest amount of supply (If there is a tie among certain giver cells for the smallest amount of supply choose any such cell )

(5) Add the squared cell of (5) to the basis, i.e., circle it in a new tableau Remove the chosen giver cell of (5) from the basis, i.e., do not circle it in a new tableau. Add the amount of goods given up by this giver cell to all amounts of supply of getter cells in C, subtract the amount of supply given up by this giver cell from all amounts of goods of giver cells in C. Go to (2)

**Example 5.1**

Solve the assignment problem below by using the transportation algorithm. The personnel assignment problem A business concern needs to fill three jobs , which demand different abilities and training. Three applicants, who can be hired for identical salaries, are available. However, because of their

different abilities, training, and experience, their value to the company depends upon the job in which each is placed The estimate of the value of each applicant to the company each year if assigned to any one of the three jobs is given below

| Jobs / Person | J1 | J2 | J3 | |
|---|---|---|---|---|
| P1 | 8 | 7 | 10 | 1 |
| P2 | 7 | 7 | 8 | 1 |
| P3 | 8 | 5 | 7 | 1 |
| | 1 | 1 | 1 | |

The parenthetical numbers below correspond to step of the transportation algorithm.

(0) The assignment problem is balanced (m=n=3)

(1)

| | 1 | 2 | 1 | |
|---|---|---|---|---|
| 1 | 8 | 7 | 10 | 1 |
| 0 | 7 | 7 | 8 | 1 |
| ~~2~~ | ~~8~~ | ~~$5^{1}$~~ | ~~7~~ | ~~1~~ |
| | 1 | 1 | 1 | |

| | 1 | 0 | 2 | |
|---|---|---|---|---|
| ~~1~~ | ~~8~~ | ~~7~~ | ~~10~~ | ~~1~~ |
| ~~0~~ | ~~7~~ | ~~7~~ | ~~8~~ | ~~1~~ |
| ~~2~~ | ~~8~~ | ~~5~~ | ~~7~~ | ~~0~~ |
| | 10 | 0 | 10 | |

Since VAM results is exactly m+n-1 = 3+3-1 = 5 circled cells, the VAM basic feasible solution is

| | | | |
|---|---|---|---|
| $8^1$ | $7^0$ | $10^0$ | 1 |
| 7 | 7 | $8^1$ | 1 |
| 8 | $5^1$ | 7 | 1 |
| 1 | 1 | 1 | |

(2)

| | 0 | -1 | 2 |
|---|---|---|---|
| 8 | $8^1$ | $7^0$ | $10^0$ |
| 6 | 7 | 7 | $8^1$ |
| 6 | 8 | $5^1$ | 7 |

(3)

| | | |
|---|---|---|
| $0^1$ | $0^0$ | $0^0$ |
| 1 | 2 | $0^1$ |
| 2 | $0^1$ | -1 |

(4) $C_{33} < 0$, contain

(5) Chosen given cell



| | | |
|---|---|---|
| $0^1$ | $+\ 0^0$ | $0^0\ -$ |
| 1 | 2 | $0^1$ |
| 2 | $-\ 0^1$ | $-1\ +$ |

(6)

| | | |
|---|---|---|
| $0^1$ | $0^0$ | 0 |
| 1 | 2 | $0^1$ |
| 2 | $0^1$ | $-1^0$ |

Go to step (2)

(2)

| | 0 | 0 | -1 |
|---|---|---|---|
| 0 | $0^1$ | $0^0$ | 0 |
| 1 | 1 | 2 | $0^1$ |
| 0 | 2 | $0^1$ | $-1^0$ |

(3)

| $0^1$ | $0^0$ | 1 |
|---|---|---|
| 0 | 1 | $0^1$ |
| 2 | $0^1$ | $0^0$ |

(4) Stop replacing all cells with their original cost from step (0) we obtain

| $8^1$ | $7^0$ | 10 |
|---|---|---|
| 7 | 7 | $8^1$ |
| 8 | $5^1$ | $7^0$ |

Therefore optimal solution is

$$x_{11} = 1, x_{12} = x_{13} = 0$$

$$x_{21} = x_{22} = 0, x_{23} = 1$$

$$x_{31} = 0, x_{32} = 1, x_{33} = 0$$

Min Cost = 8+8+5 = 21

**Theorem 5.1**

If a constant is added to any row and /or column of the cost matrix of an assignment problem (5.1), the resulting assignment problem (5.2) has the same optimal solution as the original problem

**Proof:** Let $C = [c_{ij}]$ and suppose we add $\alpha_i$ to row i, $i = 1,2,..,n$, and add $\beta_j$ to column j, $j = 1,2,..,n$ Thus the new cost matrix is $C^* = [c^*_{ij}]$, where

$$c^*_{ij} = c_{ij} + \alpha_i + \beta_j$$

If we denote by z, z* the values of the objective functions corresponding to C and C* , respectively, then

$$z^* = \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij} * x_{ij}$$

$$= \sum_{i=1}^{n}\sum_{j=1}^{n} (c_{ij}+\alpha_i+\beta_j)\, x_{ij}$$

$$= \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij} + \sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i\, x_{ij} + = \sum_{i=1}^{n}\sum_{j=1}^{n} \beta_j\, x_{ij}$$

$$= z + = \sum_{i=1}^{n} \alpha_i = (\sum_{j=1}^{n} x_{ij}) + \sum_{j=1}^{n} \beta_j (\sum_{i=1}^{n} x_{ij})$$

$$= z + \sum_{i=1}^{n} \alpha_i + \sum_{j=1}^{n} \beta_j$$

We see that z and z* differ by constant amount, independent of the values of the variables; therefore , an optimal solution to one problem must be optional for the other, and vice versa.

We can make use of this theorem in two ways first of all , if we are given an assignment problem some of whose costs are negative , we can instead deal with a related assignment problem whose casts are all nonnegative by merely adding a large enough constant to every cost. Secondly , if we choose the $\alpha_i$

and $\beta_j$ properly, we can introduce many zeros into the cost matrix, and thereby hopefully find a feasible assignment for which the corresponding $c_{ij}$ are all zero that is an optimal solution.

Essentially, this is what our proposed algorithm will do . In fact , in order to completely describe the algorithm, we need only to develop a systematic procedure for introducing the zeros into the cost matrix.

**Theorem 5.2**

Suppose the zeros of a cost matrix C in (3.5) are covered by k lines, and let m be the minimum of the uncovered elements of C. If m is subtracted form every row containing uncovered elements and added to every column covered by a line, resulting in a new cost matrix C, then the sum of the elements of C is mn(n-k)n less than the sum of the elements of C

**Proof:**First , we note that the subtraction and addition procedure described in the theorem is equivalent to the procedure of adding m to every element covered by a line (elements covered by both a row line and a column line being increased by a total of 2m) and then subtracting m from every element of C. The net effect in both procedures is to decrease uncovered elements by m, leave unchanged elements covered by one line, and increase by m elements covered by both a row and a column line

If each element in C is decreased by m, the total decrease in the sum of the elements is $mn^2$, if we then add m to each of the n elements in every line, then the total increase in the sum of the elements is nm per line, or nmk for the k line. Hence, the total net decrease in the sum of the elements is $mn^2 - nmk = mn(n-k)$

According to theorem, if we cover all the zeros by fewer than n lines ($k< n$), then the decrease in the sum of all the elements is a strictly positive quantity. Since the elements of the cost matrix are always non-negative, we can only decrease the sum of the elements until eventually the entire sum is equal to 0, which implies all the elements are zero and an procedure as soon as enough zeros are introduced to yield a complete assignment

In order to introduce a new zero, we need only be able to cover existing zeros with fewer than n lines A theorem credited to the Hungarian mathematician K Konig[3] guarantees that this is always possible.

**Theorem 5.3**

In (3.5) if k is the maximum number of zeros which can be assigned then there exists a set of k lines which will cover all the zeros

The truth of Konig's theorem is intuitively obvious, since if k is the maximum number of zeros, which can be assigned, each unassigned zero

must lie in either the same row or the same column as an assigned zero Hence, if the lines are chosen properly, there must be at least one set of k lines which covers all the zeros However, Konig's theorem does not tell us how to find such a set of lines

In order to complete the description of the algorithm we have been developing, we must describe procedures for

a) finding the maximum number of zeros which can be assigned for a given cost matrix, and

b) finding a set of k lines which covers all the zeros of the cost matrix where k is the number of zeros assigned in (a)

**Definition 5.1:**

Let T be the tableau of a balanced transportation problem in (3.1) A permutation set zeros is a subset of zero cells to T such that every row and every column of T contains exactly one zero cell of T

**Example 5. 2**

| | | |
|---|---|---|
| 1 | 0 | $0^*$ |
| $0^*$ | 1 | 0 |
| 1 | $0^*$ | 1 |

The cells distinguished by superscripted asterisks form a permutation set of zeros in T above ( in fact, this permutation set of zeros in T is unique.)

| | | |
|---|---|---|
| 1 | $0^*$ | 1 |
| $0^*$ | 1 | $0^*$ |
| 1 | 0 | 1 |

The cells distinguished by superscripted asterisks do not form a permutation set of zeros in T above (in fact, there is no permutation set of zeros in T above)

| | | |
|---|---|---|
| $0^*$ | 0 | 1 |
| 0 | $0^*$ | 0 |
| 1 | 0 | $0^*$ |

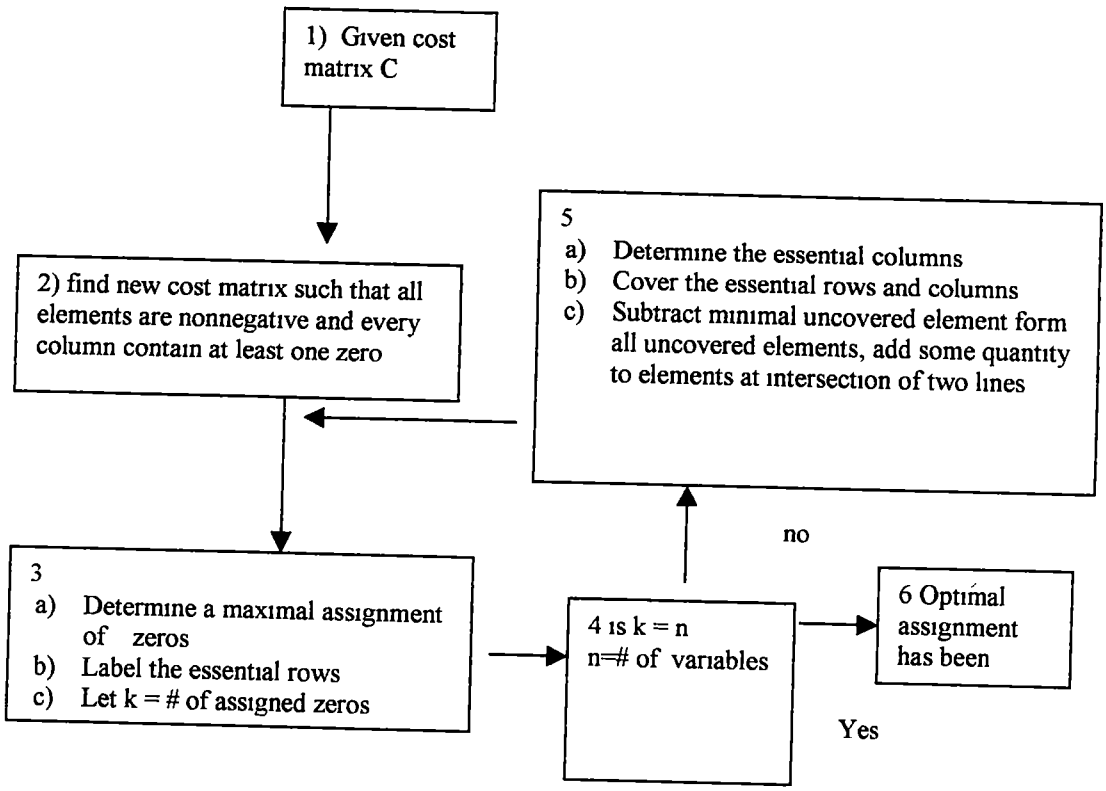
The cells distinguished by superscripted asterisks form a permutation set of zeros in T above (Find two other permutation set of zeros in T above.)

## Section 5.2 .The Assignment problem and The Hungarian Method

One of the important characteristics of the Hungarian method[3] is that the number of rows equals numbers of columns, the assignment matrix C*must be a square matrix. But in the real problem we occasionally find the situation in which this condition does not hold The number of available workers may be greater than the number of jobs to be performed at a particular time vice versa. For example if five men were to be assigned for four jobs, a dummy column is simply added to transform the assignment problem into 5x5 matrix. A dummy column means that one man is idle at a particular time. A dummy row means that one job is differed the requirement is not met

The Hungarian method · Definition 4.1 With the detailed procedures developed in the processing on the assigning zero and determining a set of covering times, we will completely describe an algorithm for solving assignment problems. This algorithm called the Hungarian method It is conveniently summarized by the flow diagram

## Hungarian method flow diagram



The Hungarian Method is a finite algorithm, in other words, it has finite number of steps. Since the "k" of Step 3 is non-decreasing, and the sum of the elements of the cost matrix generated in Step 5 is strictly decreasing at each iteration, and is bounded from below by zero. Moreover, it is obvious that the algorithm can be implemented using integer arithmetic exclusively

The Hungarian method led to the later development of the primal-dual algorithm, the later is in fact a generalization of the Hungarian Method. Box 3 of the above figure, corresponds to the solution of restricted primal, in which only certain variables are allowed to become positive ( in assignment problem these are the variables whose $c_{ij} = 0$ ) Box 5 of above figure corresponds to finding a new dual solution in such a way that at least one variable previously restricted to a value of zero is now allowed to become positive.

Computationally, the slowest part of Hungarian Method involves the generation of all possible transfers.

There are several ways in which this procedure can be modified, along

with the determination of the zero covering lines Such modifications are described by Kuhn. In one of these Kuhn suggests a procedure for finding n-1 lines to cover all the zeros. Kuhn's modification still leads to a finite algorithm, and although it reduces the computation involving the assignment of zero at each iteration, the total number of iterations may increase in the sum of elements will probably be less per iteration than in the original method. Another modification of the Hungarian Method, which should tend to make the assignment of zeros procedure more efficient is to modify Box 3 of above figure.

1 Search each row for an unmarked 0 If a row contains only one unmarked 0,assign that 0, and marked it $0^*$. Marked each zero in the column containing

this $0^*$ with an $m(0^*)$ Such 0 cannot be assigned.

If the row contains no unmarked zero or more than one, proceed next row Continue until all rows have been searched.

a All zero have been marked. In this case a maximal assignment has been found. All rows containing a $0^*$ are essential rows

b Every row and every containing unmarked zeros has at least two such unmarked zeros. All rows containing a $0^*$ should be labeled

essential To assign the remaining zeros delete the rows and columns, which contain an assigned 0. Then use the original procedure on this reduced matrix. In the above procedure, it may be necessary repeat steps 1 and 2 several times. In many cases will lead to step3 a and b.

Step1.-Assign in each row the first 0, if any, not in the column of previously assigned zero

Step2.-Search each column for a $0^*$ If a $0^*$ is found proceed to next column, if a $0^*$ is not found in a column, this column is called eligible and is searched for an unassigned zero. If 0 is not found, proceed to the next column If a 0 is found, proceed next step

Step3.-a 0 has been found in cell $(i_1,j_0)$ Find a sequence of cells of the following form

$0$ in $(i_1,J_0)$

$0^*$ in $(i_1,J_1)$

$0$ in $(i_2,J_1)$

$0^*$ in $(i_2,J_2)$

**Example 5.3**

| Territories / Salesman | I | II | III | IV |
|---|---|---|---|---|
| A | $26 | $27 | $20 | $30 |
| B | 26 | 23 | 20 | 31 |
| C | 24 | 22 | 34 | 26 |
| D | 19 | 21 | 21 | 22 |
| E | 30 | 25 | 28 | 27 |

In this problem there are five sales men with four territories to be covered It is desired to assign each sales man to exactly one territory, so that the total cost of assignment is minimized.

To apply the Hungarrian method to this problem we have to add a dummy column with zero cost, so that the problem is expressed as a square matrix

| Territories / Salesman | I | II | III | IV | Dummy |
|---|---|---|---|---|---|
| A | $26 | $27 | $20 | $30 | $0 |
| B | 26 | 23 | 20 | 31 | 0 |
| C | 24 | 22 | 34 | 26 | 0 |
| D | 19 | 21 | 21 | 22 | 0 |
| E | 30 | 25 | 28 | 27 | 0 |

In the above matrix the smallest element in each row is zero Therefore the opportunity cost matrix is exactly the same as the revised cost matrix.

The total opportunity cost is below

| Territories / Salesman | I | II | III | IV | Dummy |
|---|---|---|---|---|---|
| A | 5 | 6 | 0 | 0 | $0 |
| B | 7 | 2 | 2 | 11 | 0 |
| C | 5 | 1 | 16 | 6 | 0 |
| D | 0 | 0 | 3 | 2 | 0 |
| E | 11 | 4 | 10 | 7 | 0 |

Total opportunity-cost Matrix

The total opportunity cost matrix is obtained by subtracting the smallest element in each column from each element in that column. When the iterative procedures are followed step by step, we finally get an optimal total opportunity cost matrix

| Territories / Salesman | I | II | III | IV | Dummy |
|---|---|---|---|---|---|
| A | 5 | 6 | 0 | 0 | 1 |
| B | 6 | 1 | 1 | 11 | 0 |
| C | 4 | 0 | 15 | 5 | 0 |
| D | 0 | 0 | 3 | 2 | 1 |
| E | 10 | 3 | 9 | 6 | 0 |

Revised total opportunity cost Matrix

| Territories / Salesman | I | II | III | IV | Dummy |
|---|---|---|---|---|---|
| A | 6 | 7 | 0 | 0 | 2 |
| B | 6 | 1 | 0 | 9 | 0 |
| C | 4 | 0 | 14 | 4 | 0 |
| D | 0 | 0 | 2 | 1 | 1 |
| E | 14 | 3 | 8 | 5 | 0 |

An Optimal total opportunity cost Matrix

An optimal assignment schedule and the total cost are given below

| Assignment | Cost |
|---|---|
| A to IV | $ 20 |
| B to III | $ 20 |
| C to II | $ 22 |
| D to I | $ 19 |
| E to Dummy | $ 0 |
| Total | $ 81 |

Salesman E is idle when an optimal assignment is made

**Section 5.3 Conclusion**

In conclusion I shall try to summarize my observations We have seen that transportation problems are a particular type of linear programming problem. A basic feasible solution exists for transportation problems Although Simplex method is the backbone for solving linear programming problems, it is not the most efficient method for solving transportation problems because of tedious and lengthy procedure Simplex method essentially starts form one extreme point, and follows a path along the edges

of the feasible region towards an optimal extreme point, such that all the intermediate extreme points visited are improving the objective function For a transportation problem with a number of variables and constraints, the number of reiterations required in Simplex method thus makes this algorithm lengthy and tedious.

A number of special algorithms have been later developed for solving transportation problems in a more efficient way. All these methods lead to find an initial basic solution that eventually leads to find an optimal solution. On examining the different methods of finding the initial basic feasible solution, it is observed that Minimum Entry and Russel's approximation methods are superior than the other methods discussed in this paper These two methods obtain initial basic solution that is either optimal or closest to optimal. Assignment problems are a special type of transportation problem that are widely used in real life situations in engineering, production, and marketing. Several methods are used to solve assignment problems. Although transportation method can be used to solve assignment problems, the Hungarian method and related variations appear to be superior. With the advent of computer technology, industrial applications involving large number of variants and constraints like airlines schedule, machine

assignments, production bottleneck are being solved by using specially developed software These software primarily utilize revised simplex method. Use of computer software becomes very effective for those who must solve transportation problems repetitively using input from same but updated extensively database A brief discussion about commercially available software for solving linear programming problems has been included in Appendix. Besides assignment problems, there are many other special types of transportation problems Some of these problems are transshipment problem, restricted routes, and network problems.

A number of methods other than the ones mentioned in this paper exist for solving transportation and assignment problems. It is an open question to state, which of all the available methods is superior to solve these problems It will largely depend on the formulation and structure of the problem to decide the most efficient algorithm to solve the problem in the quickest possible way. As we have seen that special algorithms have been developed for solving special type of linear programming problems e.g. transportation, assignment etc., continual research is required to develop new algorithms for other types of specific linear programming problems

# REFERENCES

## REFERENCES


1. Cooper and Steinberg, *Methods and Applications of Linear Programming.* Philadelphia: W B. Saunders Company, 1974

2 Dantzig and Thapa, *Linear Programming 1: Introduction.* New York : Springer- Verlag, 1997

3 Gass, Saul I , *Linear Programming: Methods and Applications.* Fifth Edition New York: McGraw-Hill Book Company, 1986

4. Gribik, Paul R. and Kenneth O. Kortanek. *External methods of Operations Research.* New York: Marcel Dekker, Inc., 1985

5 Hadley, G , *Linear Programming.* Second Printing. Reading, Massachusetts: Addison-Wesley Publishing Company, 1963

6 Hartley, Roger, *Liner and Nonlinear Programming: An Introduction to Linear Methods in Mathematical Programming.* New York. Ellis Horwood Limited, 1985.

7. Kwak, N.K., *Mathematical Programming with Business Applications.* New York: McGraw-Hill Book Company, 1973

8 Hughes-Grawoig, *Linear Programming: An Emphasis on Decision Making.* Reading, Massachuests, Addison-Wesly, 1973.

9. Stayer, James K., *Linear Programminad and Its Applications.* New York: Springer-Verlag, 1989

10 Vanderbei, Robert J., *Linear Programming Foundations and Extensions* Boston· Kluwer Academic Publisher, 1996

11 Winston, Wayne L., *Operations Research: Application and Algorithms.* Belmont, California Duxbury Press, 1994.

12 Yu, Gang, *Operations Research in the Airline Industry.* Boston Kluwer Academic Publisher, 1998.

13 Goldberg, Jack L., *Matrix Theory with Applications.* McGraw-Hill, New York, 1991.

14 Anderson-Sweeney-Williams, *Linear Programming for Decision Making,* West Publishing Company, St. Paul, 1976

15 OR/MS Today- August 1999

# APPENDIX

# APPENDIX

## Applications of Computer to solve linear programming problems.

In previous chapters we discussed the different methods of solving linear programming problems in general, and transportation problems in particular All the algorithms utilized in different methods are based on manual calculations that are very often found tedious and time consuming for the real life problems encountered in industries. Many transportation and assignment problems have to deal with such a huge number of constraints and variables, it becomes impractical to solely depend on manual calculations. Further as calculations increase for large problems, so is the chance of human error. A number of software packages are available in the market today to solve linear programming problems A brief survey [15] of the different software packages commonly used in different operating interface environments is included at the end of this chapter

From a mathematical viewpoint the LiNDO program offers some definite advantages over the other software packages, and we shall take a closer look at this program. The name UNDO is derived from Linear

Interactive, Discrete, and Optimizer. It is an interactive linear, quadratic, and integer programming package that is fast and extremely user friendly for a wide range of applications. Although LINDO is commonly used in solving large problems with huge number of variables and constraints, the program can also be used for very simple problems without a need for a big setup cost for first time users. It is a command oriented program, not menu oriented It does not lead the user through a fixed sequence of steps Rather, it offers a wide range of options in the form of different commands, any of which can be executed at any time LIINDO checks whether a particular command is applicable in a particular context. The program can be run with three different user interfaces

1) Mainframe version,
2) MS- DOS version,
3) Apple Macintosh version

Although biased towards developer of models of real life problems and solutions, the program in all three versions allows the mathematician to display the tableau and steps of calculations An example problem worked out with LINDO is attached to show its different features and efficiency.

The following example, which has been solved in example 5 1 is now solved by using LINDO software

**Example:**

A business concern needs to fill three jobs, which demand different abilities and training. Three applicants, who can be hired for identical salaries are available However, because of their different abilities, training, and experience, their value to the company depends upon the job in which each is placed. The estimate of the value of each applicant to the company each year if assigned to any one of the three jobs is given below.

| Jobs / Person | J1 | J2 | J3 | J4 |
|---|---|---|---|---|
| P1 | 8 | 7 | 10 | 1 |
| P2 | 7 | 7 | 8 | 1 |
| P3 | 8 | 5 | 7 | 1 |
| | 1 | 1 | 1 | |

Input:

```
min 8a + 7b + 10c + 7d + 7e + 8f + 8g + 5h + 7i
st
a + b + c = 1
d + e + f = 1
g + h + i = 1
a + d + g = 1
b + e + h = 1
c + f + i = 1
end
integer a
integer b
integer c
integer d
integer e
integer f
integer g
integer h
integer i
```

```
LP OPTIMUM FOUND AT STEP     6
OBJECTIVE VALUE =  21 0000000

FIX ALL VARS (   4) WITH RC >  0 000000E+00

NEW INTEGER SOLUTION OF   21 0000000    AT BRANCH     0 PIVOT     6
BOUND ON OPTIMUM  21 00000
ENUMERATION COMPLETE. BRANCHES=    0 PIVOTS=     6

LAST INTEGER SOLUTION IS THE BEST FOUND
RE-INSTALLING BEST SOLUTION

     OBJECTIVE FUNCTION VALUE

     1)    21 00000

 VARIABLE       VALUE          REDUCED COST
     A        1 000000         8 000000
     B        0 000000         7.000000
     C        0 000000        10 000000
     D        0 000000         7.000000
     E        0 000000         7 000000
     F        1 000000         8 000000
     G        0 000000         8 000000
     H        1 000000         5 000000
     I        0 000000         7 000000


    ROW   SLACK OR SURPLUS     DUAL PRICES
     2)        0 000000         0 000000
     3)        0 000000         0 000000
     4)        0 000000         0 000000
     5)        0 000000         0 000000
     6)        0 000000         0 000000
     7)        0 000000         0 000000

NO. ITERATIONS=     6
BRANCHES=   0 DETERM.= 1 000E   0
```

# 1999 LINEAR PROGRAMMING SOFTWARE SURVEY

| Software Package | Vendor | Comments/Description |
|---|---|---|
| Aimms 2.20, Aimms 3.0 | Paragon Decision Technology | Aimms 3.0 is a complete modeling development environment for constructing and maintaining DSS applications in which linear programs can be embedded |
| | | |
| AMPL | Various LP software vendors | The comprehensive AMPL modeling language supports a broad variety of solvers, including the latest non-linear programming technologies For a list of vendors, visit www.ampl.com/ampl/vendors.html. |
| AMPL/CPLEX & AMPL PLUS/CPLEX | ILOG | AMPL is a recognized standard for mathematic programming modeling and analysis. AMPL Plus includes a Windows GUI and database connectivity. |
| Aspen MIMI (Manager for Interactive Modeling Interfaces) | Chesapeake Supply Chain Div. (Aspen Technology) | |
| Aspen PIMS | Aspen Technology Inc. | — |
| C-WHIZ | Ketron Management Science | Can be executed stand-alone from command line or via front-end GUI; its is directly callable from DATAFORM; and can be integrated into popular language applications as OML |
| DATAFORM | Ketron Management Science | DATAFORM is a data-driven programming language capable of doing all model data manipulation in addition to comprehensive model management and solution control. |
| DecisionPro 3.0 | Vanguard Software Corp. | DecisionPro is an integrated tool for decision-support analysis and business modeling. |
| Fort MP Optimisation System | UNICOM Consultants Ltd | FortMP is progressively refined in its performance and includes SSX, IPM, Efficient Crossover, MIP (Preprocess, Constraint Classification, Branch & Cut), QP, MIP-QP, SP Algorithms Readily connected to MPL and AMPL. |
| GAMS | GAMS Development Corp. | GAMS is an algebraic modeling system providing powerful modeling capabilities and links to SOTA solvers |
| HS/LP | Haverly Systems Inc | HS/LP is a high-performance linear programming system capable of solving large-scale LP problems Input data is accepted in several formats including the industry standard MPS format |
| ILOG CPLEX Callable Library | ILOG | Recognized world standard and leader for solving large or challengin LPs, MIPs, QPs, and network problems. Fastest and most robust solvers available anywhere. |

| | | |
|---|---|---|
| ILOG OPL Studio | ILOG | OPL Studio is the only modeling system to fully support both math programming and constraint programming. Code generation feature simplifies application development and deployment. |
| ILOG Planner | ILOG | Powerful C++ class library for solving LPs and MIPs with CPLEX engine. Flexible, object-oriented and easy-to-use C++ modeling environment simplifies application development. |
| LAMPS | Advanced Mathematical Software Ltd. | — |
| LINDO | LINDO Systems | LINDO is a fast, reliable and easy-to-use solver for linear and integer optimization models. It can be run interactively or embedded into your own application. |
| LINGO | LINDO Systems | LINGO combines fast linear, integer and non-linear solvers with a modeling language allowing easy problem expression. LINGO offers convenient data storage in Excel and popular databases. |
| Linear Programming/GAUSS | Aptech Systems Inc. | Source code included for complete customizability. |
| LOQO | Princeton University | — |
| LP-Toolkit | EURO-DECISION | LP-Toolkit is a C++ class library offering a modeling interface with MIP solvers; it allows direct inclusion of algebraic-like modeling instructions in a C++/C/Visual Basic. |
| LSSOL 1 05 | Stanford Business Software Inc. | LSSOL can accommodate problems with hundreds of constraints/variables. LSSOL was developed at Stanford and UC San Diego. |
| Maple V Release 5.1 | Waterloo Maple Inc. | Maple V 5.1 is a powerful interactive computer algebra system that provides a complete mathematical environment for the manipulation of symbolic algebraic expressions, arbitrary-precision numerics, graphics and programming. |
| MathPro 2000 | MathPro Inc. | An integrated, interactive modeling system with embedded solver, GUI, case management, symbolic modeling language, matrix generation, solver control, solution query and display, and reporting. |
| MGG | EDS | Generates efficient, industrial-strength code for use in installed black-box solutions |
| MINOS 5.5 | Stanford Business Software Inc. | MINOS can accommodate problems with many thousands of constraints/variables. Nonlinear objective and constraints. Optional sensitivity analysis/integer variables MINOS was developed by researchers at |
| MPL for Windows 5 0 | Maximal Software | — |
| O-Matrix | Harmonic Software Inc. | — |

| | | |
|---|---|---|
| OML | Ketron Management Science | Library includes LP and MIP optimization, model management, matrix generation and modification, solution access, table processing. C-WHIZ is the imbedded optimizer. |
| OMNI | Haverly Systems Inc. | OMNI is an LP model management system. It is ideally suited for the optimal purchase of feed stocks, pinpointing bottlenecks and optimizing unit operations. |
| OMP Optimization | OM Partners | — |
| OPTIMAX 2000 | Maximal Software | OptiMax 2000 will revolutionize how optimization modeling is applied to solve business problems Free download of the student/trial version will be available from our Web site: www.maximal-usa.com. |
| Optimization Solutions | IBM | — |
| Premium Solver | Frontline Systems Inc. | Upgrade to Excel's standard solver, with greater capacity and speed, and many new features. Fully compatible with Excel Solver's models, Visual Basic for applications programs. |
| SCICONIC | EDS | Uniquely powerful subroutine library interface permitting development of sophisticated and novel techniques. |
| SNOPT 5.3-4 | Stanford Business Software Inc. | SNOPT can accommodate problems with many thousands of constraints/variables Nonlinear objective and constraints (SQP Method). SNOPT was developed at Stanford and UC San Diego. |
| Solver DLL | Frontline Systems Inc. | Reliable, enhanced versions of the Excel solver optimization engines packaged in an easy-to-use Dynamic Link Library (DLL) callable from any Windows programming language. |
| SOPT | SAITECH Inc. | SOPT is good at solving large-scale, real-world problems. It is very efficient for large IP problems, such as set partitioning and covering, TSP, and knapsack problems |
| Visual XPRESS | Dash Associates | Integrated development and debugging in a GUI environment for fast development of models. Easy to learn and to use. |
| What's Best! | LINDO Systems | What's Best is a fast, reliable spreadsheet solver capable of handling models of well over 100,000 variables It's ideal for building easy-to-use models for managers and clients. |
| XA | Sunset Software Technology | The XA Callable Library is ideal for embedding optimization algorithms into your products. |
| XPRESS-MP | Dash Associates | A top-performing industrial-strength LP/MIP/QP optimizer. A combined modeling/optimizer library, EMOSL, speeds algorithm development |

# A Product Mix Example

This page illustrates the process of creating a Solver model, using a well-worn example based on the Product Mix sheet in the SOLVSAMP XLS workbook included with Microsoft Excel, and in the SOLVSAMP 123 workbook included with Lotus 1-2-3 97 (The version in the SOLVSAMP workbook is somewhat more complex and requires the nonlinear GRG Solver ) We'll take you through the steps of defining your problem, writing out formulas for the objective function and constraints, creating a worksheet, defining the problem elements through the Solver dialogs, running the Solver to find the optimal solution, and producing an Answer Report

## Setting Up a Model

To set up an optimization model as a spreadsheet, you will follow these essential steps

1. Reserve a cell to hold the value of each decision variable
2. Pick a cell to represent the objective function, and enter a formula that calculates the objective function value in this cell
3. Pick other cells and use them to enter the formulas that calculate the left hand sides of the constraints
4. The constraint right hand sides can be entered as numbers in other cells, or entered directly in the Solver's Add Constraint dialog box

Within this overall structure, you have a great deal of flexibility in how you lay out and format the cells which represent variables and constraints, and which formulas and built-in functions you use For example, the formulas needed for a linear programming problem can always be specified with the SUMPRODUCT function, or with the included with each of the products If the model is easily expressed in vector-matrix form, you may want to use defined names for the vectors and built-in functions such as MMULT (in Microsoft Excel) to compute the constraints

## A Sample Model

Imagine that you are managing a factory which is building three products TV sets, stereos and speakers Each product is assembled from parts in inventory, and there are five types of parts Chassis, picture tubes, speaker cones, power supplies and electronics units Your goal is to produce the mix of products which will maximize profits, given the inventory of products on hand

To assemble a TV set, you need 1 chassis, 1 picture tube, 2 speaker cones, 1 power supply and 2 sets of electronics To make a stereo, you need 1 chassis, 2 speaker cones, 1

power supply and 1 set of electronics And to build a speaker, all you need is 1 speaker cone and 1 set of electronics

You can sell TV sets for a gross profit of $75 each, stereos for a profit of $50 each, and speaker
cones for $35 each The inventory you have on hand consists of 450 chassis, 250 picture tubes, 800 speaker cones, 450 power supplies and 600 sets of electronics

## Writing the Formulas

Before looking at a spreadsheet, let's write out formulas corresponding to the verbal description above If we temporarily use the symbol *x1* for the number of TV sets assembled, *x2* for the number of stereos, and *x3* for the number of speakers, our total profit is

Maximize 75 *x1* + 50 *x2* + 35 *x3* (Profit)

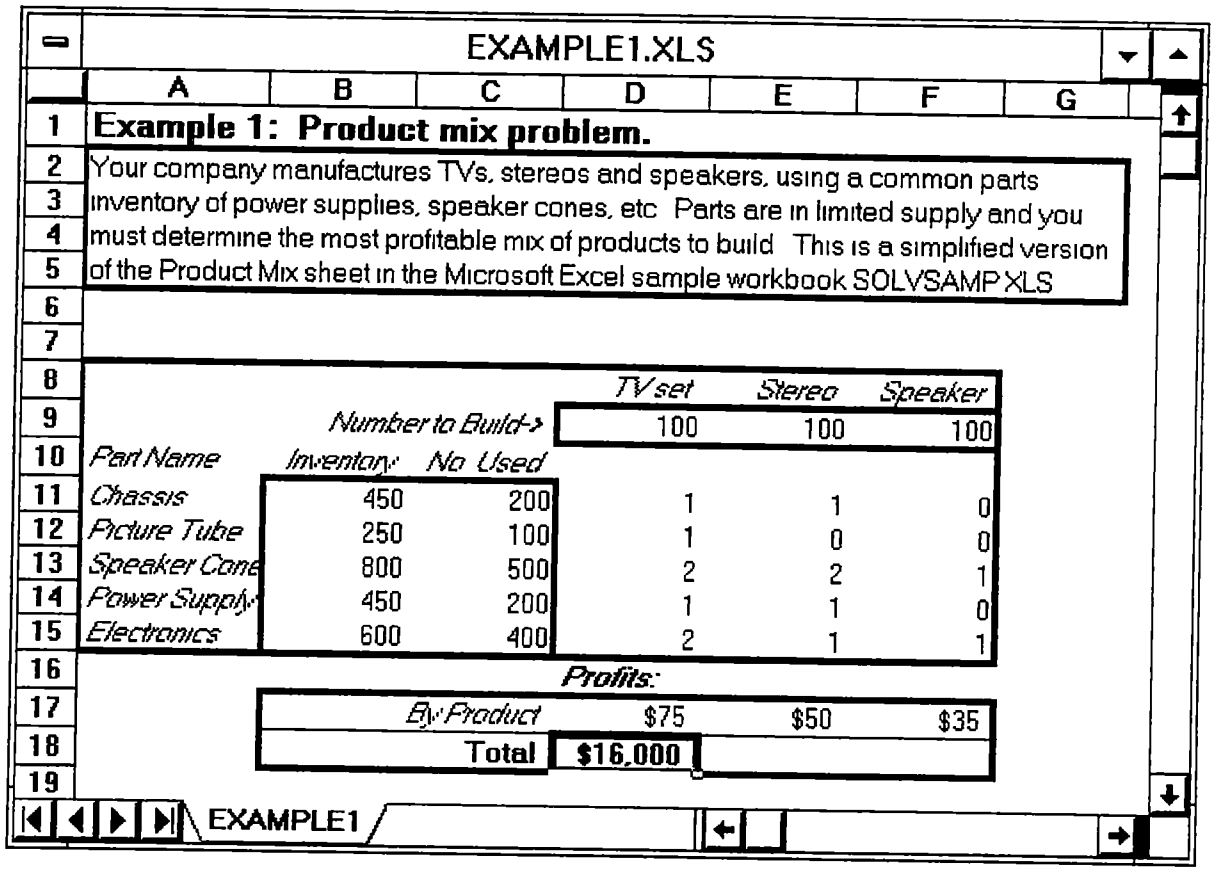
Building each product requires a certain number of parts of each type For example, TV sets and stereos each require one chassis, but speakers don't use one The number of parts used depends on the mix of products built (the left hand side of each constraint), and we have a limited number of parts of each type on hand (the corresponding constraint right hand side)

Subject to
1 *x1* + 1 *x2* + 0 *x3* <= 450 (Chassis)
1 *x1* + 0 *x2* + 0 *x3* <= 250 (Picture tubes)
2 *x1* + 2 *x2* + 1 *x3* <= 800 (Speaker cones)
1 *x1* + 1 *x2* + 0 *x3* <= 450 (Power supplies)
2 *x1* + 1 *x2* + 1 *x3* <= 600 (Electronics)

Notice that we've expressed each constraint in a uniform way Where we aren't using any parts of a particular type, we just use a quantity of 0 parts This will make it easier to enter and copy the constraint formulas in the spreadsheet We can write the actual spreadsheet formulas for the objective and the constraint left hand sides using the SUMPRODUCT function This is very readable, and it's also in the form required to take advantage of *fast problem setup* in Frontline's

Since the number of products built cannot be negative, we'll also have constraints *x1*, *x2*, *x3* >= 0

## The Worksheet (Shown in Excel)




The next step is to create a worksheet where the formulas for the objective function and the constraints are calculated In the above worksheet, we have reserved cells D9, E9 and F9 to hold our decision variables *x1*, *x2* and *x3* The number of TV sets, stereos and speakers to build Notice that the gross profit for each assembled product ($75, $50 and $35) is shown in cells D17, E17 and F17 This allows us to compute the objective function in cell D18 as

SUMPRODUCT(D17 F17,D9 F9)

In the table area D11 F15, we've entered the number of parts of each type needed to assemble each finished product For example, the 2 at D13 means that we need 2 speaker cones to build a TV set These numbers come directly from the formulas for the constraints shown earlier With these values in place, we can enter a single formula at cell C11

SUMPRODUCT(D11 F11,$D$9 $F$9)

to compute the total number of chassis we will use for any given number of TV sets, stereos and speakers Then we can copy this formula to cells C12 C15 to compute the total number of parts used of all the other types (The dollar signs in $D$9 $F$9 specify that this cell range stays constant, while the cell range D11 F11 becomes D12 F12, D13 F13, etc in the copied formulas )

In cells B11 F15, we've entered the number of parts we have in inventory of each type This allows us to express the constraints shown earlier as

C11 C15 <= B11 B15

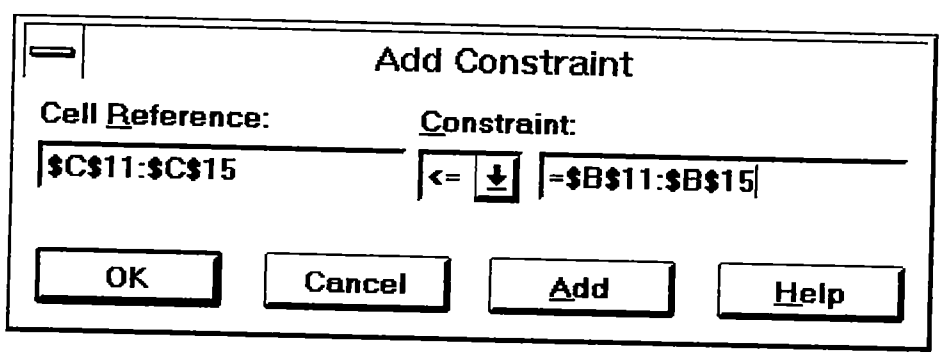
This is a shorthand for C11 <= B11, C12 <= B12, and so on through C15 <= B15 We can enter this form directly in the Solver dialogs We will also enter the constraint

D9 F9 >= 0

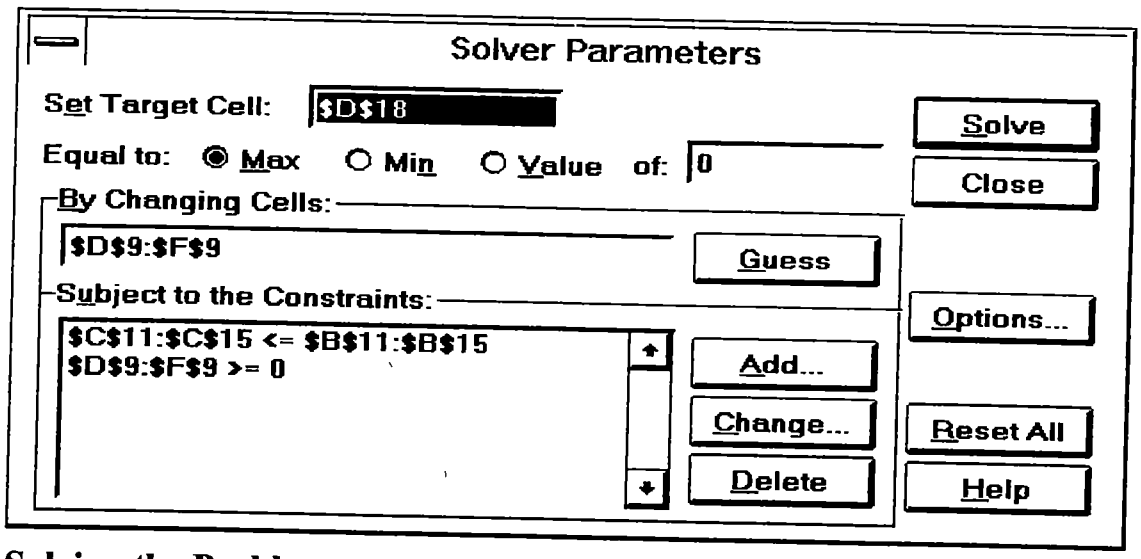
to require the decision variables to have non-negative values

## Using the Solver Dialogs

To let the Solver know which cells on the worksheet represent the decision variables, constraints and objective function, we select **Tools Solver...** in Excel, or **Range Analyze Solver...** in 1-2-3, which displays the Solver Parameters dialog In the Set Target Cell edit box, we type or click on cell D18, the objective function. In the By Changing Cells edit box, we type D9 F9 or select these cells with the mouse To add the constraints, we click on the Add button, select cells C11 C15 in the Cell Reference edit box (the *left hand side*), and select cells C11 B15 in the Constraint edit box (the *right hand side*), the default relation <= is OK

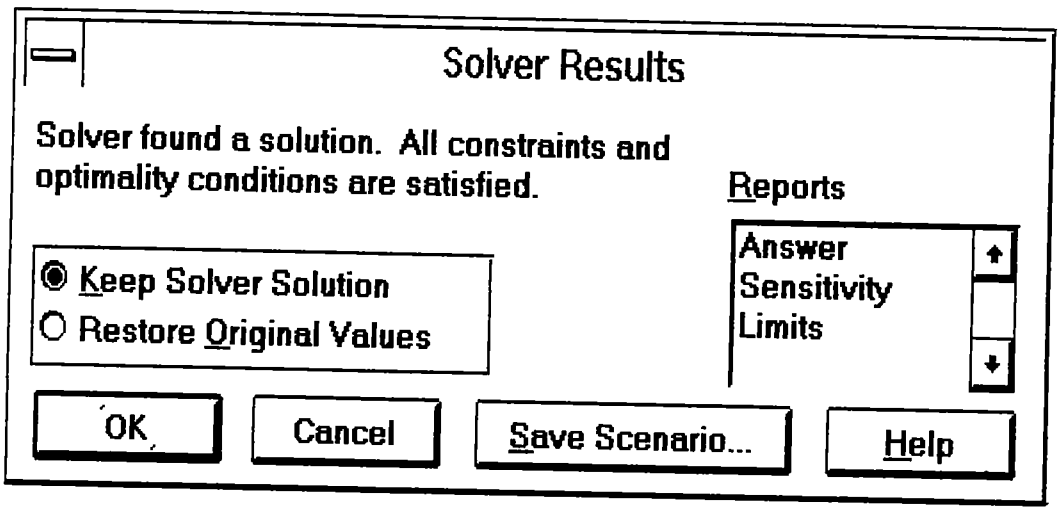



We choose the Add button again (either from the dialog above, or from the main Solver Parameters dialog) to define the non-negativity constraint on the decision variables When we've completely specified the problem, the Solver Parameters dialog looks like the one below




## Solving the Problem

To find the optimal solution, we simply click on the Solve button After a moment, the Solver returns the optimal solution 200 in cells D9 and E9, and 0 in cell F9 This means that we should build 200 TV sets, 200 stereos and no speakers, to earn a total profit of $25,000 (shown at cell D18) The message "Solver found a solution" appears in the Solver Results dialog, as shown below




We now click on "Answer" in the Reports list box to produce an Answer Report, and click OK to keep the optimal solution values in cells D9 F9 After a moment, the Solver creates another sheet containing an Answer Report, like the one below

EXAMPLE1.XLS

Microsoft Excel 5.0 Answer Report
Worksheet: [EXAMPLE1.XLS]EXAMPLE1
Report Created: 2/18/96 21:55

Target Cell (Max)

| Cell | Name | Original Value | Final Value |
|---|---|---|---|
| $D$18 | Total Profits | $16,000 | $25,000 |

Adjustable Cells

| Cell | Name | Original Value | Final Value |
|---|---|---|---|
| $D$9 | Number to Build-> TV set | 100 | 200 |
| $E$9 | Number to Build-> Stereo | 100 | 200 |
| $F$9 | Number to Build-> Speaker | 100 | 0 |

Constraints

| Cell | Name | Cell Value | Formula | Status | Slack |
|---|---|---|---|---|---|
| $C$11 | Chassis No Used | 400 | $C$11<=$B$11 | Not Binding | 50 |
| $C$12 | Picture Tube No Used | 200 | $C$12<=$B$12 | Not Binding | 50 |
| $C$13 | Speaker Cone No Used | 800 | $C$13<=$B$13 | Binding | 0 |
| $C$14 | Power Supply No Used | 400 | $C$14<=$B$14 | Not Binding | 50 |
| $C$15 | Electronics No Used | 600 | $C$15<=$B$15 | Binding | 0 |
| $D$9 | Number to Build-> TV set | 200 | $D$9>=0 | Not Binding | 200 |
| $E$9 | Number to Build-> Stereo | 200 | $E$9>=0 | Not Binding | 200 |
| $F$9 | Number to Build-> Speaker | 0 | $F$9>=0 | Binding | 0 |

Answer Report 1 | Sensitivity Report

This report shows the original and final values of the objective function (target cell) and the decision variables (adjustable cells), as well as the status of each constraint at the optimal solution Notice that the constraints on speaker cones and electronics are "binding" and have a "slack" value of 0 We used up all of the parts of these two types in building 200 TV sets and 200 stereos, but there were 50 units of each of the other parts left over If we could obtain additional speaker cones and electronics, we could further increase total profits, but extra units of the other parts would not help in the short run

If you've gotten to this point, congratulations! You have successfully set up and solved a simple Solver problem You can return to the Tutorial to learn more about the theory underlying the Solver and the relative difficulty of different problem types Or, to learn



more about good practices in Solver model design, consult
(25K)

# VITA

# VITA

Jayosree Mukherjee was born in Calcutta, India She finished her high school education in India, and later migrated to USA. She finished her Associate of Science Degree from Pellissippi State Community college majoring in Computer Science. She completed her undergraduate degree in Mathematics from University of Tennessee, Knoxville She continued her graduate studies in the University of Tennessee She also took additional education courses in the same university to pursue a high school teaching career. After serving one year as an intern in a high school in Knoxville, she is presently teaching in Stone Mountain High School, Atlanta