ASSIGNMENT OF VEHICLES TO ROUTES BY LATEX FOAM RUBBER PRODUCTS LIMITED-KUMASI

A THESIS SUBMITTED TO THE DEPARTMENT OF MATHEMATICS,

KWAME NKRUMAH UNIVERSITY OF SCIENCE AND TECHNOLOGY

KUMASI

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD

OF

MASTER OF SCIENCE DEGREE IN INDUSTRIAL MATHEMATICS

INSTITUTE OF DISTANCE LEARNING

BY

SAMUEL AMOAKO

B. Ed (Hons.)

Declaration

I hereby declare that this submission is my own work towards Master of Science degree and that, to the best of my knowledge, it contains no material previously published by another person nor material which has been accepted for the award of any other degree of the University, except where due acknowledgement has been made in the text.

Samuel Amoako	………………	……………
(PG 1834707)	Signature	Date
Student's Name and ID		

Certified by:

Mr. Kwaku Darkwah	………………..	……………..
Supervisor	Signature	Date

Certified by:

Professor E. Badu	…………………	………………
Dean, IDL	Signature	Date

Certified by:

Dr. S.K Amponsah	…………………	………………
Head of Department	Signature	Date

Dedication

This work is dedicated to my parents, Mr. P.K Amoako and Madam Grace Akyamah, all my brothers and sisters especially Mr. Robert K. Amoako of Ghana Prisons Service, my wife Alice and our children Vincent, Robert, Florence and Samuel for their unfailing and cheerful support.

Acknowledgement

I would like to express my gratitude to all those who helped me in diverse ways to complete this thesis.

I thank my thesis advisor, Mr. Kwaku Darkwah for his invaluable suggestions and supervision throughout the study.

I thank the Managers of Latex Foam Rubber Products Limited-Kumasi, for providing the data for the study.

I also thank Mr. Charles Sebil, a lecturer of Mathematics Department-KNUST, for his words of encouragement in the course of the study.

I would also like to thank my friends and course mates, Eric Ameyaw and Douglas Kparib, for their invaluable suggestions and remarks.

Last but not least, I would like to express my gratitude to my family for their support which enabled me to accomplish the study.

Abstract

In this thesis we focus on a decision model for a real world problem. The problem reveals itself as assignment of vehicles to routes by Latex Foam Rubber Products Limited-Kumasi.

This study addresses the problem of finding efficient assignments of the limited number of trucks at the company's disposal to the routes they ply while serving the company's customers outside the metropolis. The thesis seeks to minimize the total number of gallons of fuel needed for the assignments using the Munkres Assignment algorithm, a modified form of Kuhn's Hungarian algorithm.

TABLE OF CONTENTS

CHAPTER 2: LITERATURE REVIEW AND METHODS

LIST OF FIGURES

Figure *Page*

LIST OF TABLES

Table *Page*

CHAPTER 1

INTRODUCTION

The problem of distributing goods from depots to final consumers plays an important role in the management of many distribution systems, and its adequate programming may produce significant savings.

In a typical distribution system, vehicles provide delivery, pick-up or repair and maintenance services to customers that are geographically dispersed in a given area. In its numerous applications, the common objective of distribution is to find a set of routes for the vehicles to satisfy a variety of constraints so as to minimize the total fleet operation cost.

Most of the manufacturing companies in Ghana utilize vehicles (trucks) to transport their products to their customers. The general problem in such a situation is how to assign a particular vehicle to a route to minimize the total transportation cost whilst satisfying route and the available constraints to serve their customers with the demand for some commodity.

The Vehicle Assignment Problem, which is one of the logistics network problems, concerns the determination of the type of vehicle to assign to a particular route to minimize the total transportation cost.

In this thesis we use a solution procedure based on Munkres Assignment Algorithm for optimal assignment of non-homogenous fleet of vehicles to a given set of routes, where Latex Foam Rubber Products Limited-Kumasi, distributes its products to its customers.

1.1 THE ASSIGNMENT POROBLEM

The problem of assigning resources such as vehicles to task over time arises in a number of applications in transportation. In the field of freight transportation, truckload motor carriers, railways and shipping companies have to manage fleets of containers (trucks, boxcars) that move one load at a time, with orders arriving continuously over time. In the passenger arena, taxi companies and companies that manage fleets of business jets have to assign vehicles (taxicabs or jets) to move customers from one location to the next.

Ahuja, Magnanti and Orhin in Hartvigsen et al., (1999) provide an excellent review of applications of the assignment problem. Among the applications they listed are personnel assignments, scheduling on parallel machines, pairing stereo speakers and vehicle and crew scheduling. Other applications include posting military servicemen, airline commuting and classroom assignment.

1.1.2 *PRODUCTION AND DISTRIBUTION IN EALIER TIME*

The term Production is defined by Economists as the total physical and mental efforts which satisfy human wants. That is, production covers virtually all activities which directly or indirectly satisfy human wants. However, the term as used in production management refers to the transformation of raw materials into finished or semi-finished products (Mahmoud, 1996).

Distribution on the other hand concerns the series of activities and institutions which ensure the transfer of goods from the producer to the market. It basically involves the transfer or movement of goods from the producer to the consumer at the right time and at the appropriate place.

Before the Industrial Revolution (Clark et al, 1998), most goods were produced either by household or by guilds (Wiesner-Hanks, 2006). There were many households involved in the production of marketable goods. Most of the goods that were produced by these households were things that involved cloth, textiles, clothing, as well as art (Wiesner-Hanks, 2006) and tapestries (Jardine, 1996). These would be produced by the households, or by their respective guilds. It was even possible for guilds and merchants to outsource into more rural areas, to get some of the work done. These merchants would bring the raw materials to the workers, who would then make the goods. For example, young girls would be hired to make silk, because they were the only people believed to have hands dexterous enough to make the silk properly. Other occupations such as knitting, a job that was never organized into guilds, could easily be done within the household (Wiesner-Hanks, 2006)

Guild work could be contracted to the households for women and children, as well as the men would be involved with production of goods. The income of the household became dependent upon the quality and the quantity of everyone's work (De Vries, 1994). Even if people were not working for an individual guild they could still supply and make items not controlled by the guilds. These would be small, but necessary items like wooden dishes, or soaps (Wiesner-Hanks, 2006). So, basically, much of production was done by, or for, guilds. This would indicate that much of what was done was not done for one individual household, but for a larger group or organization.

Before the Industrial Revolution the household was the major site of production, and could be comparable to a factory.

However, things were to change a bit during the Industrial Revolution. There was a shift in the running of the household. The everyday goods and products used by the household would slowly shift from mostly home-made to mostly "commercially produced goods". At the same time, the women would obtain jobs outside the household (De Vries, 1994).

This is also seen within the context of the *Industrial Revolution* where women would often find small jobs to help supplement their husband's wages (Ross, 1993).This would demonstrate the gradual movement away from the household as a centre of production.

1.1.3 *INDUSTRIAL REVOLUTION*

The Industrial Revolution was a period in the late 18th and early 19th centuries when major changes in agriculture, manufacturing, production, and transportation had a profound effect on the socioeconomic and cultural conditions in Britain. The changes subsequently spread throughout Europe, North America, and eventually the world.

In the later part of the 1700s there occurred a transition in parts of Great Britain's previously manual-labor-based economy towards machine-based manufacturing. It started with the mechanization of the textile industries, the development of iron-making techniques and the increased use of refined coal. Trade expansion was enabled by the introduction of canals, improved roads and railways.

The introduction of steam power fuelled primarily by coal, wider utilization of waterwheels and powered machinery (mainly in textile manufacturing) underpinned the dramatic increases in production capacity. The development of all-metal machine tools in the first two decades of the 19th century facilitated the manufacture of more production machines for manufacturing in other industries. The effects spread throughout Western Europe and North America during the 19th century, eventually affecting most of the world. The impact of this change on society was enormous (Wikipedia, 2009).

The Industrial Revolution marked a major turning point in human society; almost every aspect of daily life was eventually influenced in some way.

1.1.4 *PRODUCTION AND DISTRIBUTION TODAY*

The revolutions in transportation and communications technologies have increased the extent of the U.S. domestic markets over the last two centuries. Moreover, the expansion of markets is associated with major changes in the course of American economic history. The introduction of canals in the late eighteenth and the early nineteenth centuries is credited with increasing the levels of inventive activity and triggering industrialization (Sokoloff, 1988). Households became less self-sufficient and became specialized consumer-labourers; firms that specialized in the production of various goods emerged in great numbers. The division of labour within firms led to a re-organization of production and increased levels of productivity (Sokoloff, 1984a, 1984b).

In the late 18th and the early 19th centuries, the expansion of the U.S. domestic markets and industrialization caused a rapid decline in household production and a proliferation of specialized manufacturing firms in the American economy (Kim, 2000). In this period, the industrial structure was composed of single-unit firms who specialized in the production of manufacturing goods and wholesale merchants and retail store owners who distributed these goods. Since the manufacturing firms typically specialized in a narrow line of products, it was simply too costly for them to market their products directly to consumers. In this setting, the wholesale merchants, who bought and sold sufficient lines of products, were able to lower the costs of transactions more efficiently. The wholesale merchants were not only able to collect information on various manufacturers by locating in major cities but were also able to collect information on rural consumer demand through the use of sales agents who traveled to rural country stores. In this period, most consumers were able to judge the quality of most products upon visual inspection. However, according to Kim, for some goods, they relied on the local producers' and retail merchants' reputation for honesty.

In the late nineteenth century, with advances in science and technology, it became increasingly difficult for consumers to discern the quality of products which they consumed. As incomes rose, consumers purchased a growing number of products for which they lacked basic knowledge to discern quality. Moreover, Kim indicated that, even the manufacturing processes of the most basic of products such as food became so sophisticated that consumers no longer had enough knowledge to discern whether a product was healthy or poisonous.

Finally, as regional domestic markets became increasingly integrated between the late 19th and the early 20th centuries, geographic specialization in economic activities increased (Kim, 1995).

1.1.4.1 CHANNELS OF DISTRIBUTION IN GHANA

Distribution could be broadly classified into Direct and Indirect distributions. There is direct distribution if the producer supplies the product directly to the consumer without the use of an intermediary or middle man. Indirect distribution involves the use of intermediaries or middlemen and retailers to make the product available to the consumer.

According to Mahmoud (1996) there are three main channels of distribution of goods in Ghana. These are from the

i. *Producer* to *Consumer*, where the producer sells directly to the consumer,

ii. *Producer* to the *Retailer* and from the *Retailer* to the *Consumer*, where the wholesaler is by passed and the producer deals directly with the retailer, and

iii. *Producer* to the *Wholesaler*, from the *Wholesaler* to the *Retailer* and from the *Retailer* to the *Consumer*, where the wholesaler buys in bulk from the producer and stores the goods for later resale to retailers.

1.1.4.2 *CHANNELS OF DISTRIBUTION USED FOR INDUSTRIAL PRODUCTS IN GHANA*

Industrial producers or sellers in Ghana today use four main channels to distribute their products in the country (Mahmoud, 1996). These are from the

i. *Producer to Consumer*: Most industrial producers such as Tema Steel Works, the Timber Processing organizations and Vehicle or Machine component manufacturing companies use this channel of distribution

ii. *Producer to Industrial distributor (customer)*: Some producers of industrial products use industrial distributors to market their products in Ghana.

iii. *Producer to an Agent, and from the Agent to the Customer*: This is the most popular method foreign organizations use when entering the Ghanaian market. Most of the organizations deal in office equipment, machines, vehicles installations and industrial raw material. Their Ghanaian counterparts provide after-sales service, training and installation services on behalf of their principals.

iv. *Producer* to an *Agent*, from the *Agent* to an *Industrial distributor* and from the *Industrial distributor* to the *Customer*: A good example of organizations involved in this sort of channel is Mechanical Lloyd- an agent of Yokohama tires in the country which markets these tires through a wide network of industrial distributors.

1.2 LATEX FOAM RUBBER PRODUCTS GHANA LIMITED

Latex Foam Rubber Products Limited was incorporated on March 8, 1969 in Accra to produce quality foam products for the Bedding and Furniture Industry in Ghana.

The company entered the Ghanaian market using the Dunlop Technology under license from the Dunlop Company. The technology gave Latex Foam the desired push in quality in a rather traditional market at the time. Since then the company has not relented in its efforts to assert itself in the Foam Industry.

In 1972, three years after its inception, Latex Foam started the production of Spring Interior Mattresses. Today the company has stood the test of time and is the oldest in the industry in Ghana. It is also the leading manufacturer of quality foam products such as

(i) Foam Mattresses (e.g. Ultraflex, Ultrafirm and High Density Honeymoon mattresses)

(ii) Pillows (e.g. Orthopedic pillows, Dona pillows and Venus pillows),

(iii) Mattress Accessories (Divan Bed, Comforter and Protection Pad),

(iv) Sofa beds, Students mattresses, Upholstery and

(v) Therapeutic products, such as Reader's Pillow and Back Care Cushion, in Ghana and West Africa.

In 2007, because of the high quality of its products, the company was chosen to provide the mattresses for the houses that hosted the visiting Heads of States for the Ghana@50 celebrations. That year, the company became the first to produce high resilient foam for the Ghanaian market when they introduced the Ultra flexes Mattress which provides excellent relaxation and body support.

1.3 BACKGROUND OF THE STUDY

Having consolidated its expansion program in Accra, Latex Foam on 12th September, 1996 established another factory in Kumasi in the premises of GIHOC Shoe Factory at Atonsu- Agogo, with the aim of increasing its proximity to its numerous customers in the northern sector of Ghana.

The main objective of the company is to continue to be the leading manufacturer of quality foam products and also satisfy its numerous customers in the northern sector and parts of Eastern and Western regions of Ghana by providing them with quality and innovative foam products.

The factory occupies an area of about 6000 square feet. At the moment, the company has about one hundred and fifty workers for the production and distribution of its cherished products. The structure that houses the machines of the company has three sections: the offices of the Personnel and Sales Managers of the company are on the left, the various manufactured products are stacked on the right, with the manufacturing machines at the extreme end of the shed, when you enter the shed through the main gate. Close to where the products are packed is a wooden structure which serves as a sales point for customers.

The company (Kumasi branch) has *three KIA* and *four TATA* trucks, a DAF cargo and a BENZ cargo trucks and *five articulator trucks* (one Renault, one DAF, one TATA and two BENZ) to distribute their products. These vehicles ply sixteen major routes in the distribution process outside Kumasi metropolis. These routes are indicated in Table 1.0 below.

Table 1.0: Major routes Latex Foam Rubber Products Limited-Kumasi ply in supplying products

ORIGIN	ROUTE	DESTINATION
1.Kumasi →	Sefwi Bekwai →	Sefwi Juabeso
2.Kumasi →	Dunkwa →	Asankraguaa
3.Kumasi →	Tamale →	Yendi
4.Kumasi →	Asante Bekwai →	Assin Fosu
5.Kumasi →	Techiman →	Kintampo
6.Kumasi →	Atebubu →	Kwame Danso
7.Kumasi →	Boodee →	Bogoso
8.Kumasi →	Sunyani →	Osei Kojokrom
9.Kumasi →	Zebilla →	Bawku
10.Kumasi →	Brekum →	Drobo
11.Kumasi →	Tepa →	Goaso
12.Kumasi →	Ejura →	Yeji
13.Kumasi →	Bolgatanga →	Lawra
14.Kumasi →	Juaso →	Obogu
15.Kumasi →	Asankare →	Nkawkaw
16.Kumasi →	Savlugu →	Gushiegu

1.4 PROBLEM STATEMENT

The aim of every business set-up is to optimize cost (to maximize profit or minimize the cost of operation) while meeting certain constraints. In order to satisfy the demand of its customers, Latex Foam Rubber Products Limited has to arrange the limited number (fourteen) of vehicles at its disposal to send their products to their various depots. The assignments of these vehicles are made depending on the time an order is placed for the products and the truck available at that time.

This thesis seeks to address the problem of finding efficient assignments of these fourteen vehicles to the sixteen major routes linking the factory to the termini destinations so as to minimize the total cost(number of gallons of diesel) required for transporting the company's products to its customers along these routes.

1.5 OBJECTIVE OF STUDY

The main objectives of the study are;

1. To determine the type of vehicle to assign to each of the routes leading to the final destinations (mostly district capitals) where Latex Foam Rubber Limited (Kumasi) has its depots or sales points, and
2. To minimize the total cost (number of gallons of diesel) needed to transport the products while satisfying routing constraints to serve their customers with the demand for the commodity.

1.6 METHODOLOGY

Data on the types of vehicles, number of gallons of diesel used per trip by each of the vehicles to transport latex foam products, and final destinations of these vehicles, will be obtained from the sales manager of the company through questioning.

The Munkres Assignment algorithm, which best solves assignment problems, will be employed. The algorithm takes the cost matrix of the assignment problem as input and proceeds by manipulating rows and columns through addition and subtraction to find the optimal assignment.

The problem will be solved using MATLAB computer program.

Search on the internet will be used to obtain the related literature. Books from the main Library at KNUST and the Mathematics Department's library will be read in the course of the project.

1.7 THESIS ORGANIZATION

Chapter one covers the historical background of the Vehicle Assignment Problem and how and when production and distribution of goods started. Chapter two contains the Literature Review and Methods. Chapter three covers data collection, analysis and discussion. The last chapter covers conclusion and recommendations.

CHAPTER 2

LITERATURE REVIEW AND METHODS

2.0 REVIEW OF LITERATURE

2.1 **The Vehicle Routing Problem (VRP)**

The VRP, in broad terms, deals with the optimal assignment of a set of transportation orders to a fleet of vehicles and the sequencing of stops for each vehicle. The VRP was first introduced by Dantzig and Ramser (1959), and was developed by Clarke and Wright (1964).

The main objective of the VRP is to minimize the distribution costs for the individual carriers, and can be described as the problem of assigning a collection of routes from a depot to a number of geographically distributed customers, subject to certain constraints. The most basic version of the VRP has also been called *vehicle scheduling*, *truck dispatching* or simply the *delivery problem* (Joubert, 2007). It has a large number of real life applications and comes in many forms, depending on the type of operation, the time frame for decision making, the objective and the type of constraint that must be adhered to. It is a computationally hard discrete optimization problem. The VRP has been a main subject for thousands of researchers since it was introduced by Dantzig and Ramser because of its important economic importance and has, therefore, gained much attention in recent years.

2.1.1 **Definition of the VRP**

The basic problem can be defined with G = (V, A) being a directed graph where $V = \{v_1, v_2, ...v_N\}$ is a set of vertices representing N customers and v_1 representing the depot where M identical vehicles, each with capacity Q, are located.

$E = \{(v_i, v_j) \mid v_i, v_j \in V, i \neq j\}$ is the edge set connecting the vertices. Each vertex, except for the depot ($V \setminus \{v_1\}$), has non negative demand q_i and non negative service time s_i .A matrix $C = (c_{ij})$ is defined on A. In some contexts, c_{ij} can be interpreted as travel cost, travel time or travel distance for any of the identical vehicles. The basic VRP is to route the vehicles one route per vehicle, each starting and finishing at the depot, so that all customers are supplied with their demands and the total travel cost is minimized.

The figure below illustrates how a solution to a VRP would look like after routes are generated. The sketch shows the vertices to be served (customers), the edges (route segments) and the depot.

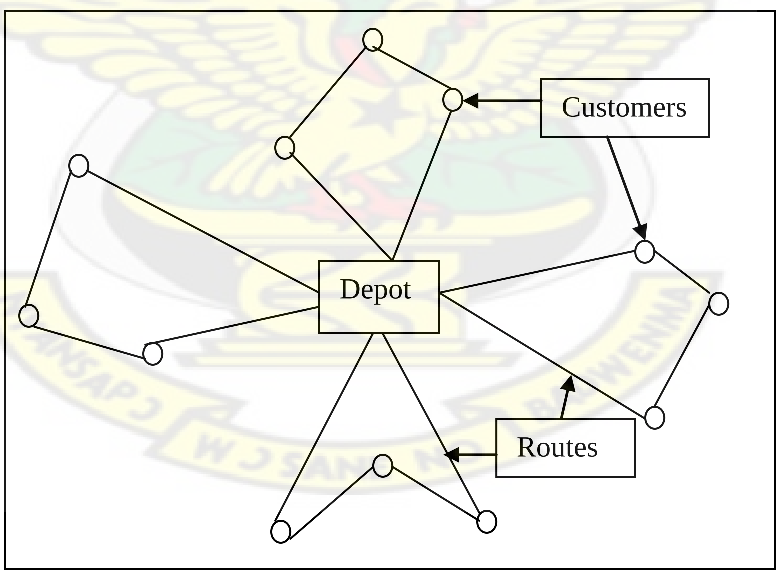


Figure 1: General representation of the Vehicle Routing Problem

2.1.1.1 **Variants of the VRP**

Several versions of the problem may be defined depending on the number of factors, constraints and the objective function addressed in the problem context. Some of the variants are

(i). *VRP with Time Windows (VRPWT)*: In the VRPWT, a number of vehicles is located at a central depot and has to serve a set of geographically dispersed customers with a demand within a specific time window.

(ii). Stochastic VRP: The stochastic vehicle routing problems arise when considering demands and travel times as stochastic variables. Other variants include heterogeneous vehicle fleet, simultaneous pick-up and delivery and periodic visits.

2.1.2 The Transportation Problem

The transportation problem is one of the subclasses of the linear programming problems for which simple and practical computational procedures have been developed that take advantage of the special structure of the problem.

Hitchcock (1941) was the first person to present, along with a constructive solution, the formulation of the transportation problem.

Koopman (1947) independently spearheaded research on the potentialities of linear programs for the study of problems in economics. Due to the fact that Kooprman's work was based on the work done earlier by Hitchcock, the classical case of the transportation problem is often referred to as Hitchcock-Koopman's transportation problem. The problem may be expressed as minimization of transport costs for moving a single commodity from *m* origins (sources) to *n* destinations (sinks) while operating within supply and demand constraints.

Hammer (1969) introduced the time-minimizing or bottleneck transportation problem, and the algorithm for solving the problem. Rather than minimizing cost, the objective is to minimize the maximum time to transport all supply to the destinations.

Sharma and Swarup (1978) and Bhatia et al. (1974) have given iterative methods for the solution of time-minimizing transportation problem.

Williams (1963) and Szwarc (1964) have discussed the stochastic transportation problems, i.e., problems with stochastic demand and penalties for over supply and under supply. The objective in such problems is to minimize total transportation cost plus expected penalty costs.

Wilson (1972, 1973, and 1975) showed that a linear approximation can be used in order to solve the stochastic transportation problem as a capacitated transportation problem.

Toth and Vigo (1997) examined the problem of determining an optimal schedule for a fleet of vehicles used to transport handicapped persons in an urban area, by using a Tabu Threshold procedure to the starting solution obtained by insertion algorithm.

2.1.3 **The Assignment Problem**

Assignment problems deal with the question of how to assign n number of items (e.g. jobs) to n number of machines (or workers) in the best possible way.

Michael and Powell (2004) addressed a simpler dynamic assignment problem, where a resource (container, vehicle, or driver) serves only one task at a time, using the language of Markov decision processes.

Woeginger et al (1995) used a branch and bound procedure to solve minimax assignment problems on tree networks. The problem involved the minimization of the maximum intermediate traffic by optimizing the message routing pattern and the embedding of communication centers.

Anshuman et al. (2007) solved the generalized "Assignment problem" using two non-traditional methods; genetic algorithm and simulated annealing. The generalized assignment problem is basically the "N men- N jobs" problem where a single job is assigned to only one person in such a way that the overall cost of assignment is minimized. While solving the problem through genetic algorithm (GA), a unique encoding scheme was used together with Partially Matched Crossover (PMX). In the simulated annealing (SA) method, an exponential cooling schedule based on Newtonian cooling process was employed.

Ye and Xu (2008) also developed a fuzzy chance-constrained model of vehicle routing assignment model according to fuzzy theory. In the model, they considered the total costs which included preparing costs of each type of vehicle and the transportation costs as the objective function and the preparing costs and the commodity flow demand as fuzzy variables, and minimized the total costs at a predetermined confidence level, α. They converted the fuzzy constraints into their crisp equivalents by using fuzzy theory and used a priority-based genetic algorithm to solve the problem.

2.2 REVIEW OF METHODS

2.2.1 The Transportation Problem

The transportation problem arises frequently in planning for the distribution of goods and services from several supply locations to several demand locations. Usually, the quantity of goods available at each supply location (origin) is fixed or limited and there is a specified amount needed (demand) at each user location (destination). With a variety of shipping routes and differing costs for the routes, the objective is to determine how many units should be shipped from each origin to each destination so that all destination demands are satisfied and the total transportation costs are minimized (David et al., 1988)

*2.2.1.1 **General Formulation of a Transportation Problem***

Let Z be the total distribution cost and x_{ij} the number of units to be distributed from source i to destination j. Let also s_i and d_j denote respectively the number of units being supplied by source i and the number of units being received by destination j and c_{ij} the unit cost of supplying s_i units from source i to destination j.

The transportation problem is generally formulated as

$$\textit{Minimize} \quad Z=\sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij} \text{.....................(1)}$$

$$\textit{subject to} \quad \sum_{j=1}^{n} x_{ij} \le s_i \quad (i=1,2,\ldots,m)\text{..............(2)} \qquad \text{(Supply constraints)}$$

$$\sum_{i=1}^{m} x_{ij} \ge d_j \quad (j=1,2,\ldots,n)\text{................(3)} \qquad \text{(Demand constraints)}$$

$$x_{ij} \ge 0 \quad (i=1,2,\ldots,m;\ j=1,2,\ldots,n)\text{.........(4)}$$

The objective function (1) is the total cost of transportation.

Constraint (2) requires that the total amount of commodity $\left(\sum_{j=1}^{n} x_{ij}\right)$ leaving source s_i must not exceed the production capacity of source s_i

Constraint (3) requires that the total amount of commodity $\left(\sum_{i=1}^{m} x_{ij}\right)$ arriving at destination d_j must not be less than the demand at destination d_j. Table 2.0 below shows the objective function $Z=\sum_{i}^{m}\sum_{j}^{n} c_{ij}x_{ij}$ as the total sum of elements in matrix table.

Table 2.0: Matrix of objective function

$c_{11}\ x_{11}$	$c_{12}\ x_{12}$	……	$c_{1n}x_{1n}$
$c_{21}\ x_{21}$	$c_{22}\ x_{22}$	…..	$c_{2n}x_{2n}$
. . .	. . .		. . .
$c_{m1}\ x_{m1}$	$c_{m2}x_{m2}$	. . .	$c_{mn}x_{mn}$

Table 2.1 shows the format of the transportation tableau. The row sum x_{ij} 's is less than or equal to s_i for each row and the column sum of each x_{ij} is greater than or equal to d_j for each column. The table is called the transportation tableau.

Table 2.1: Format of a transportation tableau

		Destination (j)				Supply (S_i)
		1	2		n	
Source (i)	1	x_{11} c_{11}	x_{12} c_{12}	……	x_{1n} c_{1n}	s_1
	2	x_{21} c_{21}	x_{22} c_{22}	…..	x_{2n} c_{2n}	s_2
	. .	. .	. .		. .	. .
	m	x_{m1} c_{m1}	x_{m2} c_{m2}		x_{mn} c_{mn}	s_m
Demand (d_i)		d_1	d_2	. . .	d_n	

2.2.1.2 ***A Balanced Transportation Problem***

In a “balanced transportation” problem, the total supply is equal to the total demand at any instant.

$$\sum_{i=1}^{m} s_i = \sum_{j=1}^{n} d_j$$

2.2.1.3 ***The Feasible Solutions Property of Transportation Problems***

According to Hiller and Lieberman (2005), a transportation problem will have a feasible solution if and only if

$$\sum_{i=1}^{m} s_i = \sum_{j=1}^{n} d_j$$

2.2.1.4 ***Integer Solutions Property of Transportation Problems***

For transportation problems, where every s_i and d_j has an integer value, all the basic variables (allocations) in every basic feasible solution (BFS) (including an optimal one) also have integer values (Hiller and Lieberman, 2005).

2.2.2 METHODS OF SOLVING TRANSPORTATION PROBLEMS

There are several methods for solving transportation problems. Two of such methods are the Stepping stone Method and Lagrangian Relaxation based methods.

These methods are variants of the Simplex Method. The methods use an initial BFS computed from methods like the Northwest corner rule or Vogel's Approximation Method, and improve upon the initial basic feasible solution to obtain an optimal solution.

Definitions

Cell: It is a small compartment in the transportation tableau.

Circuit: A circuit is a sequence of cells (in the balanced transportation tableau) such that

(i) It starts and ends with the same cell.

(ii) Each cell in the sequence can be connected to the next member by a horizontal or vertical line in the tableau.

Allocation: The number of units of items transported from a source to a destination which is recorded in a cell in the transportation tableau.

Basic Variables: The variables in a basic solution whose values are obtained as the simultaneous solution of the system of equations that comprise the functional constraints.

Basic Feasible Solution: A solution is called a basic feasible solution if

i. It involves (m + n -1) cells with non-negative allocations.

ii. There are no circuits among the cells in the solution.

2.2.2.1 FINDING INITIAL BASIC FEASIBLE SOLUTION OF BALANCED TRANSPORTATION PROBLEMS

2.2.2.1.1 ***The Northwest Corner Rule***

The North West corner rule is a method for computing an initial basic feasible solution of a transportation problem where the basic variables are selected from the North – West corner (i.e., succeeding top left corner) of the transportation tableau.

Given a balanced transportation problem in a transportation tableau,

(1) (i). Begin in the upper left (or northwest) corner of the transportation tableau.

(ii). Set x_{11} as large as possible. Clearly $x_{11} = \min\{s_1, d_1\}$.

(iii). If $x_{11} = s_1$, cross out row 1 of the transportation tableau; no more basic variables will come from row 1. Also set $d_1 = d_1 - s_1$.

(iv). If $x_{11} = d_1$, cross out column 1 of the transportation tableau; no more basic variables will come from column 1. Also set $s_1 = s_1 - d_1$.

(v). If $x_{11} = s_1 = d_1$, cross out either row 1 or column 1 (but not both).

- If you cross out row 1, set $d_1 = 0$
- If you cross out column 1, set $s_1 = 0$.

(2). Continue applying this procedure to the most northwest corner cell in the tableau that does not lie in the crossed-out row or column until you eventually reach a point where there is only one cell that can be assigned a value. Assign this cell a value equal to its row or column demand, and cross out the cell's row and column.

(3) A BFS has now been obtained.

Remark

In cases of Degeneracy, the solution obtained by the Northwest Corner method is not a basic feasible solution because it has fewer than (m + n – 1) cells in the solution. This happens because at some point during the allocation, when a supply is used up, there is no cell with unfulfilled demand in the column.

To resolve degeneracy a zero allocation is assigned to one of the unused cell.

Although there is a great deal of flexibility in choosing the unused cell for the zero allocation, the general procedure, when using the northwest corner rule, is to assign it to a cell in such a way that it maintains an unbroken chain of allocated cells.

Example 1: Consider the balanced transportation problem in Table 2.2 below, where the x_{ij}'s initially put to be $x_{ij} = 0$ or blank.

Table 2.2: A balanced transportation problem

		Destination				Supply
		1	2	3	4	
	1	6	5	7	9	40
Source	2	3	2	4	1	40
	3	7	3	9	5	25
Demand		30	20	35	20	

Applying the *Northwest Corner Rule*, we obtain the ordered allocations shown in Table 2.3 below. The number of circled allocation is 6 (that is 3 + 4 – 1 = 6) which gives the initial basic feasible solution. The arrows have been added to show the order in which the basic variables (allocations) were selected.

Table 2.3: Initial BF solution from the Northwest Corner Rule

		Destination				Supply
		1	2	3	4	
Source	1	6 30	5 10	7	9	40
	2	3	2 10	4 30	1	40
	3	7	3	9 5	5 20	25
Demand		30	20	35	20	

Hence the initial BFS is given by
$$\begin{bmatrix} x_{11} = 30, x_{12} = 10 \\ x_{22} = 10, x_{23} = 30 \\ x_{33} = 5, x_{34} = 20 \end{bmatrix}$$

Cost $Z = \sum\sum c_{ij}x_{ij} = (30\times 6) + (10\times 5) + (10\times 2) + (30\times 4) + (5\times 9) + (20\times 5)$

$= 515$

2.2.2.1.2 ***Vogel's Approximation Method (VAM)***

Vogel's approximation method has been a popular criterion for many years. VAM usually yields a better initial solution than the other initial basic feasible solution methods (Mathirajan and Meenakshi, 2004).

VAM is not quite as simple as the Northwest corner approach, but it facilitates a very good initial solution—as a matter of fact, one that is often the *optimal* solution.

Vogel's approximation method tackles the problem of finding a good initial basic feasible solution by taking into account the costs associated with each route alternative. This is something that the northwest corner rule does not do. To apply the VAM, the steps below are followed:

1. For each row and each column of the transportation tableau, we find the difference between the two lowest unit shipping costs. These numbers represent the difference between the distribution costs on the best route in the row or column and the second best least cost route in the row or column. It is also the opportunity cost.
2. We then identify the row or column with the greatest opportunity cost and assign the least of supply or demand capacities to the cell with the least cost of this row or column. Ties are broken arbitrarily.
3. We eliminate any row or column that has just been completely satisfied by the assignment just made and subtract the assignment from the supply or demand of row or column of the relevant assigned cell.

4. We re-compute the cost differences for the new transportation tableau, omitting rows or columns crossed out in the preceding step.
5. We then return to step 2 and repeat the steps until an initial feasible solution is obtained.

The method is illustrated by applying it to the balanced transportation problem in Table 2.3 of section 2.2.2.1.1 above; Table 2.4 shows the processes of obtaining solution.

Table 2.4a: Row and Column differences leading to elimination of column 4

		Destination					
		1	2	3	4	Supply	Row Difference
Source	1	6	5	7	9	40	1
	2	3	2	4	1	40-20	1
	3	7	3	9	5	25	2
Demand		30	20	35	20		
Column Difference		3	1	3	4		

Select $x_{24} = 20$

Eliminate column 4

Table 2.4b: Row and Column differences leading to elimination of column 2

		Destination 1	Destination 2	Destination 3	Supply	Row Difference
Source	1	6	5	7	40	1
	2	3	2	4	20	1
	3	7	3	9	25-20	4
Demand		30	20	35		
Column Difference		3	1	3		

Select $x_{32} = 20$

Eliminate column 2

Table 2.4c: Row and Column difference leading to elimination of row 2

		Destination 1	Destination 3	Supply	Row Difference
Source	1	6	7	40	1
	2	3	4	20	1
	3	7	9	5	2
Demand		30-20	35		
Column Difference		3	3		

Select $x_{21} = 20$

Eliminate row 2

Table 2.4d: Row and column differences leading to elimination of column 3

		Destination 1	Destination 3	Supply	Row Difference
Source	1	6	7	40-35	1
	3	7	9	5	2
Demand		10	35		
Column Difference		1	2		

Select $x_{13} = 35$

Eliminate column 3

Table 2.4e: Selection of column 1 for being the only column left

		Destination 1	Supply
Source	1	6	5
	3	7	5
Demand		10	

Select $x_{11} = 5, x_{31} = 5$

Hence, the initial BFS is given by $\begin{bmatrix} x_{11} = 5, x_{13} = 35 \\ x_{21} = 20, x_{24} = 20 \\ x_{31} = 5, x_{32} = 20, \end{bmatrix}$

Cost, $Z = \sum\sum c_{ij}x_{ij} = (20\times1)+(20\times3)+(20\times3)+(35\times7)+(5\times7)+(5\times6)$

$= 450$

2.2.3 METHODS FOR SOLVING TRANSPORTATION PROBLEMS TO OPTIMALITY

2.2.3.1 **The Stepping Stone Method**

This method determines the alternate cell with no allocation that would reduce cost if used.

Consider the balanced transportation problem shown in Table 2.2.

Suppose that the BFS of this problem consists of (m + n – 1) non negative allocations (occupied) cells. Let the cells that are not in the BFS be known as *unoccupied cells*.

The stepping Stone method uses the steps below to obtain an optimal solution to the transportation problem;

1. Test for optimality: For each of the unoccupied cells, form a circuit of horizontal and vertical lines, beginning with a plus (+) sign at the unoccupied cell. Thereafter place alternate minus (-) and plus (+) signs on each corner cell of the closed path traced, with the unoccupied cell being a corner cell and the other corners cells being occupied cells.

2. Using the unit cost of each cell, a closed path is formed for the unoccupied cell. We sign each unit cost by the relevant plus or minus. The total change in cost for the unoccupied cell that was used to form the circuit is computed. This change in cost is called *improvement index* of the unoccupied cell.
3. (i) If the improvement index of each unoccupied cell in the BFS is non negative, then the current BFS is optimal since any re-allocation increases the cost.

   (ii) If there is at least one unoccupied cell with a negative improvement index, then a re-allocation to produce a new BFS with a lower cost is possible. Go to step 4.
4. Improvement to optimality. To get a new BFS,
   i. We find the unoccupied cell whose circuit produced the most negative improvement index.
   ii. Using the above circuit, we find the smallest allocation in the cells of the circuit with the " – " sign and denote this smallest allocation by *m*. Subtract *m* from the allocations in all the cells in the circuit with " – " sign and add to all the allocations in the cells in the circuit with " + " sign. This has the effect of satisfying the constraints on demand and supply in the transportation tableau.
   iii. Since the cell which carried the allocation *m* now has a zero allocation, it is deleted from the solution and is replaced by the cell in the circuit which was originally unoccupied and now has an allocation *m*.
   iv. The result of the re- allocation is a new basic feasible solution. The cost of this new basic feasible solution is *m* less than the cost of the previous BFS.

v. Using the new BFS, go to step 1.

Example

Consider the balanced transportation problem in Table 2.2 above.

From the *Northwest Corner Rule*, the initial basic feasible solution is shown circled in Table 2.5 below and with the cost Z = 515

Table 2.5: Northwest Corner rule BFS

		Destination				
		1	2	3	4	Supply
	1	6 30	5 10	7	9	40
Source	2	3	2 10	4 30	1	40
	3	7	3	9 5	5 20	25
Demand		30	20	35	20	

The cost associated, Z = 515.

<u>First Iteration</u>

<u>Test for Optimality</u>

The unoccupied cells are (1, 3), (1, 4), (2, 1), (2, 4), (3, 1) and (3, 2).

Computing improvement indices for the unoccupied cells:

For (1, 3):

The circuit is (1, 3) → (2, 3) → (2, 2) → (1, 2) → (1, 3)

+   -   +   -   +

Improvement index = 7 – 4 + 2 – 5 = 0

For (1, 4):

The circuit is (1, 4) → (3, 4) → (3, 3) → (2, 3) → (2, 2) → (1, 2) → (1, 4)

+   -   +   -   +   -   +

Improvement index = 9 – 5 + 9 – 4 + 2 – 5 = 6

For (2, 1):

The circuit is (2, 1) → (2, 2) → (1, 2) → (1, 1) → (2, 1)

+   -   +   -   +

Improvement index = 3 – 2 + 5 – 6 = 0

For (2, 4):

The circuit is (2, 4) → (2, 3) → (3, 3) → (3, 4) → (2, 4)

+   -   +   -   +

Improvement index = 1 – 4 + 9 – 5 = 1

For (3, 1):

The circuit is (3, 1) → (1, 1) → (1, 2) → (2, 2) → (2, 3) → (3, 3) → (3, 1)

+ - + - + - +

Improvement index = 7 – 6 + 5 – 2 + 4 – 9 = - 1

For (3, 2):

The circuit is (3, 2) → (3, 3) → (2, 3) → (2, 2) → (3, 2)

+ - + - +

Improvement index = 3 - 9 + 4 – 2 = - 4

Improvement to optimality

The unoccupied cell with the most negative improvement index is **(3, 2).**

The least allocation to the cells in the circuit of (3, 2) with minus sign is 5. Subtracting this from the allocation of cells with the sign “-“, in the circuit and adding it to the allocations in the cells in the circuit with the sign “+”, we obtain the following new basic feasible solution as shown in Table 2.6

Table 2.6: New BFS obtained from Stepping Stone Method

From \ To	1	2	3	4	Supply
1	6 30	5 10	7	9	40
2	3	2 5	4 35	1	40
3	7	3 5	9	5 20	25
Demand	30	20	35	20	

The new BFS is $x_{11} = 30, x_{12} = 10, x_{22} = 5, x_{23} = 35, x_{32} = 5$ and $x_{34} = 20$

Second Iteration

Test for Optimality

The unoccupied cells in the new solution are (1, 3), (1, 4), (2, 1), (2, 4), (3, 1), (3, 3). The improvement indices are shown in Table 2.7

Table 2.7: Improvement indices of unoccupied cells in Table 2.6

Cell	**(1, 3)**	**(1, 4)**	**(2, 1)**	**(2, 4)**	**(3, 1)**	**(3, 3)**
Improvement Index	*0*	*2*	*0*	*-3*	*3*	*4*

Since there is an unoccupied cell with a negative improvement index, it follows that the current BFS is not optimal.

Improvement to optimality

The unoccupied cell with the most negative improvement index is (2, 4). The least allocation in the cells in its circuit with the sign "-"is 5. Subtracting it from the allocation in the other cell in the circuit with the sign "-" and adding to the allocations in the cells in the circuit with the sign "+", we obtain the new feasible solution in Table 2.8.

Table 2.8: Second BFS using Stepping Stone method

From \ To	1	2	3	4	Supply
1	6 30	5 10	7	9	40
2	3	2	4 35	1 5	40
3	7	3 10	9	5 15	25
Demand	30	20	35	20	

The new BFS is $x_{11} = 30, x_{12} = 10, x_{23} = 35, x_{24} = 5, x_{32} = 10$ and $x_{34} = 15.$

Third Iteration

Test for Optimality

The unoccupied cells in the current basic feasible solution are (1, 3), (1, 4), (2, 1), (2, 2), (3, 1) and (3, 3).The improvement indices are shown in Table 2.9.

Table 2.9: Improvement indices of unoccupied cells in Table 2.8

Cell	**(1, 3)**	**(1, 4)**	**(2, 1)**	**(2, 2)**	**(3, 1)**	**(3, 3)**
Improvement Index	*-3*	*2*	*3*	*3*	*3*	*1*

Since there is an unoccupied cell with a negative improvement index, it follows that the current basic feasible solution is not optimal.

The unoccupied cell with the most negative improvement index is (1, 3). The least allocation to the cells in this circuit with the sign "-" is 10. . Subtracting it from the allocation in the other cell in the circuit with the sign "-" and adding to the allocations in the cells in the circuit with the sign "+", we obtain the new feasible solution in Table 2.10.

Table 2.10: Optimal solution from Stepping Stone method

From \ To	A	B	C	D	Supply
1	6 30	5	7 10	9	40
2	3	2	4 25	1 15	40
3	7	3 20	9	5 5	25
Demand	30	20	35	20	

The new BFS is $x_{11} = 30, x_{13} = 10, x_{23} = 25, x_{24} = 15, x_{32} = 20$ and $x_{34} = 5$.

Fourth Iteration

Test for optimality

The unoccupied cells in the current basic feasible solution are (1, 2), (1, 4), (2, 1), (2, 2), (3, 1) and (3, 3). The improvement indices are shown in Table 2.11.

Table 2.11: Improvement indices of unoccupied cells in Table 2.10

Cell	(1, 2)	(1, 4)	(2 , 1)	(2, 2)	(3, 1)	(3, 3)
Improvement Index	3	5	0	3	0	1

Since there is no unoccupied cell with a negative improvement index, it follows that the current basic feasible solution is optimal. The optimal solution is given by $x_{11} = 30, x_{13} = 10, x_{23} = 25, x_{24} = 15, x_{32} = 20$ and $x_{34} = 5$ with the minimum cost of 450.

2.2.3.2 **Lagrangian Relaxation Based Methods**

One of the most computationally useful ideas of the 1970s is the observation that many hard problems can be viewed as easy problems complicated by a relatively small set of side constraints. Making the side constraints dual produces a Lagrange problem that is easy to solve, and whose optimal value is a lower bound (for minimization problems) on the optimal value of the original problem.

The "birth" of Lagrangian approach as it exists today occurred in 1970 when Held and Karp (1970, 1971) used a Lagrangian problem based on minimum spanning trees to devise a dramatically successful algorithm for the traveling salesman problem. Motivated by Held and Karp's success, Lagrange methods were applied in the early 1970s to scheduling problems (Fisher, 1973). Lagrangian methods had gained considerable currency by 1974 when Geoffrion (1974) coined the perfect name for this approach – "Lagrangian Relaxation".

2.3.2.1 ***Equality Constraints for Lagrangian Function***

Given the problem $P1$: minimize $f(x)$

$$\text{subject to } g(x) = b, \quad x \in X .$$

The Lagrangian function is defined to be

$$L(x, \lambda) = f(x) + \lambda^T (b - g(x)).$$

The components $\lambda = (\lambda_1, ..., \lambda_m)$ are known as the Lagrange multipliers.

2.3.2.2 ***Inequality Constraints and Complementary Slackness***

When the functional constraints in the problem P1 are in inequality form the problem becomes

$$P2: \text{ minimize } f(x)$$
$$\text{subject to } g(x) \le b, \quad x \in X.$$

It may be expressed in the previous form with equality constraints using slack variables as

$$\text{P3:minimize } f(x),$$
$$\text{subject to } g(x) + z = b, x \in X \text{ and z} \ge 0.$$

The Lagrangian now becomes

$$L(x, z, \lambda) = f(x) + \lambda^T (b - g(x) - z),$$

and it must be minimized over $x \in X$ and $z \ge 0$.

Consider the term in the Lagrangian involving $-\lambda_i z_i$; if $\lambda_i > 0$ then letting z_i become arbitrarily large shows that this term can be made to approach $-\infty$ which implies that $\inf_{x \in X, x \ge 0} L(x, z, \lambda) = -\infty$. Thus, for a finite minimum of the Lagrangian we require that $\lambda_i \le 0$, in which case the minimum of the term $-\lambda_i z_i$ is 0, since we could take $z_i = 0$.

Thus, with the inequality constraints in the problem, minimizing the Lagrangian always leads to sign conditions on the Lagrange multipliers, in this case $\lambda \le 0$. There is also a joint condition on the Lagrange multipliers and the slack variables in that

$\lambda_i z_i = 0 \quad$ for each $i = 1, ..., m,$ or equivalently, $\lambda^T z = 0.$

This condition is known as a complementary slackness condition; at least one of the variables λ_i and z_i must be zero (at the optimum solution) for each i.

2.3.2.3 *Lagrange Multipliers and the Transportation Problem*

A classical optimization problem is the transportation problem in which there are m sources of supply of a particular good $\{S_1,...,S_m\}$, with amounts $\{s_1,...,s_m\}$ available, and n destinations $\{D_1,...,D_n\}$ at which there are demands $\{d_1,...,d_n\}$, respectively for the good. For each pair $\{S_i, D_j\}$, there is a cost c_{ij} per unit for shipping from S_i to D_j.

***Assumption*:** $\sum_{i=1}^{m} s_i = \sum_{j=1}^{n} d_j$, that is, total supply equals total demand.

The objective is to satisfy the demand from the supplies with the minimal transportation cost. Let x_{ij} denote the flow from S_i to D_j.

The transportation problem is the linear programming problem formulated as

$$\textit{Minimize} \quad Z=\sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij} \text{.....................(1)}$$

$$\textit{subject to} \quad \sum_{j=1}^{n} x_{ij} = s_i \quad (i = 1,2,...,m) \text{ ....(2)} \qquad \text{(Supply constraints)}$$

$$\sum_{i=1}^{m} x_{ij} = d_j \quad (j = 1,2,...,n) \text{ ....(3)} \qquad \text{(Demand constraints)}$$

$$x_{ij} \geq 0 \quad (i = 1,2,...,m;\ j = 1,2,...,n) \text{ ....(4)}$$

Let λ_i and v_j be the Lagrange multipliers.

The Lagrangian for the balanced transportation problem is

$$L(x,\lambda,v)=\sum_{i=1}^{m}\sum_{j=1}^{n}c_{ij}x_{ij}+\sum_{i=1}^{m}\lambda_i\left(s_i-\sum_{j=1}^{n}x_{ij}\right)+\sum_{j=1}^{n}v_j\left(d_j-\sum_{i=1}^{m}x_{ij}\right)$$
$$=\sum_{i=1}^{m}\sum_{j}^{n}(c_{ij}-\lambda_i-v_j)x_{ij}+\sum_{i=1}^{m}\lambda_i s_i+\sum_{j=1}^{n}v_j d_j$$

The minimum of the Lagrangian over $x_{ij}\geq 0$ will be finite provided:

$$c_{ij}-\lambda_i-v_j\geq 0, \qquad \text{for each } i, j \qquad \text{(dual feasibility)}$$

and at the optimum

$$(c_{ij}-\lambda_i-v_j)x_{ij}=0, \qquad \text{for each } i, j. \qquad \text{(complementary slackness)}$$

The steps for Lagrangian procedure for solving balanced transportation problems are then indicated as follows;

1. **Initial assignment**. We start the algorithm by choosing an initial basic feasible solution (BFS) by the Northwest method.
2. **Assign the Lagrangian multipliers.** Next, we choose the values for the Lagrange multipliers $(\lambda_i),(v_j)$ so that $c_{ij}-\lambda_i-v_j=0$ for the basic cells; this ensures that the complementary slackness holds. Since only the sum λ_i+v_j enter into all the calculations one of these multipliers may be chosen arbitrarily, $\lambda_1=0$, say.
3. **Test for optimality.** We identify the non-basic cells for which $c_{ij}-\lambda_i-v_j<0$; if all cells have $c_{ij}-\lambda_i-v_j\geq 0$ then the current solution is optimal. Otherwise go to step 4.

4. **Pivoting.** Choose the non-basic cell with the most negative value of $c_{ij} - \lambda_i - v_j$ (Pivote cell). Put an amount ϵ > 0 units of flow into the pivot cell. At the same time, add or subtract from the basic cells to maintain feasibility. Now choose the largest ϵ possible such that the flow is feasible.

5. The algorithm now returns to step 2 with this flow as the basic feasible flow**.**

Let us apply these steps to the initial BFS obtained from the Northwest method in Table 2.5 with the total cost of flow of 515.

First Iteration

Step 2: We choose values for the Lagrange multipliers $(\lambda_i),(v_j)$ so that $c_{ij} - \lambda_i - v_j = 0$ for the basic cells. We obtain the following equations;

$$
\begin{aligned}
6 - \lambda_1 - v_1 &= 0 \\
5 - \lambda_1 - v_2 &= 0 \\
2 - \lambda_2 - v_2 &= 0 \\
4 - \lambda_2 - v_3 &= 0 \\
9 - \lambda_3 - v_3 &= 0 \\
5 - \lambda_3 - v_4 &= 0
\end{aligned}
$$

Letting $\lambda_1 = 0,$ we obtain $v_1 = 6, v_2 = 5, \lambda_2 = -3, v_3 = 7, \lambda_3 = 2$ and $v_4 = 3$

The values for λ_i and v_j obtained for the basic cells are as shown in Table 2.12.

Table 2.12: *Lagrange multipliers of basic cells*

λ_i \ v_j	6	5	7	3
0	$\boxed{6}$ 30	$\boxed{5}$ 10	$\boxed{7}$	$\boxed{9}$
-3	$\boxed{3}$	$\boxed{2}$ 10	$\boxed{4}$ 30	$\boxed{1}$
2	$\boxed{7}$	$\boxed{3}$	$\boxed{9}$ 5	$\boxed{5}$ 20

Step 3: Check for optimality: For each non-basic (unoccupied) cell we compute $c_{ij} - \lambda_i - v_j$ and identify those with $c_{ij} - \lambda_i - v_j < 0$. The non-basic cells and their $c_{ij} - \lambda_i - v_j$ values are shown in Table 2.13 below

Table 2.13:Non-basic cells and their c_{ij} - λ_i - v_j values

Non-basic cell	**Value of** $c_{ij} - \lambda_i - v_j$
(1,3)	7 - 0- 7 = 0
(1,4)	9 – 0 – 3 = 6
(2,1)	3 – (- 3) – 6 = 0
(2,4)	1 – (- 3) – 3 = 1
(3,1)	7 – 2 – 6 = - 1
(3,2)	3 – 2 – 5 = - 4

Since some of the $c_{ij} - \lambda_i - v_j$ values are negative, it means that the solution is not optimal and therefore the pivot operation must occur.

Step 4: The non basic cell with the most negative $c_{ij} - \lambda_i - v_j$ value is (3, 2). We increase the solution in this cell by ε and form a loop as shown in Table 2.14(a)

Table 2.14(a): New allocation with ε adjustment for first iteration

6 30	5 10	7	9
3	2 10 - ε	4 30 + ε	1
7	3 ε	9 5 - ε	5 20

We then increase ε until the allocation in one of the basic cells becomes zero; in this case when ε = 5, and this gives a new basic feasible solution as shown in Table 2.14(b) below.

Table 2.14(b): New BFS for First Iteration using Lagrange Multipliers

6 30	5 10	7	9
3	2 5	4 35	1
7	3 5	9	5 20

The solution is $x_{11} = 30, x_{12} = 10, x_{22} = 5, x_{23} = 35, x_{32} = 5$ and $x_{34} = 20$.

Total cost of flow, $Z = (6\times30)+(5\times10)+(2\times5)+(4\times35)+(3\times5)+(5\times20) = 490$

Second Iteration

The algorithm returns to step 2 with the current solution as the basic feasible solution. Following the steps in the first iteration we get λ = [0, -3, -2] and v = [6, 5, 7, 7]. An (X) has been placed in the non-basic cell for which $c_{ij} - \lambda_i - v_j < 0$. The numerical difference ($c_{ij} - \lambda_i - v_j$) of this non basic cell is - 3. Table 2.15(a) shows the new allocation with ε adjustment.

Table 2.15(a): New allocation with ϵ adjustment for Second Iteration

λ_i \ v_j	6	5	7	7
0	[6] 30	[5] 10	[7]	[9]
-3	[3]	[2] 5 - ε	[4] 35	[1] ε X (- 3)
- 2	[7]	[3] 5 + ε	[9]	[5] 20 - ε

Table 2.15(b) below shows the new BFS with total cost 485.

Table 2.15(b): New BFS for second iteration

6 30	5 10	7	9
3	2	4 35	1 5
7	3 10	9	5 15

Total cost = 485

The new BFS is $x_{11} = 30, x_{12} = 10, x_{23} = 35, x_{24} = 5, x_{32} = 10$ and $x_{34} = 15$.

Third Iteration

The algorithm returns to step 2 with the current solution as the basic feasible solution. Following the steps in the first iteration we get λ = [0, -6, -2] and v = [6, 5, 10, 7]. An (X) has been placed in the non-basic cell for which $c_{ij} - \lambda_i - v_j < 0$. The numerical difference ($c_{ij} - \lambda_i - v_j$) of the cell (i.e., (1, 3)) is – 3.

Table 2.16(a) shows the new allocation with ϵ adjustment.

Table 2.16(a): New allocation with ϵ adjustment for Third Iteration

λ_i \ v_j	6	5	10	7
0	$\boxed{6}$ 30	$\boxed{5}$ 10 - ε	$\boxed{7}$ ε X (-3)	$\boxed{9}$
-6	$\boxed{3}$	$\boxed{2}$	$\boxed{4}$ 35 - ε	$\boxed{1}$ 5 + ε
-2	$\boxed{7}$	$\boxed{3}$ 10 + ϵ	$\boxed{9}$	$\boxed{5}$ 15 - ε

The table below shows the new BFS with the total cost 450.

Table 2.16(b): New BFS for Third Iteration

$\boxed{6}$ 30	$\boxed{5}$	$\boxed{7}$ 10	$\boxed{9}$
$\boxed{3}$	$\boxed{2}$	$\boxed{4}$ 25	$\boxed{1}$ 15
$\boxed{7}$	$\boxed{3}$ 20	$\boxed{9}$	$\boxed{5}$ 5

Total cost = 450,

Since the all the non-basic cells in the above tableau satisfy the dual feasibility condition (i.e., $c_{ij} - \lambda_i - v_j \geq 0$), it means that the current basic feasible solution $x_{11} = 30, x_{13} = 10, x_{23} = 25\ x_{32} = 20, x_{34} = 5$ is optimal. The cost associated Z = 450.

2.3 THE ASSIGNMENT PROBLEM

The matching or assignment problem is one of the fundamental classes of combinatorial optimization problems. It is a special type of linear programming problem where agents are being assigned to perform tasks. The agents might be employees who need to be given work assignments. Assigning people to jobs is a common application of the assignment problem. However, the agents need not be people. They could be machines, vehicles, plants, or even time slots to be assigned tasks.

In its most general form, the assignment problem can be stated as follows: A number of *m* agents and a number of *n* tasks are given, possibly with some restrictions on which agent can perform which particular task. A cost is incurred for each agent performing some task, and the goal is to perform all tasks in such a way that the total cost of the assignment is minimized. Figure 2 shows the network representation of the assignment problem.

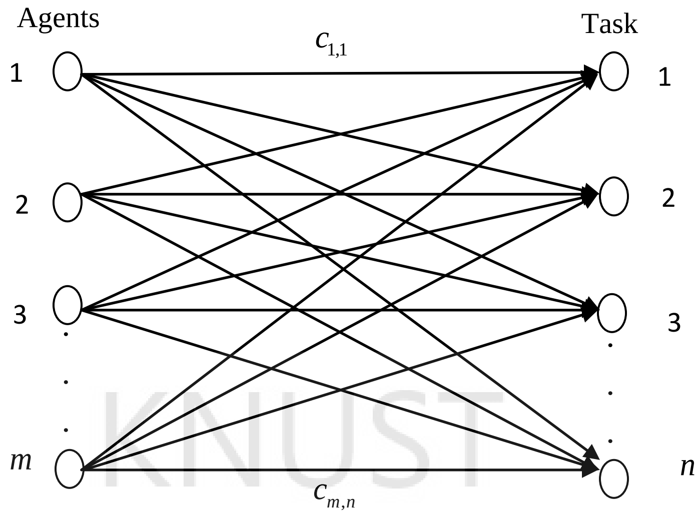


Figure 2: Network representation of assignment problem: c_{ij} denotes the cost of assigning agent i to task j.

2.3.1 *The Linear Assignment Problem (LAP)*

In the Linear Assignment Problem (LAP), the number of agents and tasks is the same and any agent can be assigned to perform any task. LAP is thus equivalent to the problem of finding an optimum weight vertex matching in an n × n cost-weighted complete bipartite graph.

2.4.2 *Formulation of Linear Assignment Problem*

Formally, LAP can be formulated as follows:

Given a set of agents $A = \{a_1, a_2, ..., a_n\}$ *and a set with the same number of tasks* $T = \{t_1, t_2, ..., t_n\}$ *and the cost function* $C : A \times T \to R$

Find a matching $m : A \to T$ *such that the cost function*

$\sum_{a\in A} C\left(a, m(a)\right)$, *is minimized.*

Usually the weight function (i.e. the cost function) is viewed as a square real-valued matrix C with elements $C_{ij} = c(a_i, t_j)$.

This problem can be expressed as an integer linear program with the objective function

$$\text{Minimize} \qquad Z = \sum_{i=1}^{n}\sum_{j=1}^{n} C_{ij} x_{ij} \text{.............}(4),$$

subject to the constraints

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

$$\sum_{j=1}^{n} x_{ij} = 1, \text{ for all } i \in \{1, 2, \ldots, n\} \qquad (6)$$

$$x_{ij} \in \{0, 1\}, \text{ for all } i, j \in \{1, 2, \ldots, n\}$$

The variable x_{ij} represents the assignment of agent i to task j, taking value 1 if the assignment is done and 0 otherwise.

Constraint (5) requires that every agent is assigned to exactly one task, and constraint (6) requires that every task is assigned exactly one agent.

Except for the assumed integrality of the decision variable x_{ij}, the assignment problem is just a balanced transportation problem in which

- the number of supply and demand nodes are equal
- supply from every supply node (agent) is one
- the demand at every demand node (task) is also one, and
- solution is required to be all integers.

The table below is a parameter table for the general assignment problem formulated as a transportation problem.

Table 2.17: Parameter table for assignment problem formulated as a transportation problem

		Task					Agent
		1	2		$n-1$	n	
Resource	1	c_{11}	c_{12}	……	$c_{1,n-1}$	$c_{1,n}$	1
	2	c_{21}	c_{22}	…..	$c_{2,n-1}$	$c_{2,n}$	1
	. .	. .	. .		. .	. .	. .
	.	$c_{m-1,1}$	$c_{m-1,2}$		$c_{m-1,n-1}$	$c_{m-1,n}$	1
	m	$c_{m,1}$	$c_{m,2}$		$c_{m,n-1}$	$c_{m,n}$	1
Activity		1	1	. . .	1	1	

For any linear assignment problem with n assignments to be made, the tableau shown in Table 2.17 has $m = n$, that is, both the number of agents (n) and the number of task (n) in this formulation equal the number of assignments (n).

Transportation problems in general have $m + n - 1$ basic variables (allocations), so every basic feasible solution of linear assignment problems has $2n - 1$ basic variables, but exactly n of these x_{ij} variables equals 1(corresponding to the n assignments being made). Therefore, since all the variables are binary variables, there are always (n – 1) degenerate variables $x_{ij} = 0$.

2.3.3 *The Vehicle Assignment Problem*

In its simplest form, the assignment problem can be formulated in terms of linear programming and solved with a help of simplex method, network algorithms (Cooke, 1985) or assignment method (Lotfi and Pegels, 1989).Some authors (Löbel, 1998), (Rushmeier and Kantogiorgis, 1997) formulate the vehicle assignment problem in terms of the linear, integer programming. Some others (Beaujon and Turnquist, 1991) transform the linear, discrete model into a non-linear, continuous form. Many models are based on the queuing theory (Green and Guha, 1995), (Whitt, 1992). The proposed models consider either the same capacity (Beaujon and Turnquist, 1991) or different capacity fleet (Ziarati et al., 1999). Some of the models combine the vehicle assignment problem with other fleet management problems, such as fleet sizing (Beaujon and Turnquist, 1991) or fleet scheduling (Rushmeier and Kantogiorgis, 1997).

The models usually refer to specific transportation environments, such as: urban transportation (Löbel, 1998), rail transportation (Ziarati et al., 1999) or air transportation (Rushmeier and Kantogiorgis, 1997). Zeleny (1992) proposes an extended multi-criteria model for the vehicle assignment problem and a solution procedure based on an assignment algorithm – Hungarian method (Bradley et al., 1977). The most popular solution procedures are decomposition techniques, such as: Frank-Wolfe's, Benders' or Dantzig-Wolfe's decomposition algorithms (Bradley et al., 1977). Heuristics and branch-and-bound algorithm are also utilized.

2.3.4 ***Variants of Assignment Problem***

i. *Multiple Optimum Solutions*

This is a situation whereby more than one optimal solution is obtained and we, therefore, have elasticity in decision making. Here one can choose any of the solutions by experience or by using further considerations.

ii. *Maximization case in Assignment Problem*

Some assignment problems entail maximizing the profit, effectiveness, or layoff of an assignment of agents to tasks or jobs to machines.

2.3.4.1 ***Unbalanced Assignment Problem***

It is an assignment problem where the number of agents is not equal to the number of tasks.

If the number of agents is less than the number of tasks then we introduce one or more *dummy agents* (rows) with zero cost values to make the assignment problem balanced. Likewise, if the number of tasks is less than the number of agents then we introduce one or more *dummy tasks* (columns) with zero cost values to make the assignment problem balanced.

2.3.4.2 ***Prohibited Assignment***

Sometimes it may happen that a particular resource (say a man or machine) cannot be assigned to perform a particular activity. In such cases, the cost of performing that particular activity by a particular resource is considered to be very high (written as M or ∞) so as to prohibit the entry of this pair of resource-activity into the final solution.

2.4 METHODS FOR SOLVING ASSIGNMENT PROBLEMS

Methods such as the Stepping Stone method and the Lagrange multipliers for solving transportation problems can be used to solve the assignment problem. However, due to its special characteristics, the Hungarian Method or Munkres Assignment Algorithm is usually used to solve such assignment problems.

2.4.1 *Introduction to the Hungarian Method*

A high degree of degeneracy in an assignment problem may cause the above mentioned methods to be inefficient in solving assignment problems. For this reason, and the fact that the algorithm is even much simpler than solution methods mentioned above, the Hungarian method is usually used to solve assignment problems.

The Hungarian Method was invented and published in 1955 by Harold Kuhn. The algorithm developed by Kuhn was largely based on the earlier works of two Hungarian mathematicians: Dénes König and Jenö Egerváry (Andras, 2004).

The main merit of Kuhn's Hungarian Method is that in the past half a century it has become the starting point of a fast developing area of efficient combinatorial algorithms. Its seminal ideas, developed originally for the weighted bipartite matching problem (that is, the assignment problem) have been applied by Ford and Fulkerson (1942) to the transportation problem and, more generally, to minimize cost flows, as well.

The algorithm is used to solve an assignment problem of $n \times n$ cost matrix where each element represents the cost of assigning the ith agent to the jth task. By default, the algorithm performs a minimization on the elements in the cost matrix.

2.4.2 The Hungarian Algorithm Due to Kuhn

Harold W. Kuhn, in his celebrated paper entitled *The Hungarian Method for the assignment Problem,* (Andras, 2004) described an algorithm for constructing a maximum weight perfect matching in a bipartite graph. Kuhn explained how the works of two Hungarian mathematicians, Dénes König and Jenö Egervåry, had contributed to the invention of his algorithm, the reason why he named it the Hungarian Method.

Definitions

- A ***graph*** is an ordered pair G = (V, E) consisting of a finite set V and a subset E of elements of the form(x, y) where x and y are in V. The elements in set V are called the vertices of the graph and those in set E are called the edges.
- A ***bipartite graph*** is a linear graph in which the nodes can be partitioned into two groups X and Y such that for every edge *(i, j)* node *i* is in X and node *j* is in Y. That is, a graph G = (V,E) is bipartite if there exists a partition $V = X \cup Y$ with $X \cap Y = \varnothing$ and $E \subseteq X \times Y$.
- The ***complete bipartite graph*** $K_{m:n}$ is the graph with bipartition {X; Y} where $|X| = m$ and $|Y| = n$, and each vertex of X is adjacent to every vertex of Y.
- A ***matching* M** of a general graph G = (V, E) is a subset of the edges with the property that no two of the edges of M share the same node. In other words, a

matching is a subset $M \subseteq E$ such that $\forall v \in V$ at most one edge in M is incident upon **v.**

- A matching M is ***perfect*** if every vertex in G is incident with an edge in the matching.
- The ***size*** of a matching is $|M|$ the number of edges in M.
- A ***path*** consists of a sequence of vertices from a starting vertex to an end vertex with edges linking successive vertices.

2.4.2.1 ***Alternating Paths***

Let **M** be a matching of graph **G**. Vertex **v** is *matched* if it is an endpoint of edge in **M**; otherwise **v** is *free* of the matching. If (x, y) is a matched edge, then y is the *mate* of x. Nodes that are not incident upon any matched edges are called e*xposed* (free) nodes. For example, in figure 3 below, the matched vertices are $x_2, x_3, x_4, x_6, y_2, y_4, y_5,$ and y_6, the matched edges (deep black edges) are the set $(x_2, y_2), (x_3, y_5), (x_4, y_4), (x_6, y_6)$ and the exposed nodes are $x_1, x_5, y_1,$ and y_3.

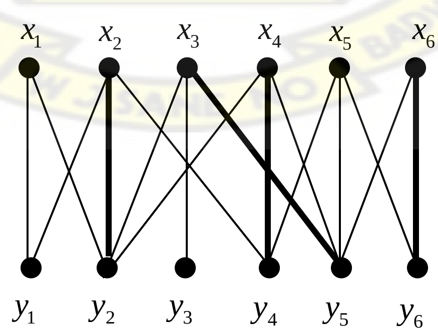


Figure 3: Graph showing matched vertices, matched edges and alternating paths

A path is alternating if its edges alternate between **M** and **E-M**. In figure 3 above the alternating paths are

(i) $x_1 \to y_2 \to x_2 \to y_4 \to x_4 \to y_5 \to x_3 \to y_3$

(ii) $y_1 \to x_2 \to y_2 \to x_3 \to y_5 \to x_6 \to y_6 \to x_5$ and

(iii) $y_1 \to x_2 \to y_2 \to x_4 \to y_4 \to x_5$

2.4.2.2 ***Augmenting Path***

An *alternating* path is a*ugmenting* if both endpoints are free or unsaturated. An augmenting path has one less edge in **M** than in **E-M**.

For example, in figure 4 below, vertices $x_1, y_2, x_2, y_4, x_4, y_5$ form an augmenting path.

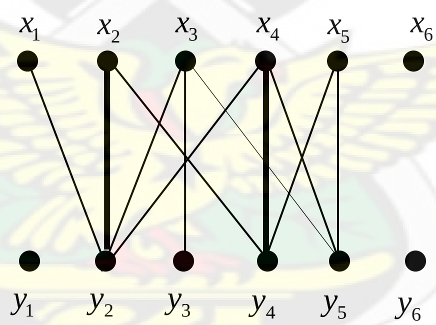


Figure 4: Graph showing Augmenting paths

Number of edges in **M** (i.e., number of deep black edges) in the augmenting path $x_1, y_2, x_2, y_4, x_4, y_5$ = 2

Number of edges **E** (i.e., the number of deep black edges plus the number of light black edges) in the augmenting path = 5

Therefore, the number of edges in **E-M** = 5 – 2 = 3, which is greater than the number of edges in **M** by 1.

2.4.2.3 ***Alternating Tree***

An *Alternating tree* is two or more alternating paths all ending on some free vertex v as the root. Considering the matching **M** in figure 5(a), x_5 is the root because at x_5 three alternating paths

(i) x_5, y_6, x_6,

(ii) x_5, y_5, x_3, y_3 and

(iii) $x_5, y_4, x_4, y_2, x_2, y_1$ form alternating trees as indicated in figure 5(b).

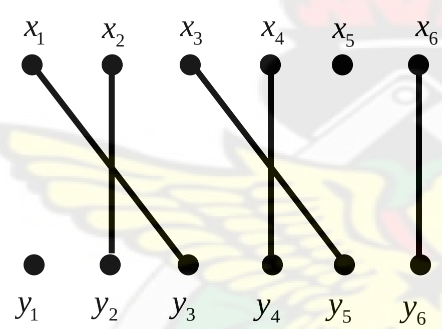


*Figure 5(a): A matching **M***

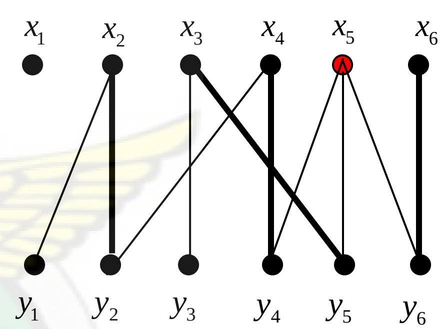


Figure 5(b): An alternating tree

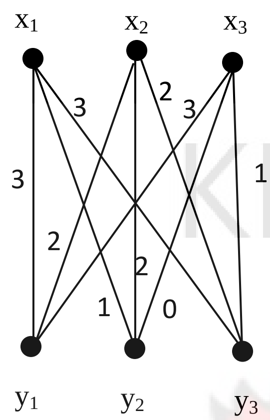
2.4.2.4 ***Weighted Matching Bipartite Graphs***

These are graphs in which each edge (i, j) has a weight or value $w(i, j)$. The weight of a matching M is the sum of the weights of the edges in M,

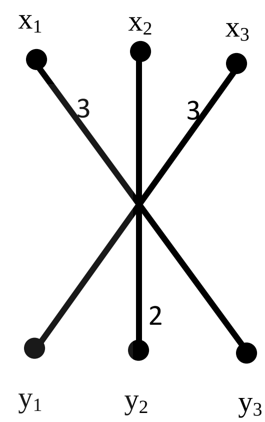
$$w(M) = \sum_{e \in M} w(e)$$

Each entry w_{ij} represents the weight of the edges between x_i and y_j.

Problem: Given a bipartite weighted graph G (figure 6(a)), find a maximum weight matching.



Solution:



(By inspection)

Figure 6(a): Bipartite weighted graph *Figure 6(b): Maximum weight matching*

2.5.2.5 ***Feasible Vertex Labeling***

Let *N* be a network with each edge *e* giving an integer weight *w (e).* A feasible vertex labeling for *N* is a function ℓ: V (N) → $\mathbb{R}$ such that $\ell(x) + \ell(y) \geq w(x, y)$ for all $x \in X$ and $y \in Y$. $\ell(x)$ and $\ell(y)$ are the labeling of vertices x and y respectively and w (x, y) is the maximum edge weight from vertex *x* to vertex *y.*

We define the size of ℓ by $size(\ell) = \sum_{v \in V(N)} \ell(v)$.

Lemma 1: *Let ℓ be a feasible vertex labeling for N and M be a perfect matching in N. Then w (M) $\leq$ size (ℓ).*

Proof: Let $\mathrm{M} = \{x_1 y_1, x_2 y_2, ..., x_m y_m\}$.

Then $w(M) = \sum_{i=1}^{m} w(x_i y_i) \leq \sum_{i=1}^{m} [\ell(x_i) + \ell(y_i)] = \sum_{v \in V} \ell(v) = size(\ell)$

, since ℓ is a feasible vertex labeling.

Lemma 1 implies that the maximum weight of a perfect matching in N is less than or equal to the minimum size of a feasible vertex labeling of N.

For example, let N be the weighted $K_{3:3}$ with bipartition X = {x_1, x_2, x_3} and Y = {y_1, y_2, y_3}, and weights shown in the figure 7 below.

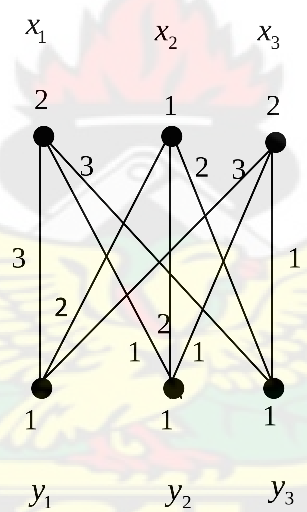


Figure 7: A Feasible Vertex Labeling

The weight of the edge (x_1 y_1) is w(x_1 y_1) = 3, and the weight of the matching $M = \{x_1 y_1, x_2 y_2, x_3 y_3\}$ is given by $w\ (M) = \sum_{e \in M} w(e)$ = 3 + 2 + 1 = 6.

We may define an initial feasible vertex labeling ℓ of N by putting $\ell(x_i)$ equal to the maximum weight of an edge incident to x_i , and $\ell(y_i)$ equal to zero for all $1 \leq i \leq 3$. This gives $\ell(x_1)$ = 3, $\ell(x_2)$ = 2, $\ell(x_3)$ = 3, and $\ell(y_i)$ = 0 for all $1 \leq i \leq 3$. Thus *size (ℓ)* = 3 + 2 + 3 = 8 > *w (M).*

2.5.2.6 ***Equality sub graph***

Let ℓ be a feasible vertex labeling of N. The equality sub graph (with respect to ℓ) in N, $G(\ell) = (V, E_\ell)$ is the spanning sub graph of N containing all edges (x, y) for which

$$\mathrm{E}_\ell = \{(x, y) : \ell(x) + \ell(y) = w(x, y)\}$$

Lemma 2: *Let ℓ be a feasible vertex labeling for N and M be a perfect matching in the equality sub graph G (ℓ). Then w (M) = size (ℓ) and hence M is a maximum weight perfect matching in N and ℓ is a minimum size feasible vertex labeling of N.*

Proof: Let $\mathrm{M} = \{x_1 y_1, x_2 y_2, ..., x_m y_m\}$. Since G ($\ell$) is the equality sub graph of ℓ in N, we have

$$\ell(x_i) + \ell(y_i) = w(x_i y_i), \text{ for all } 1 \le i \le m .$$

Thus

$$w(M) = \sum_{i=1}^{m} w(x_i, y_i) = \sum_{i=1}^{m} ((\ell(x_i) + \ell(y_i)) = \sum_{v \in V(N)} \ell(v) = size(\ell)$$

The facts that M is a maximum weight perfect matching in N and ℓ is a minimum size feasible vertex labeling of N now follows from Lemma 2.

Figure 8 below shows the equality sub graph of the feasible vertex labeling (figure 6) for N.

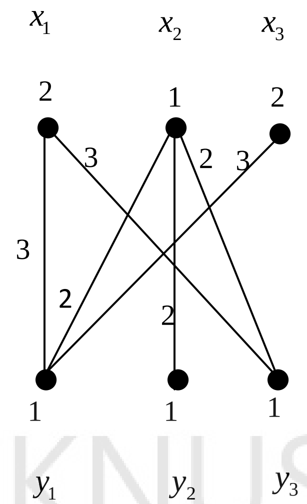


Figure 8: Equality sub graph $G(\ell)$ for N

Theorem 1(Egerváry, 1931): *Let N be a weighted complete bipartite graph. Then the maximum weight of a perfect matching in N is equal to the minimum size of a feasible vertex labeling of N.*

Proof: Let ℓ be a minimum size feasible vertex labeling of N and G = G (ℓ) be the equality sub graph for ℓ in N. By Lemma 2 it suffices to show that G has a perfect matching. We proceed by contradiction.

Suppose that G does not have a perfect matching. There exists a set $S \subseteq X$ such that $|T| < |S|$.

Let α = min {ℓ(x) + ℓ(y) - w(x y): x $\in$ S, y $\notin T$ }

Note that α > 0 since there are no edges in the equality sub graph from S to Y – T and hence we have

$$\ell(x) + \ell(y) > w(x\,y), \text{ for all } x \in S,\ y \notin T$$

We may now define a feasible vertex labeling ℓ' of N as follows:

For each $v \in V(N)$, let

$$\ell'(v) = \begin{cases} \ell(v) - \alpha, \text{ for } v \in S \\ \ell(v) + \alpha, \text{ for } v \in T \\ \ell(v) \qquad \text{otherwise} \end{cases}$$

Suppose $\ell'(v)$ is not a feasible vertex labeling of N.

Then, we have

$$\ell'(x) + \ell'(y) < w(x\ y) \text{ for some } x \in X \text{ and } y \in Y$$

Since ℓ is a feasible vertex labeling of N, we must have $x \in S$, $y \in Y - T$

But then the definition of α implies that

$$\ell(x) + \ell(y) - w(x\ y) \geq \alpha, \text{ and hence}$$

$$\ell`(x) + \ell`(y) - w(x\ y) \geq 0$$

Thus $\ell`$ is a feasible vertex labeling of N.

Since $\alpha > 0$ and $|S| > |T|$ we have

$$Size\ (\ell`) = size\ (\ell) - \alpha(|S| - |T|) < size(\ell)$$

This contradicts the fact that ℓ is a minimum size feasible vertex labeling of N. Thus G has a perfect matching.

2.4.2.8 The *Kuhn-Munkres Algorithm (Hungarian Method)*

Suppose N is a network obtained from *Km:m* by giving each edge *e* an integer weight *w(e).* The algorithm iteratively constructs a sequence of feasible vertex labeling $\ell_1, \ell_2, \ldots$ for N such that *size* (ℓ_{i+1}) < *size* (ℓ_i), and a sequence of matching M_i such that M_i is a maximum matching in the equality sub graph G (ℓ_i), for all $i \geq 1$. It stops when it finds a feasible vertex labeling ℓ_i for which M_i is perfect matching in G (ℓ_i).

Initial Step

(i) Construct a feasible vertex labeling ℓ_1 for N by putting $\ell_1(x) = \max\{w(x\,y): y \in Y\}$ for each $x \in X$, and $\ell_1(y) = 0$ for all $y \in Y$.

(ii) Construct a maximum matching M_1 in G(ℓ_1)

Iterative Step: Suppose we have constructed a feasible vertex labeling ℓ_i of N, and a maximum matching M_i in G = G (ℓ_i), for some $i \geq 1$.

i. If M_i is complete for G (ℓ_i), then M_i is optimal. Stop. Otherwise, there is some unmatched $x \in X$. Set $S = \{x\}$ and $T = \varnothing$.

ii. Let $N_{G(\ell_i)}(S)$ be the neighbour of set S in the equality sub graph G (ℓ_i), where $S \subseteq X$ .If $N_{G(\ell_i)}(S) \neq T$, go to step (iii). Otherwise, $N_{G(\ell_i)}(S) = T$ .Compute $\alpha = \min\{\ell_i(x) + \ell_i(y) - w(xy); x \in S, y \in T^c\}$, where T^c denotes the complement of T in Y and construct a new labeling ℓ_{i+1} by

$$\ell_{i+1}(v) = \begin{cases} \ell_i(v) - \alpha, & \text{if } v \in S \\ \ell_i(v) + \alpha, & \text{if } v \in T \\ \ell_i(v) & \text{otherwise,} \qquad \text{for each } v \in V(N). \end{cases}$$

iii. Choose a vertex y in $N_{G(\ell_i)}(S)$, not in T. If y is matched in M_i, say with $z \in X$, replace S by $S \cup \{z\}$ and T by $T \cup \{y\}$, and go to step (ii). Otherwise, there will be an M – alternating path from x to y, and we may use this path to find a larger matching M_{i+1} in G (ℓ_i). Replace M_i by M_{i+1} and go to step (i).

Example: A mattress company wishes to introduce a new product to its customers. The company has three salespeople and three sales districts to be worked. Based on past sales experience the company can estimate the relative sales productivity ratings for each salesperson S_i in each of the sales districts D_j. The ratings are shown in the table below.

District

Salesperson

	D_1	**D_2**	**D_3**
S_1	**$ 2**	**$ 6**	**$ 3**
S_2	**$ 5**	**$ 8**	**$ 6**
S_3	**$ 4**	**$ 3**	**$ 2**

What is the best way to assign the salespersons for the company to maximize profit?

Solution: The weighted bipartite graph for the assignment is as shown in figure 9(a) below.

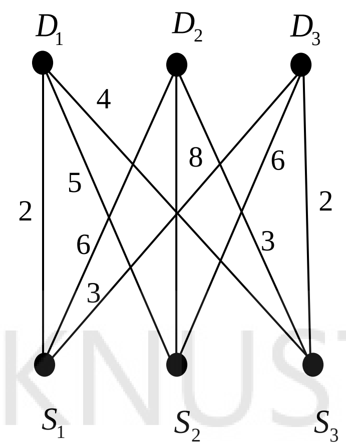


Figure 9(a): Weighted bipartite Graph

First Iteration

(i) We construct a feasible vertex labeling ℓ_1 for N by putting $\ell_1(S_i)$ = max {w (S_i, D_j)} for all $S_i \in X$, and $\ell_1(D_j) = 0$ for all $D_j \in Y$ as shown in figure 9(b).

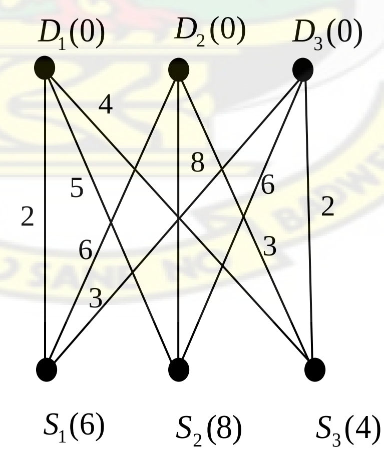


Figure 9(b): Feasible vertex labeling ℓ_1 for N

The equality sub graph G (ℓ_1) for ℓ_1 is shown in figure 9(c).

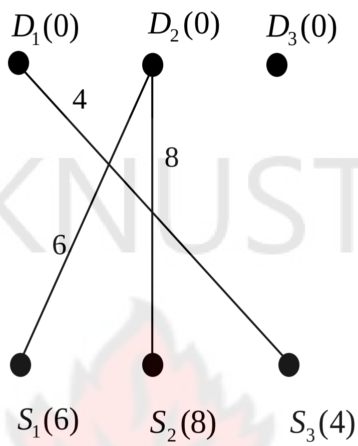


Figure 9(c): Feasible Vertex Labeling ℓ_1 + G(ℓ_1)

(ii) We then choose arbitrary matching $M_1 = \{S_3D_1, S_2D_2\}$ in G (ℓ_1) as shown in figure 9(d).

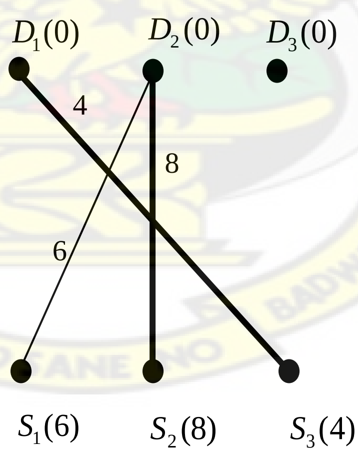


Figure 9(d): Equality sub graph G (ℓ_1) + M_1

$w(M_1) = 4 + 8 = 12$

Iterative Step

Since $|M_1| = 2 < 3$, M_1 is not optimal. We construct a new feasible vertex labeling ℓ_2 for N as follows:

(i) We set $S = \{S_1\}$ and $T = \varnothing$

(ii) Because $N_{G(\ell 1)}(S) = \{D_2\} \neq T$, we go to step (iii)

(iii) We choose a vertex $D_2 \in N_{G(\ell_1)}S - T$ which is in Y. Since D_2 is matched in M_1, we grow tree by adding (D_2, S_2). i.e., $S = \{S_1, S_2\}$ and $T = \{D_2\}$.

At this point, $N_{G(\ell_1)}S = \{D_2\} = T$.

$$\text{We compute } \alpha_{\ell_1} = \min \begin{cases} 6+0-2 \ (S_1,D_1) \\ 6+0-3 \ (S_1,D_3) \\ 8+0-5 \ (S_2,D_1) \\ 8+0-6 \ (S_2,D_3) \end{cases}$$

$= 2$

We then reduce labels of S by 2 and increase labels of T by 2 to obtain the new equality sub graph G (ℓ_2) shown in figure 9(e) below.

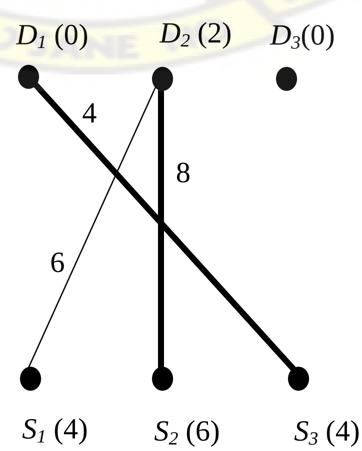


Figure 9(e): Equality sub graph G (ℓ_2) + M_1

Now:

- $S = \{S_1, S_2\}, N_{G(\ell 2)}S = \{D_2, D_3\} \neq T$
- Choose $D_3 \in N_{G(\ell_2)}S - T$ and add it to T.
- D_3 is NOT matched in M_1 so an alternating path $S_1 \to D_2 \to S_2 \to D_3$ with two free ends have been found. We can therefore augment M_1 to get a larger matching M_2 (figure 10(b)) in the new equality graph (figure 10(a)). This matching is perfect, so it must be optimal.

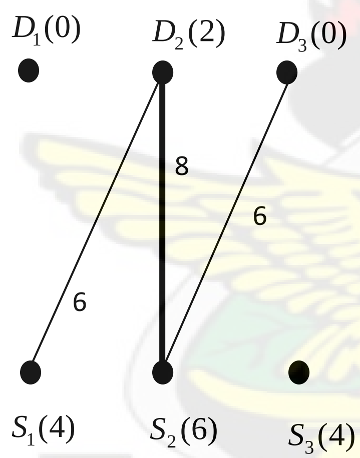


Figure10 (a): New Alternating tree obtained from G (ℓ_2)

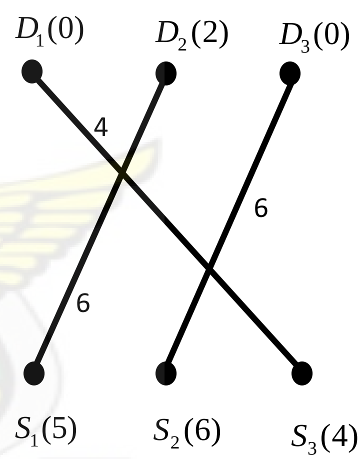


Figure 10(b): New matching M_2

Thus, for the company to maximize profit,

- Salesperson S_1 must be assigned to sales district D_2,
- Salesperson S_2 must be assigned to sales district D_3, and
- Salesperson S_3 must be assigned to sales district D_1.

The optimal value is w (M_2) = 6 + 6 + 4 = 16 > w (M_1) = 12, and it is exactly the labels in the final feasible vertex labeling for M_2 = 4 + 6 + 4 + 2 = 16.

2.4.2.9 ***Correctness of the Method***

- We can always take the trivial labeling ℓ and empty matching $M = \varnothing$ to start the algorithm.
- If the labeling ℓ in the neighborhood of S is equal to T, we saw that we could always update labels to create a new feasible matching ℓ'.
- If the labeling ℓ in the neighborhood of S is not equal to T, we can by definition, always augment the alternating tree by choosing some $x \in S$ and $y \in Y - T$ such that $(x, y) \in E_\ell$. Note that at some point, the y chosen must be free, and in which case we augment M. So, the algorithm always terminates and when it does terminate M is a perfect matching in E_ℓ so by Kuhn-Munkres theorem, it is optimal.

2.5 MATRIX REDUCTION FORM OF THE HUNGARIAN METHOD

One way of looking at the assignment problem and the Hungarian method is in terms of a matrix. Given *n* agents and *n* tasks, and non negative edges $e(i, j), i = 1, 2, ..., n,\ j = 1, 2, ..., n$ represented by the cost c_{ij} of assigning agent i to task j, the problem is to find the cost minimizing assignment.

The method operates directly on the cost table for the problem. More precisely, it converts the original cost table into a series of equivalent cost tables until it reaches one where an optimal solution is obtained.

The steps of the method as outlined by Hiller and Lieberman (2005) are as follows:

1. Subtract the smallest number in each row from every number in the row. Enter the result in a new table.
2. Subtract the smallest number in each column of the new table from every number in the column. Enter the results in another table.
3. Test whether an optimal assignment can be made. We do this by counting the minimum number of lines needed to cover (i.e., cross out) all zeros. If the number of lines equals the number of rows, then an optimal set of assignments is possible. In that case, go to step 6. Otherwise, go to step 4.
4. If the number of lines is less than the number of rows, modify the table in the following way:
   a. Subtract the smallest uncovered number from every uncovered number in the table.
   b. Add the smallest uncovered number in 4(a) to the intersections of covering lines.
   c. Numbers cross out but not at the intersections of cross-out lines carry over unchanged to the next table.
5. Repeat steps 3 and 4 until an optimal set of assignments is possible.
6. Make the assignments one at a time in positions that have zero elements. Begin with rows or columns that have only one zero. Since each row and each column needs to receive exactly one assignment, cross out both the row and the column involved after each assignment is made. Then move on to the rows and columns that are not yet crossed out to select the next assignment, with reference again

given to any such row or column that has only one zero that is not crossed out. Continue until every row and every column has exactly one assignment and so has been crossed out.

Prototype Example

The Better Products Company has decided to initiate the production of four new products, using three plants that currently have excess production capacity. The products require a comparable production effort per unit, so the available production capacity of the plants is measured by the number of units of any product that can be produced per day as given in Table 2.18 below.

Table 2.18: Data for the Better Products Co. problem

		Unit cost ($) for product Product				Capacity Available
		1	2	3	4	
Plant	1	41	27	28	24	75
	2	40	29	-	23	75
	3	37	30	27	21	45
Production rate		20	30	30	40	

The bottom row gives the required production rate per day to meet the rejected sales. Each plant can produce any of these products, except that Plant 2 cannot produce product

3. However, the variable costs per unit of each product differ from plant to plant as shown in the main body of Table 2.18 above.

Management now needs to make a decision on how to prohibit product splitting among the plants and further specifies that every plant should be assigned at least one of the products

Solution

Without product splitting, each product must be assigned to just one plant. Therefore, producing the products can be interpreted as the tasks for an assignment problem, where the plants are the agents.

Management has specified that every plant should be assigned at least one of the products. There are more products (four) than plants (three), so one of the plants will need to be assigned two products. Plant 3 has only enough excess capacity to produce one product (see Table 2.18), so either Plant 1 or Plant 2 will take the extra product.

To make this assignment of an extra product possible within an assignment problem formulation, Plants 1 and 2 each are split into two assignees, as shown in Table 2.19.

Table 2.19: Cost table for the assignment problem formulation for the Better Products Co. problem

		Task (Product)				
		1	2	3	4	5
Assignee (Plant)	1a	820	810	840	960	0
	1b	820	810	840	960	0
	2a	800	870	M	920	0
	2b	800	870	M	920	0
	3	740	900	810	840	M

The reason for M here is that Plant 3 must be assigned a real product (a choice of product 1, 2, 3 or 4), so the Big M method is needed to prevent the assignment of the fictional product to Plant 3.

For an assignment problem the cost c_{ij} is the total cost associated with assignee *i* performing task *j*. For Table 2.19, the total cost (per day), for Plant *i* to produce product *j* is the unit cost of production times the number of units produced (per day). For example, consider the assignment of Plant 1 to product 1. By using the corresponding unit cost in Table 2.18 ($41) and the corresponding demand (number of units produced per day) in Table 2.18 (20), we obtain

Cost of Plant 1producing one unit of product 1 = $41

Required (daily) production of product 1 = 20 units

Total (daily) cost of assigning Plant 1to product 1 = 20×\$41 = \$820, so 820 is entered into Table 2.19 for the cost of either Assignee 1a or 1b performing Task 1.

To illustrate the algorithm, let us consider the table below which is the cost table from Table 2.19.

Table 2.20: Cost table obtained from Table 2.19

	1	2	3	4	5
1a	820	810	840	960	0
1b	820	810	840	960	0
2a	800	870	M	920	0
2b	800	870	M	920	0
3	740	900	810	840	M

Step 1: Subtracting the smallest number in each row from every number in the row results in the following equivalent cost table

	1	2	3	4	5
1a	820	810	840	960	0
1b	820	810	840	960	0
2a	800	870	M	920	0
2b	800	870	M	920	0
3	0	160	70	100	M

Step 2: Subtracting the smallest number in each column of the above equivalent cost table yields the following equivalent cost table.

	1	2	3	4	5
1a	820	650	770	860	0
1b	820	650	770	860	0
2a	800	710	M	820	0
2b	800	710	M	820	0
3	0	0	0	0	M

Step 3: We then test whether an optimal assignment can be made by drawing lines to cover all the zeros.

	1	2	3	4	5
1a	820	650	770	120	0
1b	820	650	770	120	0
2a	800	710	M	80	0
2b	800	710	M	80	0
3	0	0	0	0	M

Since two lines (less than five lines) are required to cover all zeros, it follows that an optimal solution has not been obtained. We go to step 4

Step 4: The minimum element not crossed out is 80 (column 4). We subtract 80 from every uncovered element in the entire table and add 80 to doubly crossed elements to obtain the equivalent cost table below. We then go to step 3 and test for optimality.

	1	2	3	4	5
1a	740	570	690	40	0
1b	740	570	690	40	0
2a	720	630	M	0	0
2b	720	630	M	0	0
3	0	0	0	0	M

Step 3: Test for optimality: We obtain the tableau below.

	1	2	3	4	5
1a	740	570	690	40	0
1b	740	570	690	40	0
2a	720	630	M	0	0
2b	720	630	M	0	0
3	0	0	0	0	M

Since three lines (less than five lines) are required to cover all zeros, it follows that an optimal solution has not been obtained. We then go to step 4.

Step 4: The minimum element not crossed out is 570 (column 2). We subtract 570 from every uncovered element in the entire table and add 570 to doubly crossed elements to obtain the equivalent cost table below.

	1	2	3	4	5
1a	170	0	120	40	0
1b	170	0	120	40	0
2a	150	60	M	0	0
2b	150	60	M	0	0
3	0	0	0	570	M

We then go to step 3.

Step 3: Test for optimality

	1	2	3	4	5
1a	170	0	120	40	0
1b	170	0	120	40	0
2a	150	60	M	0	0
2b	150	60	M	0	0
3	0	0	0	570	M

Since four lines (less than five lines) are required to cover all zeros, it follows that an optimal solution has not been obtained. We then go to step 4.

Step 4: The minimum element not crossed out is 120 (column 3). We subtract 120 from every uncovered element in the entire table and add 120 to doubly crossed elements to obtain the equivalent cost table below.

	1	2	3	4	5
1a	50	0	0	40	0
1b	50	0	0	40	0
2a	30	60	M	0	0
2b	30	60	M	0	0
3	0	120	0	690	M

We go to step 3.

Step 3: Test for optimality

	1	2	3	4	5
1a	50	0	0	40	0
1b	50	0	0	40	0
2a	30	60	M	0	0
2b	30	60	M	0	0
3	0	120	0	690	M

Since all the zeros are covered by lines (five in number), it follow that the zeros provide an optimal solution to the problem. The zeros are as shown below:

	1	2	3	4	5
1a		$\boxed{0}$	0		0
1b		0	$\boxed{0}$		0
2a				$\boxed{0}$	0
2b				0	$\boxed{0}$
3	$\boxed{0}$		0		

This table has several ways of making a complete set of assignments to zero element positions (several optimal solutions), including the one shown by the five boxes.

The resulting total cost is seen in Table 2.19 to be

$$Z = 740 + 810 + 840 + 920 + 0 = 3,310$$

2.6 THE MUNKRES ASSIGNMENT ALGORITHM (MODIFIED HUNGARIAN METHOD)

The Hungarian Method was later revised by James Munkres in 1957 and has since been known as the Munkres assignment algorithm or the Kuhn-Munkres algorithm. His contribution to Kuhn's algorithm was that he introduced the procedure for finding

(i) a minimal set of lines which contain all zeros

(ii) a maximal set of independent zeros,

(iii) "starred zeros" and 'primed zeros" and

(iv) alternating sequence between "starred zeros" and "primed zeros"

By default, the algorithm performs a minimization on the elements in the cost matrix The modified Hungarian method on the cost matrix of an assignment problem involves the following steps:

1. Subtract the row minimum from each row.
2. Find a zero (Z) in the resulting matrix. If there are no starred zeros in its column or row, star the zero. Repeat for each zero.
3. Cover each column that has a starred zero.
   - (i) If all the columns are covered, then the assignment is optimal.
   - (ii) Otherwise, go to Step 4.
4. (a) (i) Find a non-covered zero and prime it.
   - (iii) If there is no starred zero in the row containing this primed zero, go to Step 5.
   - (iv) Otherwise, cover this row and uncover the column containing the starred zero.
   - (v) Continue in this manner until there are no uncovered zeros left.

   (b) Save the smallest uncovered value in the cost matrix and go to Step 6.
5. Construct a series of alternating primed and starred zeros as follows. Let Z_0 represent the uncovered zero found in Step 4. Let Z_1 denote the starred zero in the column of Z_0 (if any). Let Z_2 denote the primed zero in the row of Z_1 (there will always be one). Continue until the series terminates at a primed zero that has no starred zero in its column. Un-star each starred zero of the series, prime each starred zero of the series, erase all primes and uncover every line in the matrix. Return to Step 3.

6. Add the value found in Step 4 to every element of each covered row, and subtract it from every element of each uncovered column. Return to Step 4(b) without altering any stars, primes, or covered lines. Return to Step 4 without altering any stars, primes or covered lines.

Prototype Example

A building firm possesses three cranes each of which has a distance (km) from three different construction sites as shown in the table below:

Construction Site Number

Crane Number		*p*	*q*	*r*
	a	1	2	3
	b	2	4	6
	c	3	6	9

Place the cranes (one for each construction sites) in such a way that the overall distance required for the transfer of Cranes to Sites is as small as possible.

Solution: The cost matrix is as shown below.

1	2	3
2	4	6
3	6	9

Step1: The smallest elements of rows 1, 2 and 3 are respectively 1, 2 and 3. We subtract each of these numbers from every element in their respective rows and obtain the matrix below.

0	1	2
0	2	4
0	3	6

Go to step 2.

Step 2: We then find a zero (Z) in the resulting matrix. If there is no starred zero in its row or column, then we star that zero. We repeat for each zero in the matrix.

0*	1	2
0	2	4
0	3	6

Go to step 3

Step 3: We cover the column containing the starred zero (i.e., column one).

0*	1	2
0	2	4
0	3	6

Since the number of columns covered (one) is less than the total number of columns (three), we go to Step 4.

Step 4: Since all the zeros of the matrix in Step 3 are covered, we save the smallest uncovered value (i.e., a (1, 2) = 1) and then go to Step 6.

0*	1	2
0	2	4
0	3	6

Step 6: Since none of the rows of the matrix is covered, we subtract the smallest uncovered value found in Step 4 (i.e., 1) from every element of each uncovered column and return to Step 4 without altering any stars, primes, or covered lines.

0*	0	1
0	1	3
0	2	5

Step 4: (i) The zero in a (1, 2) is not covered so we prime it.

0*	0^1	1
0	1	3
0	2	5

(ii). There is a starred zero in the row containing the primed zero. We therefore cover row 1, uncover column 1 and prime the zero just below the starred zero.

0*	0^1	1
0^1	1	3
0	2	5

We then go to step 5.

Step 5: We construct a path of alternating primed and starred zeros as follows. Let Z_0 represent the uncovered primed zero found in Step 4 above (i.e., entry (2,1). Let Z_1 denote the starred zero in the column of Z_0 (i.e., entry (1, 1). Let Z_2 denote the primed zero in the row of Z_1 (there will always be one) (i.e., entry (1, 2). Continue until the series terminates at a primed zero that has no starred zero in its column. We then un-star starred zero Z_1 of the series, star each primed zero (Z_0, Z_2) of the series, erase all primes and uncover every line in the matrix (row 1), and return to Step 3.

0	0*	1
0*	1	3
0	2	5

Step 3: We cover each column that has a starred zero found in step 5 above.

0	0*	1
0*	1	3
0	2	5

Since only two columns are covered with starred zeros we go to step 4.

Step 4 We store the minimum uncovered value (i.e. 1) and go to step 6.

0	0*	1
0*	1	3
0	2	5

Step 6: We subtract the minimum uncovered value found in step 4 (i.e, 1) from every element of each uncovered column.

0	0*	0
0*	1	2
0	2	4

We go to step 4.

Step 4: (i) The zero in a (1, 3) is not covered so we prime it.

0	0*	0^1
0*	1	2
0	2	4

(ii). There is a starred zero in the row containing the primed zero (i.e., column 2) We therefore cover row 1 and uncover column 2.

0	0*	0^1
0*	1	2
0	2	4

Since all the zeros are covered, we save the minimum uncovered number (i.e., 1) and go to step 6

Step 6: We add the minimum uncovered number (i.e., 1) to the zero at entry (1,1) because it is doubly crossed, and subtract it from all the elements that are not covered.

1	0*	0^1
0*	0	1
0	1	3

We then go to step 4.

Step 4: (i) We prime the non-covered zero at entry (2, 2).

1	0*	0^1
0*	0^1	1
0	1	3

(ii) We cover row 2, uncover column 1and prime the zero at entry (3, 1).

1	0*	0^1
0*	0^1	1
0^1	1	3

We go to step 5.

Step 5

1	0	0*
0	0*	1
0*	1	3

We go to step 3.

Step 3

1	0	0*
0	0*	1
0*	1	3

All three columns are covered.

The assignment pairs are indicated by the positions of the starred zeros in the cost matrix below.

1	0	0*
0	0*	1
0*	1	3

This means that Crane ***a*** must be placed at Construction site ***r*,** Crane ***b*** at Construction site ***q*** and Crane ***c*** at Construction site ***p***

The cost associated is 3 + 4 + 3 = 10.

CHAPTER 3

DATA COLLECTION, ANALYSIS AND DISCUSSION

3.1 DATA COLLECTION

Data was collected from the factory site of Latex Foam Company Limited, Kumasi. The company operates eight models of vehicles. The list below gives the types of vehicles;

i. KIA truck (K)

ii. TATA truck (T)

iii. Renault (articulator) truck (RA)

iv. TATA (articulator) truck (TA)

v. Benz (articulator) truck (BA)

vi. DAF (articulator) truck (DA)

vii. DAF (cargo) truck (DC)

viii. Benz (cargo) truck (BC)

There are three KIA trucks (K1, K2, and K3), four TATA trucks (T1, T2, T3, and T4) and two Benz articulator trucks (BA1, BA2). The rest are single vehicles.

The vehicles ply routes along which they serve various customers with the final destinations mostly being District Capitals. The list of these final destinations is given below;

Sefwi Juaboso (D1), Asankraguaa (D2),Yendi (D3), Assin Fosu (D4), Kintampo (D5), Kwame Danso (D6), Bogoso (D7), Osei Kojokrom (D8), Bawku (D9), Drobo (D10), Goaso (D11), Yeji (D12), Lawra (D13), Juaso/Obogu (D14), Nkawkaw (D15), Gushiegu (D16).

The cost of a trip from the factory shed in Kumasi to a destination is measured in gallons of diesel used. Table 3.1 shows the cost of a trip when the vehicles are assigned to the various destinations.

<u>Table 3.0: Types of Vehicle and quantity of diesel (in gallons) used per trip</u>

Type of vehicle	Sefwi Juabeso	Asankraguaa	Yendi	Assin Fosu	Kintampo	Kwame Danso	Bogoso	Osei Kojokrom	Bawku	Drobo	Goaso	Yeji	Lawra	Obogu	Nkawkaw	Gushiegu
K	28	22	40	12	16	25	20	35	-	16	12	20	-	9	10	-
T	29	23	43	13	18	24	22	37	-	18	13	21	-	10	11	-
RA	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
TA	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
BA	54	49	83	28	34	53	48	57	108	34	28	43	113	24	18	96
DA	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
DC	40	42	73	20	28	30	38	55	83	25	20	30	75	16	17	76
BC	39	41	72	19	27	28	37	53	81	23	19	29	73	15	16	74

The KIA and TATA trucks are restricted from going far places and therefore data on the fuel consumption for these trucks was not obtained for places like Gushiegu, Lawra and Bawku.

The M found in Table 3.2 shows that the vehicles involved are prohibited from going to those places

The cost (c_{ij}) of Table 3.2 below was obtained from Table 3.1 for all the fourteen trucks. C_{ij} represent the cost of assigning vehicle $i \in V$ to rout $j \in D$ where V1 = K1,V2 = K2,V3 = K3,V4 = T1,V5 = T2 V6 = T3,V7 = T4,V8 = RA,V9 = TA,V10 = BA1,V11= BA2,V12 = DA,V13 = DC and V14 = BC.

Table 3.1: Cost matrix obtained from Table 3.0

	D1	**D2**	**D3**	**D4**	**D5**	**D6**	**D7**	**D8**	**D9**	**D10**	**D11**	**D12**	**D13**	**D14**	**D15**	**D16**
V1	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V2	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V3	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V4	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V5	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V6	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V7	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V8	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V9	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V10	54	49	83	28	34	53	48	57	108	34	28	43	113	24	18	96
V11	54	49	83	28	34	53	48	57	108	34	28	43	113	24	18	96
V12	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V13	40	42	73	20	28	30	38	55	83	25	20	30	75	16	17	76
V14	39	41	72	19	27	28	37	53	81	23	19	29	73	15	16	74

3.2 PROBLEM FORMULATION

The problem is to find the minimum total cost in assigning each vehicle to a distinct destination. The problem is formulated as an assignment problem with the assumption that a vehicle is assigned to only one route on which there may be more than depot.

The mathematical notation and formulation are as follows.

Let

$C_{ij} =$ cost coefficients(number of gallons of diesel) of assigning vehicle type i from factory to route j.
$V =$ Set of all vehicles.
$D =$ Set of all destinations

m = Total number of vehicles

n = number of routes to the final destinations

The Boolean variables, X_{ij} , representing assignment realization are defined by

$$X_{ij} = \begin{cases} 1 \text{ If vehicle type } i \text{ is assigned from factory to route } j \\ 0 \text{ otherwise} \end{cases}$$

The objective function (Z) can be written as

$$\text{Minimize } Z=\sum_{i\in v}^{m}\sum_{j\in u}^{n} c_{ij}x_{ij} \text{....................(1)}$$

Subject to

$$\sum_{j}^{n} x_{ij} = 1, \text{ for all } i \in V \text{....................(2)}$$

$$\sum_{i\in v}^{m} x_{ij} = 1, \text{ for all } j \in U \text{....................(3)}$$

$$x_{ij} = 0 \text{ or } x_{ij} = 1 \text{............(4)}$$

The objective function (1) is to minimize the total cost in terms of number of gallons of diesel used for the assignments.

Constraint (2) requires that each vehicle is assigned exactly one route to a destination.

Constraint (3) requires that every route to a destination is assigned to only one vehicle.

Constraint (4) requires that a particular vehicle i is assigned to a distinct destination j . (i.e. $x_{ij} = 1$) or otherwise ($x_{ij} = 0$)

For efficient assignments of these trucks, the cost matrix of Table 3.2 must be a square one. In order to obtain a square cost matrix, two vehicles from the six brands (in terms of fuel consumption), that is KIA truck, TATA truck, Renault or TATA or DAF (articulator) truck, Benz (articulator) truck, DAF (cargo) truck and Benz (cargo) truck, were selected and added in turn to the existing fourteen vehicles to obtain a 16×16 matrix. In all, twenty-one cost matrices were obtained. Tables 3.3 - 3.21 show these matrices.

Table 3.2: Cost matrix for adding two KIA trucks

	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	D16
V1	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V2	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V3	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V4	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V5	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V6	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V7	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V8	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V9	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V10	54	49	83	28	34	53	48	57	108	34	28	43	113	24	18	96
V11	54	49	83	28	34	53	48	57	108	34	28	43	113	24	18	96
V12	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V13	40	42	73	20	28	30	38	55	83	25	20	30	75	16	17	76
V14	39	41	72	19	27	28	37	53	81	23	19	29	73	15	16	74
V1	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V1	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M

Table 3.3: Cost matrix for adding a KIA truck and a TATA truck

	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	D16
V1	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V2	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V3	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V4	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V5	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V6	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V7	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V8	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V9	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V10	54	49	83	28	34	53	48	57	108	34	28	43	113	24	18	96
V11	54	49	83	28	34	53	48	57	108	34	28	43	113	24	18	96
V12	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V13	40	42	73	20	28	30	38	55	83	25	20	30	75	16	17	76
V14	39	41	72	19	27	28	37	53	81	23	19	29	73	15	16	74
V1	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V4	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M

Table 3.4: Cost matrix for adding a KIA truck and a Renault, TATA or DAT (articulator) truck

	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	D16
V1	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V2	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V3	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V4	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V5	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V6	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V7	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V8	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V9	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V10	54	49	83	28	34	53	48	57	108	34	28	43	113	24	18	96
V11	54	49	83	28	34	53	48	57	108	34	28	43	113	24	18	96
V12	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V13	40	42	73	20	28	30	38	55	83	25	20	30	75	16	17	76
V14	39	41	72	19	27	28	37	53	81	23	19	29	73	15	16	74
V1	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V8	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98

Table 3.5: Cost matrix for adding a KIA truck and a Benz (articulator) truck

	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	D16
V1	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V2	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V3	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V4	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V5	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V6	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V7	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V8	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V9	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V10	54	49	83	28	34	53	48	57	108	34	28	43	113	24	18	96
V11	54	49	83	28	34	53	48	57	108	34	28	43	113	24	18	96
V12	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V13	40	42	73	20	28	30	38	55	83	25	20	30	75	16	17	76
V14	39	41	72	19	27	28	37	53	81	23	19	29	73	15	16	74
V1	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V10	54	49	83	28	34	53	48	57	108	34	28	43	113	24	18	96

Table 3.6 Cost matrix for adding a KIA truck and a DAF (cargo) truck

	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	D16
V1	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V2	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V3	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V4	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V5	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V6	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V7	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V8	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V9	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V10	54	49	83	28	34	53	48	57	108	34	28	43	113	24	18	96
V11	54	49	83	28	34	53	48	57	108	34	28	43	113	24	18	96
V12	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V13	40	42	73	20	28	30	38	55	83	25	20	30	75	16	17	76
V14	39	41	72	19	27	28	37	53	81	23	19	29	73	15	16	74
V1	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V13	40	42	73	20	28	30	38	55	83	25	20	30	75	16	17	76

Table 3.7: Cost matrix for adding a KIA truck and a Benz (cargo) truck

	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	D16
V1	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V2	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V3	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V4	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V5	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V6	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V7	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V8	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V9	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V10	54	49	83	28	34	53	48	57	108	34	28	43	113	24	18	96
V11	54	49	83	28	34	53	48	57	108	34	28	43	113	24	18	96
V12	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V13	40	42	73	20	28	30	38	55	83	25	20	30	75	16	17	76
V14	39	41	72	19	27	28	37	53	81	23	19	29	73	15	16	74
V1	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V14	39	41	72	19	27	28	37	53	81	23	19	29	73	15	16	74

Table 3.8: Cost matrix for adding two TATA trucks

	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	D16
V1	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V2	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V3	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V4	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V5	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V6	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V7	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V8	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V9	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V10	54	49	83	28	34	53	48	57	108	34	28	43	113	24	18	96
V11	54	49	83	28	34	53	48	57	108	34	28	43	113	24	18	96
V12	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V13	40	42	73	20	28	30	38	55	83	25	20	30	75	16	17	76
V14	39	41	72	19	27	28	37	53	81	23	19	29	73	15	16	74
V4	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V5	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M

Table 3.9: Cost matrix for adding a TATA truck and a Renault, TATA or DAF (arti.) truck

	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	D16
V1	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V2	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V3	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V4	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V5	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V6	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V7	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V8	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V9	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V10	54	49	83	28	34	53	48	57	108	34	28	43	113	24	18	96
V11	54	49	83	28	34	53	48	57	108	34	28	43	113	24	18	96
V12	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V13	40	42	73	20	28	30	38	55	83	25	20	30	75	16	17	76
V14	39	41	72	19	27	28	37	53	81	23	19	29	73	15	16	74
V4	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V8	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98

Table 3.10: Cost matrix for adding a TATA truck and a Benz (arti.) truck

	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	D16
V1	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V2	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V3	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V4	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V5	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V6	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V7	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V8	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V9	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V10	54	49	83	28	34	53	48	57	108	34	28	43	113	24	18	96
V11	54	49	83	28	34	53	48	57	108	34	28	43	113	24	18	96
V12	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V13	40	42	73	20	28	30	38	55	83	25	20	30	75	16	17	76
V14	39	41	72	19	27	28	37	53	81	23	19	29	73	15	16	74
V4	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V10	54	49	83	28	34	53	48	57	108	34	28	43	113	24	18	96

Table 3.11: Cost matrix for adding a TATA and a DAF (cargo) truck

	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	D16
V1	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V2	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V3	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V4	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V5	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V6	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V7	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V8	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V9	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V10	54	49	83	28	34	53	48	57	108	34	28	43	113	24	18	96
V11	54	49	83	28	34	53	48	57	108	34	28	43	113	24	18	96
V12	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V13	40	42	73	20	28	30	38	55	83	25	20	30	75	16	17	76
V14	39	41	72	19	27	28	37	53	81	23	19	29	73	15	16	74
V1	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V13	40	42	73	20	28	30	38	55	83	25	20	30	75	16	17	76

Table 3.12: Cost matrix for adding a TATA and a Benz (cargo) truck

	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	D16
V1	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V2	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V3	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V4	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V5	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V6	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V7	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V8	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V9	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V10	54	49	83	28	34	53	48	57	108	34	28	43	113	24	18	96
V11	54	49	83	28	34	53	48	57	108	34	28	43	113	24	18	96
V12	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V13	40	42	73	20	28	30	38	55	83	25	20	30	75	16	17	76
V14	39	41	72	19	27	28	37	53	81	23	19	29	73	15	16	74
V4	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V13	39	41	72	19	27	28	37	53	81	23	19	29	73	15	16	74

Table 3.13: Cost matrix for adding two of Renault, TATA or DAF (articulator) trucks

	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	D16
V1	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V2	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V3	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V4	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V5	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V6	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V7	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V8	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V9	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V10	54	49	83	28	34	53	48	57	108	34	28	43	113	24	18	96
V11	54	49	83	28	34	53	48	57	108	34	28	43	113	24	18	96
V12	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V13	40	42	73	20	28	30	38	55	83	25	20	30	75	16	17	76
V14	39	41	72	19	27	28	37	53	81	23	19	29	73	15	16	74
V8	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V9	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98

Table 3.14: Cost matrix for adding Renault, TATA or DAT (arti.) truck and a Benz (arti.) truck

	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	D16
V1	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V2	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V3	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V4	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V5	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V6	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V7	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V8	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V9	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V10	54	49	83	28	34	53	48	57	108	34	28	43	113	24	18	96
V11	54	49	83	28	34	53	48	57	108	34	28	43	113	24	18	96
V12	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V13	40	42	73	20	28	30	38	55	83	25	20	30	75	16	17	76
V14	39	41	72	19	27	28	37	53	81	23	19	29	73	15	16	74
V8	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V10	54	49	83	28	34	53	48	57	108	34	28	43	113	24	18	96

Table 3.15: Cost matrix for adding Renault, TATA or DAT (arti.) truck and a DAF (cargo) truck

	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	D16
V1	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V2	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V3	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V4	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V5	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V6	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V7	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V8	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V9	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V10	54	49	83	28	34	53	48	57	108	34	28	43	113	24	18	96
V11	54	49	83	28	34	53	48	57	108	34	28	43	113	24	18	96
V12	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V13	40	42	73	20	28	30	38	55	83	25	20	30	75	16	17	76
V14	39	41	72	19	27	28	37	53	81	23	19	29	73	15	16	74
V8	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V13	40	42	73	20	28	28	38	55	83	25	20	30	75	16	17	76

Table 3.16: Cost matrix for adding Renault, TATA or DAT (arti.) truck and a Benz (cargo) truck

	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	D16
V1	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V2	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V3	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V4	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V5	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V6	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V7	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V8	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V9	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V10	54	49	83	28	34	53	48	57	108	34	28	43	113	24	18	96
V11	54	49	83	28	34	53	48	57	108	34	28	43	113	24	18	96
V12	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V13	40	42	73	20	28	30	38	55	83	25	20	30	75	16	17	76
V14	39	41	72	19	27	28	37	53	81	23	19	29	73	15	16	74
V8	55	50	85	29	35	55	50	60	110	35	22	45	115	25	19	98
V14	39	41	72	19	27	28	37	53	81	23	19	29	73	15	16	74

Table 3.17: Cost matrix for adding two Benz (arti.) trucks

	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	D16
V1	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V2	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V3	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V4	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V5	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V6	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V7	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V8	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V9	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V10	54	49	83	28	34	53	48	57	108	34	28	43	113	24	18	96
V11	54	49	83	28	34	53	48	57	108	34	28	43	113	24	18	96
V12	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V13	40	42	73	20	28	30	38	55	83	25	20	30	75	16	17	76
V14	39	41	72	19	27	28	37	53	81	23	19	29	73	15	16	74
V10	54	49	83	28	34	28	48	57	108	34	28	43	113	24	18	96
V11	54	49	83	28	34	28	48	57	108	34	28	43	113	24	18	96

Table 3.18: Cost matrix for adding a Benz (arti.) truck and a DAF (cargo) truck

	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	D16
V1	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V2	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V3	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V4	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V5	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V6	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V7	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V8	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V9	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V10	54	49	83	28	34	53	48	57	108	34	28	43	113	24	18	96
V11	54	49	83	28	34	53	48	57	108	34	28	43	113	24	18	96
V12	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V13	40	42	73	20	28	30	38	55	83	25	20	30	75	16	17	76
V14	39	41	72	19	27	28	37	53	81	23	19	29	73	15	16	74
V10	54	49	83	28	34	53	48	57	108	34	28	43	113	24	18	96
V13	40	42	73	20	28	30	38	55	83	25	20	30	75	16	17	76

Table 3.19: Cost matrix for adding a Benz (arti.) truck and a Benz (cargo) truck

	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	D16
V1	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V2	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V3	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V4	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V5	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V6	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V7	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V8	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V9	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V10	54	49	83	28	34	53	48	57	108	34	28	43	113	24	18	96
V11	54	49	83	28	34	53	48	57	108	34	28	43	113	24	18	96
V12	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V13	40	42	73	20	28	30	38	55	83	25	20	30	75	16	17	76
V14	39	41	72	19	27	28	37	53	81	23	19	29	73	15	16	74
V10	54	49	83	28	34	53	48	57	108	34	28	43	113	24	18	96
V14	39	41	72	19	27	28	37	53	81	23	19	29	73	15	16	74

Table 3.20: Cost matrix for adding two DAF (cargo) trucks

	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	D16
V1	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V2	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V3	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V4	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V5	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V6	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V7	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V8	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V9	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V10	54	49	83	28	34	53	48	57	108	34	28	43	113	24	18	96
V11	54	49	83	28	34	53	48	57	108	34	28	43	113	24	18	96
V12	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V13	40	42	73	20	28	30	38	55	83	25	20	30	75	16	17	76
V14	39	41	72	19	27	28	37	53	81	23	19	29	73	15	16	74
V13	40	42	73	20	28	30	38	55	83	25	20	30	75	16	17	76
V13	40	42	73	20	28	30	38	55	83	25	20	30	75	16	17	76

Table 3.21: Cost matrix for adding a DAF (cargo) and a Benz (cargo) trucks

	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	D16
V1	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V2	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V3	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V4	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V5	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V6	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V7	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V8	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V9	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V10	54	49	83	28	34	53	48	57	108	34	28	43	113	24	18	96
V11	54	49	83	28	34	53	48	57	108	34	28	43	113	24	18	96
V12	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V13	40	42	73	20	28	30	38	55	83	25	20	30	75	16	17	76
V14	39	41	72	19	27	28	37	53	81	23	19	29	73	15	16	74
V13	40	42	73	20	28	30	38	55	83	25	20	30	75	16	17	76
V14	39	41	72	19	27	28	37	53	81	23	19	29	73	15	16	74

Table 3.22: Cost matrix for adding two Benz (cargo) trucks

	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	D16
V1	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V2	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V3	28	22	40	12	16	25	20	35	M	16	12	20	M	9	10	M
V4	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V5	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V6	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V7	29	23	43	13	18	26	22	37	M	18	13	21	M	10	11	M
V8	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V9	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V10	54	49	83	28	34	53	48	57	108	34	28	43	113	24	18	96
V11	54	49	83	28	34	53	48	57	108	34	28	43	113	24	18	96
V12	55	50	85	29	35	55	50	60	110	35	29	45	115	25	19	98
V13	40	42	73	20	28	30	38	55	83	25	20	30	75	16	17	76
V14	39	41	72	19	27	28	37	53	81	23	19	29	73	15	16	74
V14	39	41	72	19	27	28	37	53	81	23	19	29	73	15	16	74
V14	39	41	72	19	27	28	37	53	81	23	19	29	73	15	16	74

3.3 RESULTS AND DISCUSSION

A Mat Lab code, written by Buehren (2008), was used to implement the Munkres algorithm on a Pentium (IV) computer of processor speed 2.60GHz using the data of Tables 3.3 – 3.23. The output of the program for each of the twenty-one cost matrices is given in Table 3.23 below.

Table 3.23(a): Results obtained from Cost Matrices using Mat lab code

Table Number	**Assignments($x_{ij} = 1$ values)**	***Cost Z*** $= \sum_{i=1}^{16}\sum_{j=1}^{16} c_{ij}x_{ij}$
3.3	$x_{1,10}=1, x_{2,3}=1, x_{3,8}=1, x_{4,2}=1, x_{5,12}=1, x_{6,4}=1, x_{7,1}=1, x_{8,14}=1$ $x_{9,6}=1, x_{10,11}=1, x_{11,16}=1, x_{12,15}=1, x_{13,13}=1, x_{14,9}=1, x_{15,5}=1, x_{16,7}=1$	572
3.4	$x_{1,10}=1, x_{2,3}=1, x_{3,8}=1, x_{4,7}=1, x_{5,1}=1, x_{6,12}=1, x_{7,4}=1, x_{8,15}=1$ $x_{9,11}=1, x_{10,16}=1, x_{11,6}=1, x_{12,14}=1, x_{13,9}=1, x_{14,13}=1, x_{15,5}=1, x_{16,2}=1$	574
3.5	$x_{1,8}=1, x_{2,3}=1, x_{3,10}=1, x_{4,7}=1, x_{5,2}=1, x_{6,12}=1, x_{7,1}=1, x_{8,4}=1$ $x_{9,14}=1, x_{10,15}=1, x_{11,16}=1, x_{12,11}=1, x_{13,9}=1, x_{14,13}=1, x_{15,5}=1, x_{16,6}=1$	590
3.6	$x_{1,8}=1, x_{2,3}=1, x_{3,10}=1, x_{4,7}=1, x_{5,2}=1, x_{6,12}=1, x_{7,1}=1, x_{8,4}=1$ $x_{9,14}=1, x_{10,15}=1, x_{11,6}=1, x_{12,11}=1, x_{13,9}=1, x_{14,13}=1, x_{15,5}=1, x_{16,16}=1$	589
3.7	$x_{1,8}=1, x_{2,3}=1, x_{3,10}=1, x_{4,7}=1, x_{5,2}=1, x_{6,12}=1, x_{7,1}=1, x_{8,11}=1$ $x_{9,4}=1, x_{10,14}=1, x_{11,15}=1, x_{12,6}=1, x_{13,13}=1, x_{14,9}=1, x_{15,5}=1, x_{16,16}=1$	569
3.8	$x_{1,10}=1, x_{2,8}=1, x_{3,7}=1, x_{4,5}=1, x_{5,2}=1, x_{6,12}=1, x_{7,1}=1, x_{8,15}=1$ $x_{9,6}=1, x_{10,4}=1, x_{11,11}=1, x_{12,14}=1, x_{13,9}=1, x_{14,16}=1, x_{15,3}=1, x_{16,13}=1$	567
3.9	$x_{1,8}=1, x_{2,3}=1, x_{3,5}=1, x_{4,2}=1, x_{5,1}=1, x_{6,10}=1, x_{7,4}=1, x_{8,11}=1$ $x_{9,6}=1, x_{10,14}=1, x_{11,16}=1, x_{12,15}=1, x_{13,9}=1, x_{14,13}=1, x_{15,7}=1, x_{16,12}=1$	576
3.10	$x_{1,10}=1, x_{2,3}=1, x_{3,5}=1, x_{4,12}=1, x_{5,7}=1, x_{6,8}=1, x_{7,1}=1, x_{8,11}=1$ $x_{9,4}=1, x_{10,14}=1, x_{11,16}=1, x_{12,15}=1, x_{13,9}=1, x_{14,13}=1, x_{15,2}=1, x_{16,6}=1$	592
3.11	$x_{1,10}=1, x_{2,3}=1, x_{3,5}=1, x_{4,12}=1, x_{5,7}=1, x_{6,8}=1, x_{7,1}=1, x_{8,11}=1$ $x_{9,4}=1, x_{10,14}=1, x_{11,15}=1, x_{12,6}=1, x_{13,9}=1, x_{14,13}=1, x_{15,2}=1, x_{16,16}=1$	591

Continuation of Table 3.23:

Table Number	**Assignments($x_{ij} = 1$ values)**	*Cost Z* $= \sum_{i=1}^{16}\sum_{j=1}^{16} c_{ij}x_{ij}$
3.12	$x_{1,10}=1, x_{2,3}=1, x_{3,5}=1, x_{4,12}=1, x_{5,7}=1, x_{6,8}=1, x_{7,1}=1, x_{8,11}=1$ $x_{9,4}=1, x_{10,14}=1, x_{11,15}=1, x_{12,6}=1, x_{13,13}=1, x_{14,9}=1, x_{15,2}=1, x_{16,16}=1$	571
3.13	$x_{1,10}=1, x_{2,3}=1, x_{3,5}=1, x_{4,12}=1, x_{5,7}=1, x_{6,8}=1, x_{7,1}=1, x_{8,11}=1$ $x_{9,4}=1, x_{10,14}=1, x_{11,15}=1, x_{12,6}=1, x_{13,9}=1, x_{14,16}=1, x_{15,2}=1, x_{16,13}=1$	569
3.14	$x_{1,8}=1, x_{2,3}=1, x_{3,10}=1, x_{4,2}=1, x_{5,7}=1, x_{6,12}=1, x_{7,1}=1, x_{8,5}=1$ $x_{9,4}=1, x_{10,15}=1, x_{11,16}=1, x_{12,11}=1, x_{13,9}=1, x_{14,13}=1, x_{15,14}=1, x_{16,6}=1$	609
3.15	$x_{1,8}=1, x_{2,3}=1, x_{3,10}=1, x_{4,2}=1, x_{5,7}=1, x_{6,12}=1, x_{7,1}=1, x_{8,5}=1$ $x_{9,6}=1, x_{10,15}=1, x_{11,4}=1, x_{12,11}=1, x_{13,9}=1, x_{14,13}=1, x_{15,14}=1, x_{16,16}=1$	608
3.16	$x_{1,10}=1, x_{2,3}=1, x_{3,8}=1, x_{4,7}=1, x_{5,2}=1, x_{6,12}=1, x_{7,1}=1, x_{8,5}=1$ $x_{9,4}=1, x_{10,14}=1, x_{11,15}=1, x_{12,6}=1, x_{13,16}=1, x_{14,9}=1, x_{15,11}=1, x_{16,13}=1$	588
3.17	$x_{1,10}=1, x_{2,3}=1, x_{3,8}=1, x_{4,7}=1, x_{5,2}=1, x_{6,12}=1, x_{7,1}=1, x_{8,5}=1$ $x_{9,4}=1, x_{10,14}=1, x_{11,6}=1, x_{12,11}=1, x_{13,9}=1, x_{14,16}=1, x_{15,15}=1, x_{16,13}=1$	586
3.18	$x_{1,8}=1, x_{2,3}=1, x_{3,10}=1, x_{4,2}=1, x_{5,7}=1, x_{6,12}=1, x_{7,1}=1, x_{8,5}=1$ $x_{9,6}=1, x_{10,15}=1, x_{11,16}=1, x_{12,11}=1, x_{13,9}=1, x_{14,13}=1, x_{15,14}=1, x_{16,4}=1$	607
3.19	$x_{1,10}=1, x_{2,3}=1, x_{3,8}=1, x_{4,7}=1, x_{5,2}=1, x_{6,12}=1, x_{7,1}=1, x_{8,5}=1$ $x_{9,4}=1, x_{10,14}=1, x_{11,15}=1, x_{12,6}=1, x_{13,16}=1, x_{14,9}=1, x_{15,11}=1, x_{16,13}=1$	587
3.20	$x_{1,10}=1, x_{2,3}=1, x_{3,8}=1, x_{4,7}=1, x_{5,2}=1, x_{6,12}=1, x_{7,1}=1, x_{8,5}=1$ $x_{9,4}=1, x_{10,14}=1, x_{11,6}=1, x_{12,11}=1, x_{13,9}=1, x_{14,16}=1, x_{15,15}=1, x_{16,13}=1$	584
3.21	$x_{1,7}=1, x_{2,3}=1, x_{3,8}=1, x_{4,5}=1, x_{5,2}=1, x_{6,12}=1, x_{7,1}=1, x_{8,15}=1$ $x_{9,4}=1, x_{10,14}=1, x_{11,6}=1, x_{12,11}=1, x_{13,13}=1, x_{14,10}=1, x_{15,16}=1, x_{16,9}=1$	577
3.22	$x_{1,7}=1, x_{2,3}=1, x_{3,8}=1, x_{4,5}=1, x_{5,2}=1, x_{6,12}=1, x_{7,1}=1, x_{8,11}=1$ $x_{9,4}=1, x_{10,14}=1, x_{11,15}=1, x_{12,6}=1, x_{13,9}=1, x_{14,13}=1, x_{15,16}=1, x_{16,10}=1$	576
3.23	$x_{1,7}=1, x_{2,3}=1, x_{3,8}=1, x_{4,5}=1, x_{5,2}=1, x_{6,12}=1, x_{7,1}=1, x_{8,15}=1$ $x_{9,6}=1, x_{10,14}=1, x_{11,4}=1, x_{12,11}=1, x_{13,9}=1, x_{14,13}=1, x_{15,10}=1, x_{16,16}=1$	574

From the Table 3.23 of results, table number 3.8 gives the smallest Z value (567). Hence the assignments

$$x_{1,10} = 1, x_{2,8} = 1, x_{3,7} = 1, x_{4,5} = 1, x_{5,2} = 1, x_{6,12} = 1, x_{7,1} = 1, x_{8,15} = 1$$

$$x_{9,6} = 1, x_{10,4} = 1, x_{11,11} = 1, x_{12,14} = 1, x_{13,9} = 1, x_{14,16} = 1, x_{15,3} = 1, x_{16,13} = 1 \text{ is optimal}$$

Thus, the assignments made in Table 3.24 below is optimal

Table 3.24: Optimal Assignment of Trucks

Type of Vehicle	**Route to be assigned/Final destination**
KIA (4)	Drobo, Oseikojokrom, Bogoso and Yendi
TATA (4)	Kintampo, Asankraguaa, Yeji and Sefwi Juabeso
Renault (art.)	Nkawkaw
TATA (art.)	Kwame Danso
DAF (art.)	Obugu
Benz (art.) (2)	Assin Fosu and Goaso
DAF (Cargo)	Bawku
Benz (Cargo) (2)	Lawra and Gushiegu

CHAPTER 4

CONCLUSSION AND RECOMMENDATION

4.1 CONCLUSION

This study has found an optimal assignment of trucks to routes in terms of the total number of gallons of diesel required to travel to the final destinations.

Table 4.0 below shows the assignments of the trucks to the final destinations made in Table 3.24.

Table 4.0: Optimal assignment of trucks obtained from Table 3.24

Type of Vehicle	**Route to be assigned/Final destination**
KIA 1	Drobo
KIA 2	Osei Kojokrom
KIA 3	Bogoso
KIA 4	Yendi
TATA 1	Kintampo
TATA 2	Asankrguaa
TATA 3	Yeji
TATA 4	Sefwi Juabeso
Renault (art.)	Nkawkaw
TATA (art.)	Kwame Danso
DAF (art.)	Obugu
Benz (art.) 1	Assin Fosu
Benz (art.) 2	Goaso
DAF (Cargo)	Bawku
Benz (Cargo) 1	Lawra
Benz (Cargo) 2	Gushiegu

The minimum total cost, Z $= \sum_{i=1}^{16} \sum_{j=1}^{16} c_{ij} x_{ij}$ = 567 gallons of diesel.

4.2 RECOMMENDATIONS

It is suggested that Latex Foam Rubber Products Limited-Kumasi implements the Munkres assignment algorithm in assigning their trucks.

Also, to maintain the current set of routes and to avoid multiple assignments of the KIA and Benz (cargo) trucks, it is suggested that the company adds one each of these trucks to the existing set of fourteen trucks for optimal assignment to avoid delay in supply to customers, because this may cause some of their customers to switch to other suppliers of the products, due to the competitive nature of market.

References

Andras, F. (2004). *On Kuhn's Hungarian Method-A tribute from Hungary* EGRES Technical Report No 2004-14

Anshuman, S. and Rudrajit, T. (2007). *Solving the Assignment Problem Using Genetic Algorithm and Simulated Annealing*. IAENG International Journal of Applied Mathematics, 36:1

Beaujon G. J. and Turnquist M. A. (1991). *A Model for Fleet Sizing and Vehicle Allocation*: *Transportation Science*, **25** (1), pp. 19-45

Bhatia, H.L., Swarup, K. and Puri, M.C. (1975). *A Procedure for Time Minimizing Transportation Problem*. http://w.w.w.new.dli.ernet.in/rawdataupload/upload/insa/INSA_2/200(Retrieved on 24/05/2009)

Bradley, S. P., Hax, A.C. and Magnanti T.L. (1977). *Applied Mathematical Programming*, Addison-Wesley Publishing Company, San Francisco

Buehren, M. (2008). *Functions for Rectangular Assignment Problem*

http://www.mathworks.com/matlabcentral/fileexchange/6543 (Retrieved on 28/02/2009)

Clark, G. and Van Der Werf, Y. (1998). *Work in Progress? The Industrious Revolution*. *The Journal of Economic History* **58**; 3, pp 830-843

Clarke, C. and Wright, J.W. (1964). *Scheduling of Vehicles from Central Depot to a Number of Delivery Points*. Operations Research, 12; pp 568-581

Cooke, W.P. (1985). *Quantitative Methods for Management Decision*. McGraw-Hill, New York.

Dantzig, G.B. and Ramser, J.H. (1959). *The Truck Dispatching Problem*, Management Science; pp 80-91

David, R A., Dennis, J.S. and Thomas, A. W. (1988). *An Introduction to Management Science (5th Edition)*: West Publishing Company, San Francisco.

De Vries (1994).*The Industrial Revolution and Industrious Revolution*: *The journal of Economic History* 54: (2) 249-270

Fisher, M.L. (1973). *Optimal Solution of Scheduling Problems using Lagrange Multipliers: Part I*. Operations Research: Vol. 21, pp 1114-1127.

Ford, L.R. and Fulkerson, D.R. (1962). *Flow in Networks*, Princeton Univ. Press, Princeton NJ.

Geoffrion, A.M. (1974). *Lagrangian Relaxation and its Uses in Integer Programming*, Mathematics Programming Study: Vol. 2, pp 82-114

Green L.V. and Guha, D. (1995). *On the Efficiency of Imbalance in Multi-Facility Multi-Server Service Systems*: *Management Science*, **41**, pp. 179-187

Hammer, P.L. (1969). *Time Minimizing Transportation Problems*. Naval Research Quarterly 16; pp 345-357

Hartvigsen, D., Wei, J.C. and Czuchlewski, R. (1999) "*conference paper-reviewer assignment problem, The*" Decision Sciences.
http://www.findarticles.com/p/articles/mi_qa3713/is_199907 (Retrieved on 24/10/2008)

Held, M. and Karp, R. M. (1970).*The Travelling Salesman Problem and Minimum Spanning Trees: Part I*. Operations Research: Vol. **18**, pp 1138-1162

Held, M. and Karp, R.M. (1971).*The Travelling Salesman Problem and Minimum Spanning Trees: Part II*. Mathematical Programming: Vol. 1, pp 6-25

Hiller, F.S. and Lieberman,G. J. (2005). *Introduction to Operations Research*. McGraw-Hill, New York.

Hitchcock, F.L. (1941). *The Distribution of a Product from Several Sources to Numerous Locations,* J. Math Phys.20, pp 224-230

Jardine, L. (1996*). Conspicuous Consumption Worldly Goods: A New History of the Renaissance*. Norton, New York.

Joubert, J.W. (2007). *The Vehicle Routing problem: Origins and Variants*, University of Pretoria edt., pp 4

Kim, S. (2000). *Markets and Multiunit Firms from an American Historical Perspective*

http://soks.wustl.edu/mumm.p.d.f (Retrieved on 20/05/2009)

Kim, S. (1995). *Expansion of Markets and the Geographic Distribution of Economic Activities*: *Trends in U.S. Regional Manufacturing Structure, 1860-1987*, Quarterly Journal of Economics 110, 4: 881-908.

Koopman, T.C. (1947). *Optimum Utilization of Transportation System*, Proc. Intern. Statis. Conf., Washington, D.C

Löbel, A. (1998). *Vehicle Scheduling in Public Transit and Lagrangian Pricing*, *Management Science*, **44** (12), pp. 1637-1649

Lotfi, V. and Pegels, C. (1989). *Decision Support Systems for Management Science.* Operations Research, Irwin, Homewood

Mahmoud, T. A. (1996). *A -Level Business Management* (4th edition), Folie Books Ent., Tema.

Mathirajan, M. and Meenakshi, B. (2004). *Experimental Analysis of some Variants of Vogel's Approximation Method*: Journal of Operations Research. Vol.21 (4), pp 449

Michael, Z. P. and Powell, W. B. (2004). *The Dynamic Assignment Problem*. Transportation Science, Vol. 38, 4, pp 399-419.

Toth, P. and Vigo, D. (1997). Heuristic Algorithms for the Handicapped Persons Transportation Problem, Transportation Science, Vol. 31, No. 1, pp 60-71

Ross, E (1993). *Love and Toil Motherhood in Outcast London, 1870-1918*. Oxford University Press, New York

Rushmeier, R. A. and Kantogiorgis, S.A. (1997). *Advances in the Optimization of Airline Fleet Assignment: Transportation Science*, **31** (2), pp. 159-169.

Sharma, J. K. and Swarup, K. (1978). *Time Minimization in Transportation Problems*, N.Z.O.R 6; pp 75-88

Sokoloff, K. L. (1988). "Inventive Activity in Early Industrial America: Evidence From Patent Records, 1790-1846," *Journal of Economic History* 48: 813-849.

Sokoloff, K. L. (1984a). *Investment in Fixed and Working Capital during Early Industrialization: Evidence from U.S. Manufacturing Firms*, Journal of Economic History 44: 545-556.

Sokoloff, K. L. (1984b). *Was the Transition from the Artisanal Shop to the Non-Mechanized Factory Associated with Gains in Efficiency? Evidence from the U.S. Manufacturing Censuses of 1820 and 1850*, Explorations in Economic History 21: 351-382.

Szwarc, W. (1964). *The Transportation Problem with Stochastic Demand*, Management Science, 11; pp 33-50

Whitt, W. (1992). *Understanding the Efficiency of Multi-Server Service Systems. Management Science*, **38**, pp. 708-723

Wiesner-Hanks, M. E. (2006). *Early Modern Europe 1450-1789*, pp 202-205,418-419

Wikipedia. (2009). *Industrial Revolution*

http://en.wikipedia.org/wiki/Industrial_revolution (Retrieved on 16/02/2009)

Williams, A. C. (1963*). A Stochastic Transportation Problem*, Operations Research, 11; pp 759-770

Wilson, D. (1972). *An a Priori Bounded Model for Transportation Problems with Stochastic Demand and Integer Solution*. AIIE Trans., 3; pp 186-193

Wilson, D. (1973). *Tighter Bounds for Stochastic Transportation Models*. AIIE Trans., 5; pp 180-185

Wilson, D. (1975). *A Mean Cost Approximation for Transportation Problems with Stochastic Demand*, Naval Research Quarterly; 22

Woeginger,G. J, Burkard, R. and Cela, E. (1995). *A minimax assignment problem in treelike communication networks* **,** European Journal of Operational Research http://www.faqs.org/abstracts/Business-international/A-minimax-assignment-problem-in-treelike-communication-networks.html#ixzz0Sg8mXbSH (Retrieved on 05/08/2009)

Xu, J. and Ye, X. (2008). A fuzzy Vehicle Routing Assignment Model, World Journal of Modeling and Simulation, Vol. 4; 2, pp 257-268

Zeleny, M. (1982). *Multiple Criteria Decision Making*, McGraw-Hill, New York.

Ziarati, K., Soumis ,F., Desrosiers, J. and Solomon ,M. M. (1999). *A Branch-First*, *Cut-Second Approach for Locomotive Assignment*. *Management Science*, **45**, pp. 1156-1168

Appendix

Mat Lab code for implementing Munkres Algorithm

```
function [assignment, cost] = assignmentoptimal(distMatrix)
%ASSIGNMENTOPTIMAL    Compute optimal assignment by Munkres algorithm
```

```
%       ASSIGNMENTOPTIMAL (DISTMATRIX) computes the optimal assignment
%(minimum overall costs) for the given cost matrix.
%The result is a column vector containing the assigned column
% number to each row (or 0 if no assignment could be done).
% [ASSIGNMENT, COST] = ASSIGNMENTOPTIMAL(DISTMATRIX) returns the
%assignment vector and the overall cost (total number of gallons of
%diesel)
% The distMatrix may contain infinite values (forbidden assignments).
%Internally, the infinite values are set to a very large
% finite number, so that the Munkres algorithm itself works on
% finite-number matrices. Before returning the assignment, all
% assignments with infinite distance are deleted (i.e. set to zero).
%
%       Code written by Markus Buehren
%       Last modified 30.01.2008

% save original distMatrix for cost computation
originalDistMatrix    = distMatrix;

% check for negative elements
if any(distMatrix(:) < 0)
    error('All matrix elements have to be non-negative.');
end

% get matrix dimensions
[nOfRows, nOfColumns] = size(distMatrix);

% check for infinite values
finiteIndex   = isfinite(distMatrix);
infiniteIndex = find(~finiteIndex);
if ~isempty(infiniteIndex)
    % set infinite values to large finite value
    maxFiniteValue = max(max(distMatrix(finiteIndex)));
    if maxFiniteValue > 0
        infValue = abs(10 * maxFiniteValue * nOfRows * nOfColumns);
    else
        infValue = 10;
    end
    if isempty(infValue)
        % all elements are infinite
        assignment = zeros(nOfRows, 1);
        cost       = 0;
        return
    end
    distMatrix(infiniteIndex) = infValue;
end

% memory allocation
coveredColumns = zeros(1,nOfColumns);
coveredRows    = zeros(nOfRows,1);
starMatrix     = zeros(nOfRows, nOfColumns);
primeMatrix    = zeros(nOfRows, nOfColumns);

% preliminary steps
if nOfRows <= nOfColumns
    minDim = nOfRows;
```

```
    % find the smallest element of each row
    minVector = min(distMatrix,[],2);

    % subtract the smallest element of each row from the row
    distMatrix = distMatrix - repmat(minVector, 1, nOfColumns);

    % Steps 1 and 2
    for row = 1:nOfRows
        for col = find(distMatrix(row,:)==0)
            if ~coveredColumns(col)%~any(starMatrix(:,col))
                starMatrix(row, col) = 1;
                coveredColumns(col)  = 1;
                break
            end
        end
    end

else % nOfRows > nOfColumns
    minDim = nOfColumns;

    % find the smallest element of each column
    minVector = min(distMatrix);

    % subtract the smallest element of each column from the column
    distMatrix = distMatrix - repmat(minVector, nOfRows, 1);

    % Steps 1 and 2
    for col = 1:nOfColumns
        for row = find(distMatrix(:,col)==0)'
            if ~coveredRows(row)
                starMatrix(row, col) = 1;
                coveredColumns(col)  = 1;
                coveredRows(row)     = 1;
                break
            end
        end
    end
    coveredRows(:) = 0; % was used auxiliary above

end

if sum(coveredColumns) == minDim
    % algorithm finished
    assignment = buildassignmentvector__(starMatrix);
else
    % move to step 3
    [assignment, distMatrix, starMatrix, primeMatrix, coveredColumns,
coveredRows] = step3__(distMatrix, starMatrix, primeMatrix,
coveredColumns, coveredRows, minDim); %#ok
end

% compute cost and remove invalid assignments
[assignment, cost] = computeassignmentcost__(assignment,
originalDistMatrix, nOfRows);
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%
function assignment = buildassignmentvector__(starMatrix)

[maxValue, assignment] = max(starMatrix, [], 2);
assignment(maxValue == 0) = 0;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%
function [assignment, cost] = computeassignmentcost__(assignment,
distMatrix, nOfRows)

rowIndex   = find(assignment);
costVector = distMatrix(rowIndex + nOfRows * (assignment(rowIndex)-1));
finiteIndex = isfinite(costVector);
cost = sum(costVector(finiteIndex));
assignment(rowIndex(~finiteIndex)) = 0;

% Step 2:
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [assignment, distMatrix, starMatrix, primeMatrix,
coveredColumns, coveredRows] = step2__(distMatrix, starMatrix,
primeMatrix, coveredColumns, coveredRows, minDim)

% cover every column containing a starred zero
maxValue = max(starMatrix);
coveredColumns(maxValue == 1) = 1;

if sum(coveredColumns) == minDim
    % algorithm finished
    assignment = buildassignmentvector__(starMatrix);
else
    % move to step 3
    [assignment, distMatrix, starMatrix, primeMatrix, coveredColumns,
coveredRows] = step3__(distMatrix, starMatrix, primeMatrix,
coveredColumns, coveredRows, minDim);
end

% Step 3:
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [assignment, distMatrix, starMatrix, primeMatrix,
coveredColumns, coveredRows] = step3__(distMatrix, starMatrix,
primeMatrix, coveredColumns, coveredRows, minDim)

zerosFound = 1;
while zerosFound

    zerosFound = 0;
    for col = find(~coveredColumns)
        for row = find(~coveredRows')
            if distMatrix(row,col) == 0
```

```
                primeMatrix(row, col) = 1;
                starCol = find(starMatrix(row,:));
                if isempty(starCol)
                    % move to step 4
                    [assignment, distMatrix, starMatrix, primeMatrix,
coveredColumns, coveredRows] = step4__(distMatrix, starMatrix,
primeMatrix, coveredColumns, coveredRows, row, col, minDim);
                    return
                else
                    coveredRows(row)        = 1;
                    coveredColumns(starCol) = 0;
                    zerosFound              = 1;
                    break % go on in next column
                end
            end
        end
    end
end

% move to step 5
[assignment, distMatrix, starMatrix, primeMatrix, coveredColumns,
coveredRows] = step5__(distMatrix, starMatrix, primeMatrix,
coveredColumns, coveredRows, minDim);

% Step 4:
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [assignment, distMatrix, starMatrix, primeMatrix,
coveredColumns, coveredRows] = step4__(distMatrix, starMatrix,
primeMatrix, coveredColumns, coveredRows, row, col, minDim)

newStarMatrix          = starMatrix;
newStarMatrix(row,col) = 1;

starCol = col;
starRow = find(starMatrix(:, starCol));

while ~isempty(starRow)

    % unstar the starred zero
    newStarMatrix(starRow, starCol) = 0;

    % find primed zero in row
    primeRow = starRow;
    primeCol = find(primeMatrix(primeRow, :));

    % star the primed zero
    newStarMatrix(primeRow, primeCol) = 1;

    % find starred zero in column
    starCol = primeCol;
    starRow = find(starMatrix(:, starCol));

end
starMatrix = newStarMatrix;
```

```
primeMatrix(:) = 0;
coveredRows(:) = 0;

% move to step 2
[assignment, distMatrix, starMatrix, primeMatrix, coveredColumns,
coveredRows] = step2__(distMatrix, starMatrix, primeMatrix,
coveredColumns, coveredRows, minDim);


% Step 5:
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [assignment, distMatrix, starMatrix, primeMatrix,
coveredColumns, coveredRows] = step5__(distMatrix, starMatrix,
primeMatrix, coveredColumns, coveredRows, minDim)

% find smallest uncovered element
uncoveredRowsIndex    = find(~coveredRows');
uncoveredColumnsIndex = find(~coveredColumns);
[s, index1] =
min(distMatrix(uncoveredRowsIndex,uncoveredColumnsIndex));
[s, index2] = min(s); %#ok
h = distMatrix(uncoveredRowsIndex(index1(index2)),
uncoveredColumnsIndex(index2));

% add h to each covered row
index = find(coveredRows);
distMatrix(index, :) = distMatrix(index, :) + h;

% subtract h from each uncovered column
distMatrix(:, uncoveredColumnsIndex) = distMatrix(:,
uncoveredColumnsIndex) - h;

% move to step 3
[assignment, distMatrix, starMatrix, primeMatrix, coveredColumns,
coveredRows] = step3__(distMatrix, starMatrix, primeMatrix,
coveredColumns, coveredRows, minDim);
```