A GOAL PROGRAMMING MODEL OF THE STOCHASTIC VEHICLE ROUTING PROBLEM

By

YAHIA ZARE-MEHRJERDI

Bachelor of Science in Mathematics
Iranian National University
Tehran, Iran
1976

Bachelor of Science in Civil Engineering
Texas A & I University
Kingsville, Texas
1980

Master of Science in Operations Research
St. Mary's University
San Antonio, Texas
1983

Submitted to the Faculty of the Graduate College
of the Oklahoma State University
in partial fulfillment of the requirements
for the Degree of
DOCTOR OF PHILOSOPHY
December 1986

A GOAL PROGRAMMING MODEL OF THE STOCHASTIC

VEHICLE ROUTING PROBLEM

Thesis Approved:

Marvin Palmer Terrell
Thesis Adviser

Wayne C. Turner

Allen C. Schuermann

R. Gerald Folks

James E. Shamblin

Norman N. Durham
Dean of the Graduate College

PREFACE

This research incorporates the concept of chance-constrained programming and multiple objective goal programming in the area of vehicle routing problems. The research led to the development of the model of the Goal Programming (GP) Stochastic Vehicle Routing Problem (SVRP) that allows the decision makers involvement in the solution process of problem to obtain satisfactory vehicle routes for the SVRP. It is shown that, mathematically, a new set of deterministic linear time constraints are equivalent to the nonlinear set of time constraints of the problem for distributions such as poisson and chi-square. Additionally, the effects of the route failing probabilities on the total elapsed time of the whole delivery system, and the existence of the optimum solution for the "F" type problem are proven mathematically.

A modification of the Clarke and Wright algorithm is developed to determine the most favorable vehicle routes of the SVRP for the "E" type problem. Additionally, two heuristic algorithms which are the modification of the Clarke and Wright "savings" approach are developed for solving the "F" type problem. Computational experiments are performed on three test problems to justify the proposed algorithms. Two interactive computer programs are developed for the SVRP and goal programming technique which allows the decision maker to provide satisfactory vehicle routes.

I wish to express my sincere appreciation and respect to my committee chairman, Dr. M. Palmer Terrell, for his guidance and support throughout this research and my doctoral program. I also appreciate Dr. Keolling, who first brought this problem to my attention.

Appreciation is also extended to my committee members, Drs. Allen C. Schuermann, James Shamblin, Wayne C. Turner, and J. Leroy Folks, for their interest and assistance during this study.

I also want to thank Shirley Motsinger and Kenna Long for their virtually faultless typing and valuable suggestions concerning this dissertation.

I wish to convey my special appreciation to my parents, particularly my mother, and to my brothers, sisters, and other relatives for their encouragement and support during my study in the United States.

Finally, to my wife, Mary, and my son, Omid, I wish to express my heartfelt thanks for the many sacrifices they made to allow me to pursue my goals. Without their understanding, support, and love, the days and nights spent in school preparing this study would have been impossible.

TABLE OF CONTENTS

Chapter Page

Chapter Page

LIST OF TABLES

Table | Page

LIST OF FIGURES

CHAPTER I

INTRODUCTION

1.1 Background

Managers of private firms involved in the distribution of goods from a warehouse or depot to designated delivery points, as well as authorities responsible for public or private transportation systems, have become increasingly aware of the need to maximize operational efficiency and minimize delivery costs such as fuel, replacement of vehicles, and labor. To illustrate the economical significance of the Vehicle Routing Problem (VRP), Bodin [7] used a survey by Kearney (1980) to show that about 16 percent of the sales value of an item is based on the physical distribution costs of that item, and of this, about one-fourth is due to downstream distribution of the final product from distribution centers to customers. Turner and Vu further reported that in 1974 about 10 percent of the average community's budget was spent on refuse collection and disposal, a total of 7.8 billion dollars [60]. Factors such as these have attracted a great deal of attention to the VRP, and very recently, to the Stochastic Vehicle Routing Problem (SVRP).

The result of this unprecedented interest has been the development and utilization of computerized procedures to solve certain types of vehicle routing problems which reduce associated distribution costs and delivery time and increase customer satisfaction.

Briefly then, an essential element in any logistics routing system is the allocation and routing of vehicles for the purpose of collecting and delivering goods and services on a regular basis. However, routing decisions are complicated by the need of managers to reduce associated costs, and at the same time, satisfy customer demands by making certain that goods are delivered safe, at the right time, and in the right quantity.

The Vehicle Routing Problem (VRP) is a generic name given to a whole class of problems involving the visiting of customers by vehicles [6, 13]. The VRP is also known in the literature as "vehicle scheduling" [8, 11, 17, 24, 35, 41, 49, 64, 65, 67], "vehicle dispatching" [19, 26, 27, 50], or "delivery problem" [57, 58, 59]. Dantzig and Ramser are generally credited for the first formulation of the VRP as presented in their 1959 paper "The Truck Dispatching Problem" [19]. The VRP can be stated as follows: given a set of nodes and arcs to be visited by a fleet of vehicles, construct a low-cost, feasible set of routes for each vehicle [7].

The SVRP is to design a set of feasible routes starting from and eventually returning to a central depot, in order to deliver commodities to a finite number of demand points with randomly distributed customer demands, randomly distributed travel and unload times having known distribution functions, such that the capacity constraint and time constraints of the problem are satisfied. If, however, the amounts of demand at each location, travel time between any two stations, and unload time at each location are known with certainty, and providing that a vehicle capacity restriction exists, this problem is a deterministic VRP.

In the multiple objective SVRP, more than one criteria are considered in the same problem which, depending on the nature of the criteria, are either maximized or minimized. For example, if the safety of the products on the vehicle route is considered to be one of the criteria, it is to be maximized. If, on the other hand, the total cost or total elapsed time is considered to be one of the criteria, it is to be minimized.

Relevant objective functions for the SVRP may contain the following:

1. Minimize the total cost

2. Minimize the total elapsed time on the route (travel time and unload time)

3. Maximize the safety of products on the route

4. Maximize the fulfillment of emergency services [49]

5. Maximize the fulfillment of conditional dependencies of stations such as deadlines and earliest delivery times [49]

6. Minimize the total deterioration of goods on the route

7. Minimize the safety stock for each vehicle route (this is due to the nature of the probabilistic demands)

Frequently, managers are interested in achieving two or more of the above objectives up to satisfactory levels instead of optimizing a single criteria. Goal Programming (GP), which is one of the techniques for multiple objective decision analysis, can be employed to provide a simultaneous solution to this system of competing objectives. Hence, it is desirable to formulate a GP model of the problem within the framework of the SVRP, such that capacity constraints, time constraints, customer demand, and decision making requirements are satisfied. Due to the complexity of the SVRP, a set of stations to be visited by a fleet of vehicles needs to be partitioned into feasible sets of routes, one for each vehicle enabling the application of the multiple objective GP technique to each of the vehicle routes. The multiple objective SVRP, then, consists of the following two major stages:

Stage I: Route Construction Stage (RCS)

Stage II: Route Improvement Stage (RIS)

The primary task of the RCS is partitioning a set of stations which are scattered around the central depot, into feasible subsets by applying a VRP heuristic approach. Using concepts of the GP technique, the RIS is used to sequence the stations on each vehicle route to meet the customers' and decision makers' requirements.

1.2 Research Objectives

The primary and secondary objectives of this research are described more specifically in the following sections.

1.2.1 Primary Objectives

The primary objectives of the proposed research are as follows:

1. Within the framework of the SVRP, develop a mathematical formulation for a multiple objective GP model. To accomplish this objective, the following subobjectives must be met:

   a. Develop a formulation of the SVRP in which travel time, unload time, and customer demands may be represented as random variables having known distribution functions.

b. Transform the general SVRP into an equivalent deterministic VRP for each stage of the problem.

c. Mathematically prove the existence of a set of deterministic linear time constraints which are equivalent to the nonlinear set of time constraints of the problem for distributions such as the Poisson and chi-square.

d. Develop the Linear Goal Programming (LGP) mathematical formulation of the RIS of the problem where the conflicting multiple objectives are treated explicitly.

e. Mathematically prove the effects of the route failing probabilities of α_k and β_k of the total elapsed time of the system where $0 \leq \alpha_k \leq 1$ and $0 \leq \beta_k \leq 1$ for all k.

2. Determine an appropriate solution technique for the RCS of the problem. In order to accomplish this objective, the following subobjectives must be met:

a. Mathematically prove the existence of the optimum solution for the RCS of the problem.

b. Develop an algorithm that gives the most satisfactory vehicle routes for the RCS of the problem.

1.2.2 Secondary Objectives

The secondary objectives to be achieved are as follows:

1. Develop a computer program of the algorithm for the heuristic approach which is designed to construct feasible vehicle routes in the RCS of the problem.

2. Develop a computer program of the interactive LGP procedure that will allow the decision maker's involvement in the solution process of the RIS of the problem.

The scope of the proposed research is limited to the single depot, multiple vehicle, node routing problem with stochastic demand and travel and unload times and the development of the multiple objective goal programming formulation of the SVRP.

1.3 Outline of Succeeding Chapters

Chapter I defines the problem and states the objectives and subobjectives of this research. Chapter II reviews the existing literature and the solution techniques of the VRP and SVRP. Chapter III discusses Chance-Constrained Programming (CCP) used with random variables in programming models. Chapter IV reviews the literature on linear goal programming techniques. In Chapter V, the SVRP and its equivalent deterministic forms are developed and some necessary theorems are proven. Chapter VI is devoted to the development of linear integer goal programming (LIGP) techniques. Chapter VII demonstrates the development of an appropriate heuristic approach for solving the SVRP. The heuristic approach developed in this study is a modification of the Clark and Wright algorithm. Chapter IX discusses the details of the interactive computer programs for the SVRP and LIGP techniques. Chapter X gives a conclusion and recommendations for future research in the field of SVRP.

CHAPTER II

LITERATURE REVIEW

2.1 Vehicle Routing Problem

The VRP is a challenging logistics management problem with variations that range from school bus routing to the dispatching of delivery trucks for consumer goods. Regardless of the variations, the basic components of the problem are a fleet of vehicles with fixed capacities and a set of demands for transporting passengers or certain objects (consumer goods, etc.) between specified depots and delivery points. The problem is complicated because managers must also take into consideration a variety of constraints such as fixed vehicle capacity and the duration of a route.

Some of the problems classified under the generic name are the Travelling Salesman Problem (TSP) and its variants; Multiple TSP and Time Constrained TSP; Single Depot, Multiple Vehicle Node Routing (SMVR); Multiple Depot, Multiple Vehicle, Node Routing (MMVR); and Single Depot, Multiple Vehicle, Node Routing Problem (SMVR) with stochastic demands. These problems have a pronounced discrete and combinational structure and are problems in the mathematical programming area known as "combinatorial optimization."

The TSP, a combinatorial optimization problem with some real life applications, is the substructure of all VRP's [14] and has been studied extensively in the literature. Dantzig and Ramser [19] describe the TSP as follows: "Find the shortest route (tour) for a salesman, starting from a given city, visiting each of a specified group of cities, and returning to the original point of departure" [19, p. 80]. Mathematically, this problem can be formulated as:

$$\text{Minimize} \quad \sum_{i=1}^{N} \sum_{j=1}^{N} C_{ij} X_{ij} \tag{2.1}$$

$$\text{Subject to:} \quad \sum_{i=1}^{N} X_{ij} = 1, \text{ for all } j \in S = \{1,2,\ldots,N\} \tag{2.2}$$

$$\sum_{j=1}^{N} X_{ij} = 1, \qquad \text{for all } i \in S \tag{2.3}$$

$$X_{ij} = \begin{cases} 0 \\ 1 \end{cases}, \qquad \text{for all } i,j \in S \tag{2.4}$$

$$X_{ij} \text{ (form a tour)} \tag{2.5}$$

where C_{ij} is the cost of travelling from node i to node j, $C_{ii} = \infty$, where $i = 1,2,\ldots,N$. Constraint (2.5) can thus be written in the form of

$$Z_i - Z_j + NX_{ij} \leq N - 1, \text{ for } 2 \leq i \neq j \leq N \tag{2.6}$$

and for some nonnegative real numbers Z_i.

Since 1959, when Dantzig and Ramser [19] first introduced the VRP and proposed a linear programming based heuristic for its solution, the heuristic method has been widely researched [15, 27]. Christofides and Eilon [13] indicated the largest VRP of any complexity solved to date by exact methods and reported in the open literature contains only 31 demand points. Before considering different approaches for solving the VRP, a formulation of the problem as a 0-1 integer program is given. This problem, known as the "pure delivery" problem, can be formulated as follows [31]:

$$\text{Minimize:} \quad \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{k=1}^{NV} d_{ij} X_{ijk} \tag{2.7}$$

$$\text{Subject to:} \quad \sum_{i=1}^{N} \sum_{k=1}^{NV} X_{ijk} = 1 \qquad j = 2,3,\ldots, N \tag{2.8}$$

$$\sum_{i=1}^{N} X_{ipk} - \sum_{j=1}^{N} X_{pjk} = 0 \qquad k = 1,2,\ldots, NV \quad p = 1,2,\ldots, N \tag{2.9}$$

$$\sum_{i=1}^{N} d_i \left(\sum_{j=1}^{N} X_{ijk}\right) \le Q_k \qquad k = 1,2,\ldots, NV \tag{2.10}$$

$$\sum_{j=2}^{N} X_{1jk} \le 1 \qquad k = 1,\ldots, NV \tag{2.11}$$

$$Z_i - Z_j + N \sum_{k=1}^{NV} X_{ijk} \le N - 1 \qquad i \neq j = 1,2,\ldots,N \tag{2.12}$$

$$\sum_{i=1}^{N} t_{ik} \sum_{j=1}^{N} X_{ijk} + \sum_{i=1}^{N} \sum_{j=1}^{N} t_{ijk} X_{ijk} \le T_k \qquad k = 1,2,\ldots,NV \tag{2.13}$$

$$X_{ijk} = \begin{cases} 0 \\ 1 \end{cases} \quad \text{for all } i,j,k, \text{ and } i \neq j \tag{2.14}$$

where

N = number of nodes

NV = number of vehicles

Q_k = capacity of truck k

T_k = maximum time allowed for vehicle k on a route

d_i = demand at node i ($d_1 = 0$)

t_{ik} = time required for vehicle k to deliver or collect at node i ($t_{1k} = 0$)

t_{ijk} = travel time for vehicle k from node i to node j ($t_{iik} = \infty$)

d_{ij} = distance from node i to node j

$$X_{ijk} = \begin{cases} 1 & \text{if arc } (i,j) \text{ is traversed by vehicle } k \\ 0 & \text{otherwise} \end{cases}$$

Z_i = arbitrary real numbers, $i = 1,2,\ldots,N$

The objective function (2.7) represents minimization of total distance travelled by NV vehicles. Alternatively, costs could be minimized by replacing d_{ij} with C_{ij}, depending on the vehicle type. Equation (2.8) ensures that each demand node is served by exactly one vehicle; equation (2.9) ensures that if a vehicle enters a demand node it must exit from that node; equation (2.10) is the vehicle capacity constraint and (2.11) guarantees that vehicle availability is not exceeded; equation (2.12) prohibits subtours; and finally, equation (2.13) is the total elapsed route time constraint.

2.2 Solution Techniques for the VRP

2.2.1 Background

Solution techniques for the VRP fall into two categories: those which solve the problem heuristically and those which solve the problem optimally. Basically, heuristic techniques have proved to be an attractive alternative to exact methods because they are easy to understand, readily accepted by managers, easy to program and maintain for computerized planning, and effective in solving a wide range of practical problems which provide solutions that are usually accepted as "reasonable" [35]. The literature review concentrates on single-depot, multiple-vehicle and multiple-depot, and multiple-vehicle situations.

2.2.2 Heuristic Algorithms

The majority of the previous efforts on the VRP have involved heuristic algorithms. Also, the heuristic methods which have been developed for the VRP are largely modifications of TSP heuristics. These algorithms can be categorized into the following four groups:

1. Tour building heuristics,
2. Tour improvement heuristics,
3. Two-phase methods, and
4. Lagrangian relaxation heuristics.

2.2.2.1 Tour Building Heuristics. The Clarke and Wright "savings" approach is the one used most often in tour building heuristics [17, 30, 31]. This approach calculates the saving between nodes i and j, S_{ij}, as shown below:

$$S_{ij} = C_{0i} + C_{0j} - C_{ij} \tag{2.15}$$

where C_{ij} is the delivery cost for moving goods from node i to node j. More detail of this approach is given in Section 7.2. Gaskell [25] introduced the following alternatives that give results which are at least as good as the one found by Clarke and Wright's procedure. The savings are calculated as shown below:

$$\lambda_{ij} = S_{ij}\ [\bar{d} + | d_{oi} - d_{ij} | - d_{ij}] \text{ and} \tag{2.16}$$

$$\pi_{ij} = S_{ij} - d_{ij} \tag{2.17}$$

where $\bar{d}$ is the average of all d_{oi}. The rest of the procedure is the same as Clark and Wright's; however, the concept of modified savings can be given by $\pi_{ij} = S_{ij} - \theta d_{ij}$ where θ is a shape parameter. By varying θ, the analyst can place greater or less emphasis on the cost of travel between two nodes, depending on their position relative to the depot.

Yellow [67] suggested using a simple geometrical search technique on an ordered list of the polar coordinates of the delivery points. The saving was defined as

$$S = d_{oi} + d_{oj} - \gamma d_{ij} \tag{2.18}$$

where γ is the shape parameter. Special cases are $\gamma = 1$ for the Clarke and Wright procedure and $\gamma = 2$ for Gaskell's π method. Equation (2.18) may be expressed by polar coordinates relative to the delivery depot

$$S = r_i + r_j - \gamma\ (r_i^2 + r_j^2 - 2r_ir_j \cos\ (\theta_i - \theta_j))^{\frac{1}{2}} \tag{2.19}$$

where r_i and θ_i are the polar coordinates of point i. The rest of the procedure is the same as Clarke and Wright's.

Tillman and Cochran [59] modified the Clarke and Wright algorithm. The essential difference of the two methods is that Tillman and Cochran's method allows for the inclusion of restrictions on the system, and in some cases, will yield a better answer.

Holmes and Parker [35] constructed an extension of Clarke and Wright's approach. This new approach is concerned with the classical VRP where a set of vehicles with known capacities service a known set of points with deterministic demands at the lowest possible cost. The mechanics of the so-called "saving" approach are utilized as the foundation of the algorithm. This procedure is capable of handling the symmetric and nonsymmetric interpoint distances (costs) matrix.

Mole and Jameson [48] proposed a technique which is largely dependent on the Clarke and Wright savings criterion and the r-opt method introduced by Lin and Kernighan [42]. In this technique, a general parametric criterion of the following form was developed for including a node C between nodes A and B in the tour:

$$MSAV_c\ (A,B) = \lambda d_{oc} + \mu d_{AB} - d_{AC} - d_{BC}$$

where λ and μ are the route shape parameters. In the case where node C is introduced between depot o and node K, the above equation can modify as:

$$MSAV_c\ (K,o) = (\lambda - 1)\ d_{oc} + \mu d_{ok} - d_{kc} \tag{2.20}$$

For λ in the range of $1 \leq \lambda \leq 2$ and $\mu = \lambda - 1$, a ranking identical to Gaskell's π criterion would be generated from the latter equation where $\pi = (\lambda - 1)^{-1} - 1$. This sequential route building algorithm may be thought of in terms of a repeating sequence of the following steps:

1. Determine the most advantageous position to introduce customer C.

2. Identify the next customer to be placed on the emerging route.

3. Possible resequencing of customers on the emerging route is explored using the r-optimal technique.

Buxery [8] proposed a new model for planning the VRP using the "savings" heuristic rule along with the Monte Carlo simulation, subject to a maximum load restriction. The heart of this technique is similar to the one developed by Clarke and Wright; i.e., "Its function is to monitor the feasibility of the chosen new journey, at any particular juncture, for incorporation into the existing route pattern" [8, p. 566]. The main idea for utilizing the Monte Carlo simulation is based on (1) all methods rely a great deal on time consuming "trial and error" evaluation procedures, and (2) good solutions cannot be obtained without explicitly constructing some alternatives. The procedure requires various parameters such as location of depot, location of demand points, demands, the number of point-pairs contained in the selection list, weighting factor M to control the relative probability of generating each point-pair from the selection list, and finally, the run length if it is desired.

Williams [64] proposed a heuristic technique that could be used in attaining a visual solution. This method is based on joining customers farthest from the depot to the closest feasible customers within the immediate proximity. The route construction starts with nodes at extreme points in the area in order to avoid single long journeys and to minimize the total distance as nodes are added to the solution. Linking together the closest nodes to the peripheral starting point will generally minimize the distance travelled to service those nodes; thus, sorting of the distance matrix is highly reduced because initially only the closest node is required. After the initial link of a route has been found, then, from the distance matrix, the closest two feasible nodes to the farthest node is a link which has two nodes to which nodes can be assigned. A feasible node is a node that, if added to a link, will not cause the link to violate any restrictions.

As previously mentioned, the VRP has been studied widely, but the multidepot VRP has attracted less attention and only a few articles are presented in the literature. Tillman [57], however, is credited for introducing the multiple terminal delivery problem. Specifically, the procedure begins with an initial feasible solution by assigning each vehicle to its closest depot. The algorithm is based on the "saving" criterion that was developed by Clarke and Wright [17]. Generally, this method involves determining savings from joining points on routes and making possible assignments as a function of the maximum savings for joining demand points on routes. The algorithm permits restrictions to be imposed on the system. One such procedure, however, is Tillman and Cain's [58] modification of the Clarke and Wright procedure which determines the initial solution by passing exactly one route from each demand point to the closest depot. When the distance between demand points i and j (d_{ij}) and the farther distance between demand point i and depot k (U_{ik}) is known, then the total distance of all routes is defined as

$$D = \sum_{i=1}^{N} 2 \min_{k} \{U_{ik}\}$$

where N is the number of demand points. This method successfully links pairs of nodes in order to decrease the total distance travelled. However, it should be noted that the computation of savings is not as straightforward as in the case of a single depot problem. Hence, the savings S_{ijk} must be evaluated by

$$S_{ijk} = \tilde{U}_{ik} + \tilde{U}_{jk} - d_{ij} \qquad (2.21)$$

where

$$\tilde{U}ik = \begin{cases} 2 \min_t \{Uit\} - Uik & \text{if i has not yet been given a permanent assignment} \\ Uik & \text{otherwise.} \end{cases} \qquad (2.22)$$

Savings S_{ijk} are computed for $i,j = 1,2,\ldots,N$ $(i \neq j)$ and $k = 1,2,\ldots,M$ at each step and can be stored in M matrices, each N by N.

Golden, Magnanti, and Nguyen [31] have proposed two algorithms for the multiterminal VRP. The first is based on the saving criterion method and the other is based on the Gillett and Johnson's philosophy [27]. The "saving" based algorithm uses Tillman and Cain's approach for computing savings but excludes the idea of a penalty function. The second algorithm is precisely developed for large problems where the multi-depot VRP is viewed as a two-step process: first, nodes have to be allocated to depots and then routes are built which link nodes assigned to the same depot. A large problem is introduced by dividing it into as many subproblems as there are depots and then solving each problem separately [27].

2.2.2.2 Tour Improvement Heuristics Approach. The best known heuristic approach for the TSP is the branch exchange approach introduced by Lin (1965) and later modified by Lin and Kernighan [45]. Lin and Kernighan define a tour to be r-optimal if no improvement can be made by replacing any r of its links with any other set of r links. An r-optimal tour has a certain probability of being optimal, and Lin suggests that three-optimal tours should normally be used since these give the best trade-off between computing time and probability of the tour is optimal.

Christofides and Eilon [13], who have modified the Lin "r-opt" procedure, developed a new approach that starts with a feasible solution and tests perturbations to obtain r-optimality. This approach for r = 2 examines each pair of arcs to build a new feasible and economical route which is replaced by any two old arcs from the route. The chief advantage, however, is that it is able to handle restrictions such as

1. Customer wants delivery at a certain time,
2. Capacity may vary between vehicles, and
3. Customer wants delivery by a certain vehicle.

Christofides and Eilon [15] and Lin and Kernighan [45] have shown that the number of operations needed for an r-optimal tour is polynomial in n (number of customers on a tour), exponential in r, and bounded below by n^r; thus, only "small" values of r can be used. Additionally, Christofides and Eilon discovered that when all possible links are considered in joining r changes into a tour, approximately $\binom{n}{r}(r-1)!2^{r-1}$ combinations need to be checked in order to ensure r-optimality.

Wren and Holliday [65] generated a customer list in order of the angular coordinate along the most sparse direction. In contrast to the Clarke and Wright method, the number of vehicles available at the depot must first be specified, which allows routes to be built up regarding the number of vehicles available. Customers are then introduced into the algorithm and each customer is assigned to a vehicle. Next, a "refine" procedure is activated to determine whether improvements can be obtained by simple categories, resequencing within routes, or reallocation between routes. The coordinate axis is rotated in equal increments of 90°, the algorithm is repeated each time, and the best of the four resulting route structures is chosen. This heuristic approach is capable of handling both single and multiple depot VRPs.

Russell [50] extended the Lin and Kernighan heuristic procedure to an approach called "MTOUR." It is directly analogous to the Christofides and Eilon [15] method in which they extend Lin's 3-opt TSP heuristic procedure to solve the vehicle dispatching problem. MTOUR is able to handle side conditions such as due date or interval constraints requiring that a visit be made only during certain time intervals. The MTOUR algorithm requires a feasible solution of the VRP with M vehicles as input. This MTOUR solution is expressed as a travelling salesman tour on an expanded network, then a modified 3-opt procedure or any other improvement scheme is used to reduce total cost [7]. At each step of the modified 3-opt procedure, a check for feasibility must be carried out to have an improved total cost and feasible solution. Run times, however, grow approximately as $N^{2.3}$, where N is the number of demand points.

2.2.2.3 Two-Phase Methods. In the two-phase method, customers are first assigned to vehicles without specifying the sequence in which customers are visited. In the second phase, routes are obtained for each vehicle using a TSP heuristic. The procedures introduced by Gillett and Miller [26] and Christofides and Eilon [15] are two-phase methods that use a modified Lin-Kernighan heuristic in phase two.

Gillett and Miller [26] also introduced an algorithm called the "sweep" algorithm. This algorithm consists of two parts, a forward sweep and a backward sweep. In this procedure, the problem is broken down into smaller subproblems which can be solved more easily. The locations are ordered according to their polar coordinate angles from a central depot and assigned to a single route as they are swept by, going through an increasing list of the angles until the vehicle capacity or distance constraints are exceeded. Rectangular coordinates for each demand point are required in order to evaluate the polar coordinates. A customer is chosen at random and the ray from the origin through the customer is "swept" either clockwise or counter-clockwise. Customers are assigned to a given vehicle as they are "swept" until the capacity constraint for that vehicle is reached. A new vehicle is then selected and the sweep continues with assignments now being made to the new vehicle. The "refine" phase checks for improvement which could result from resequencing of customers within a route and reassignment of customers between routes. The procedure is repeated twice, once in the direction of increasing angular coordinates and once in the direction of decreasing angular coordinates. In most cases, the two procedures produce different routes and consequently different minimum total distances. The best approximate solution is the one that has the smallest value. According to Turner and Vu [61], the main disadvantages of this procedure are as follows:

1. It applies only to single-depot problems

2. The computer time increases quadratically with the average number of sites per route if the total number of sites remain relatively constant

3. The second phase requires a TSP procedure to solve each route individually

The main advantages of this procedure are (1) little computer time is required to solve large problems with small numbers of sites per route, and (2) it is quite simple to program.

The Christofides and Eilon ([13] pp. 332-333) two-phase method begins with a minimal insertion cost heuristic for inserting customers into emerging routes. The following scores are calculated for all the unrouted customers:

$$\delta_r = C_{0r} + \lambda X_{ri_h} \quad (\lambda \geq 1) \tag{2.23}$$

where X_{ri_h} indicates an unrouted customer on the route R_h. Then a feasible customer $x_r{*}$ is inserted into route R_h where

$$\delta_r{*} = \min \{\delta_r\}, \; x_r \text{ unrouted and feasible} \tag{2.24}$$

At each step, route R_h is optimized by using the Lin-Kernighan r-optimal method. In the second phase, a customer is designated in each of the routes formed in Phase 1. Beginning with the K routes that join the depot to i_k, k = 1,2,...,K, the remaining customers are inserted using a rule based on the cost of inserting a customer into alternative routes.

Cheshire, Malleson, and Noccache [11] presented a technique which is a dual heuristic because it retains local optimality at each step while gradually approaching feasibility. This procedure, where solutions are built up by retaining feasibility while gradually approaching optimality, is in contrast with other VRP approaches which are primal heuristic. However, the main features of the proposed algorithm are initial schedule building, the construction of a complete but infeasible schedule, and feasibility enforcement. An initial schedule is generated by one delivery per vehicle route. At each step of construction of the initial schedule, the next delivery which is farthest from the depot and from those deliveries are already included in the initial schedule is chosen. The total number of vehicle routes must be estimated either by the schedule or by the algorithm. The complete schedule is built up by including deliveries one at a time, but before a new delivery is included in the partial schedule, the existing partial schedule is locally optimized until it is impossible to gain an improvement by repositioning any delivery already included.

The cost function for a delivery on a route is made up of a time and a penalty function. The penalty function is a sum of terms, each proportional to the additional degree of violation of any constraint caused by the inclusion of the delivery into the existing partial schedule. The Lagrangian multipliers are initially set to some low values, then, when all deliveries have been included in the schedule, the Lagrangian multipliers associated with each violated constraint are increased in value and the total schedule is adjusted by single delivery repositioning until 1-opt is again achieved. Next, the cost reduction in each vehicle route is checked. This process is repeated until a feasible total schedule is achieved.

The Gillett and Johnson algorithm [27], an extension of the Gillett and Miller [26] "sweep" algorithm, is a two-stage procedure. During the first stage the assignment of locations to depots are determined and during the second stage, several single depot VRP's are solved. For any location i,t'(i) and t''(i) are considered to be the closest and the second closest depot to i. Then, based on the value of r(i), where

$$r(i) = d_{i,t'(i)}/d_{i,t''(i)} \text{ for all } i \tag{2.25}$$

locations are ranked in an increasing value of r(i). Based on this ranking, the nodes that are relatively close to a depot are considered first and assignment of nodes then starts from the list of r(i) until a cluster is constructed around every depot. Say that two nodes, j and k, are already assigned to a depot t and then a new node i is inserted between j and k on a route linked to i, an additional distance $d_{ji} + d_{ij} - d_{jk}$, which represents a part of the total distance (or costs), will be created. The sweep algorithm, however, is utilized to construct and sequence a route in the cluster around the depot independently.

2.2.2.4 Lagrangian Relaxation Heuristic. According to Bodin [7], Stewart and Golden presented a heuristic algorithm that considers the customer demands explicitly. This procedure treats the capacity constraints by moving them into the objective function and then imposing a penalty when demand on a route exceeds capacity. The mathematical formulation of this VRP is

$$\text{Minimize:} \quad \sum_{k=1}^{N} \sum_{i=1}^{N} \sum_{j=1}^{N} C_{ij} X_{ijk} \tag{2.26}$$

$$\text{Subject to:} \quad \sum_{i=1}^{N} \sum_{j=1}^{N} d_i X_{ijk} \le Q, \quad k = 1,2,\ldots,M \tag{2.27}$$

$$X = \{X_{ijk}\} \in S^* \tag{2.28}$$

$$X_{ijk} = \begin{cases} 0 \\ 1 \end{cases} \tag{2.29}$$

where

C_{ij} = cost or distance of moving from i to j,

d_i = demand at point i,

Q = vehicle capacity,

S* = the set of all M-TSP solutions.

Then the Lagrangian problem associated with this VRP is

$$\text{Minimize:} \quad \sum_{k=1}^{M} \sum_{i=1}^{N} \sum_{j=1}^{N} C_{ij} X_{ijk} + \sum_{k=1}^{M} \lambda_k \sum_{i=1}^{N} \sum_{j=1}^{N} d_i X_{ijk} \tag{2.30}$$

$$\text{Subject to:} \quad \sum_{i=1}^{N} \sum_{j=1}^{N} d_i X_{ijk} \geq \sum_{i=1}^{N} \sum_{j=1}^{N} d_i X_{ijr} \tag{2.31}$$

for all r = k + 1 and k = 1,2,...,M-1 and

$$X = \{X_{ijk}\} \in S* \tag{2.32}$$

$\lambda_k \geq 0$ (is the penalty route failure).

Constraint (2.31) is redundant, however, the effect is to assign the largest demand route (number 1), the second largest (number 2), and so on. The Lagrangian problem is solved for $x(\lambda)$ each time that λ is varied. Hence, the procedure is heuristic due to the fact that the VRP is solved approximately and not exactly. Also, the exact procedure might not give an optimal solution to the VRP since there may be a duality gap between the objective value for the "best" $x(\lambda)$ and the optimal solution to the VRP.

Stewart and Golden [55] also proposed a newer heuristic algorithms for the VRP which makes use of the lagrangian relaxation to transform the VRP into a M-TSP. The new formulation suggested by the authors is

$$\text{Minimize} \quad \sum_{k=1}^{M} \sum_{i,j=1}^{N} C_{ij} X_{ijk} + \sum_{k=1}^{M} \lambda_k \left(\sum_{i,j=1}^{N} \mu_i X_{ijk} - Q\right) \tag{2.33}$$

$$\text{Subject to:} \quad \sum_{i,j=1}^{N} \mu_i X_{ijk} - \sum_{i,j=1}^{N} \mu_i X_{ijr} \geq 0, \quad r = k + 1 \tag{2.34}$$

$$k = 1,2,\ldots,M-1$$

$$X = \{X_{ijk}\} \in S^*, \qquad \text{for all } i,j,k \tag{2.35}$$

$$\lambda_k \geq 0, \qquad \text{for all } k$$

where λ_k can be thought of as a penalty for each demand on route k in excess of vehicle capacity. The penalties for larger demand routes are considered to be higher than the small demand routes; however, the arc exchange procedure is used to solve this heuristic algorithm. The key to the algorithm is in the selection of values for the Lagrangian multipliers (λ_k, k = 1,2,...,M). Only λ_1 is set at a positive level (all other λ are zero), and the value of λ_1 is increased at each iteration until the 3-opt procedure produces a feasible solution to the original VRP. However, λ_1 is the multiplier associated with the first and largest demand route. Usually, a better solution is generated when λ_1 is applied to each route that is infeasible. Then the objective function becomes

$$\text{Minimize:} \quad \sum_{k=1}^{M} \sum_{i,j=1}^{N} C_{ij} X_{ijk} + \lambda_1 \sum_{k \in S} \left(\sum_{i,j} \mu_i X_{ijk} - Q\right) \tag{2.36}$$

where

$$S = \{k \mid \sum_{i,j} \mu_i X_{ijk} > Q\}. \tag{2.37}$$

2.3 Exact Solutions to the VRP

The VRP formulation as an integer program is actually the one presented previously (2.7) - (2.14), as originally formulated by Golden et al. [31].

Balinski and Quandt [5] formulated the VRP as an integer program where it is a representative of a cluster-first, route-second approach to the VRP in which demand points are first assigned to the vehicle clusters and then each vehicle is routed over the demand points assigned to it to determine a delivery sequence. The formulation is

$$\text{Minimize:} \quad \sum_{j=1}^{M} C_j \, Z_j \tag{2.38}$$

$$\text{Subject to:} \quad \sum_{j=1}^{M} a_{ij} \, Z_j = 1 \qquad i = 1,2,\ldots,N \tag{2.39}$$

$$\sum_{i=1}^{N} a_{ij} \, d_i \le Q \qquad j = 1,\ldots,M \tag{2.40}$$

$$Z_j = \begin{cases} 0 \\ 1 \end{cases} \qquad j = 1,\ldots,M \tag{2.41}$$

where decision variables Z_j are binary and specify whether or not cluster j is used; $a_{ij} = 1$, if demand point i is assigned to clusters j and 0, otherwise, these coefficients are fixed and defined for each cluster j; d_j is the demand of station j and Q is the capacity of any vehicles in the fleet which are assumed to be homogeneous; C_j is the minimum cost of any vehicle route passing the demand points i assigned to the j^{th} cluster (i.e., the demand point i with $a_{ij} = 1$).

Foster and Ryan [24] proposed an integer programming formulation of the VRP which is solved using the Revised Simplex Method. This method is strictly primal in that both feasibility and integrality are withheld at all stages. An integer programming formulation of the VRP with a planning horizon of more than one day is extended to incorporate the linear constraints. The suggested formulation is

$$\text{Minimize:} \quad \sum_{j \in J} (V + m_j)\, X_j \qquad (2.42)$$

$$\text{Subject to:} \quad \sum_{j \in J} a_{ij}\, X_j = 1 \qquad i = 1,2,\ldots,N \qquad (2.43)$$

where X_j is 0 or 1, represents the probability that route j is in the schedule; V is the mileage equivalent cost of each vehicle; m_j is the total mileage of route j; $a_{ij} = 1$ if delivery i is made on route j; and N is the number of deliveries. J is the set of all feasible routes.

Fisher and Jaikumer [23] have formulated a heuristic approach which describes the VRP as consisting of two interrelated components: the TSP and the Generalized Assignment Problem. Finally, Christofides, Mingozzi, and Toth [12] have formulized the VRP as a dynamic program problem.

2.4 Stochastic Vehicle Routing Problem (SVRP)

2.4.1 An Overview

The SVRP has attracted less attention in the literature than the deterministic VRP. However, the SVRP is a problem of interest to operation researchers due to the wide applicability of such a model in real life situations. The SVRP is to design a set of routes starting from and eventually returning to a central depot and to deliver products to a fixed number of demand points such that the capacity constraints, probabilistic customer demands, and the duration of the routes are satisfied.

Tillman [57] proposed a modification of the Clarke and Wright procedure for multidepot delivery and collection problems having probabilistic demands that are poisson distributed. The objective function of the delivery problem for a given number of stop points on a proposed route is

$$\text{Min } E[\text{cost}] = \underset{R}{\text{Min}} \left\{ \int_0^R C_1(D)h(D)dD + \int_R^\infty C_2(D)h(D)dD \right\} \tag{2.42}$$

where the first expression from the right indicates the cost of not hauling enough commodity to satisfy all customer demands on a route and the second expression from the right represents the cost of hauling excess commodity on the route that is not needed.

The value of R determined for each route is the load assigned to the truck for that route. Notations are

$C_1(D) = \begin{cases} \text{cost of hauling excess commodity on the route that is not needed, or} \\ \text{cost of completing scheduled route and having unfilled capacity} \end{cases}$

$C_2(D) = \begin{cases} \text{cost of not hauling enough commodity to satisfy all the demands on the route, or} \\ \text{cost of filling truck prior to completing the scheduled route} \end{cases}$

$$D = d_1 + d_2 + d_3, \ldots, + d_n \tag{2.43}$$

d_i = the probabilistic demand for the i^{th} stop

$f_i(d_i)$ = probability density function of the random variable d_i

$h(D)$ = probability density function of D

Golden and Stewart [30] have extended Tillman's SVRP in a different way considering only a single depot problem. In this technique the locations on the route are n_1, n_2, $n_3,\ldots,n_k$, and it is assumed that all vehicles have the same capacity Q and that the total demand for all locations is

$$X = dn_1 + dn_2 + dn_3 + ,\ldots, + dn_k \tag{2.44}$$

where dn_i is the demand at location i which is described by the independent poisson distribution with mean and variance λn_i. Then

$$E(X) = Var(X) = \lambda n_1 + \lambda n_2 + ,\ldots, + \lambda n_k \tag{2.45}$$

for that route. Using the central limit theorem and approximating with normal distributions, then $\mu = \lambda n_1 + ,\ldots, + \lambda n_k$, and $\sigma = \sqrt{\mu}$. However, by considering the definitions of primary and secondary errors from Chapter V, Section 5.5.5, one can write,

$$P(X \geq Q) = P\text{ (primary error)} = P(Z \geq \frac{Q - \mu}{\sqrt{\mu}}) \geq (1 - \alpha) \tag{2.46}$$

and

$$P(X \leq aQ) = P\text{(secondary error)} = P(Z \leq \frac{aQ - \mu}{\sqrt{\mu}}) \geq \alpha \tag{2.47}$$

where Q is the truck capacity and $0 < a, \alpha \leq 1$.

Assuming that μ is nearly the same for most of K routes, then an artificial capacity $\bar{\mu}$, as the vehicle capacity, can be used along with the λ_{n_i} as demand points and the "saving" approach of Clarke and Wright to obtain a fixed set of routes. Therefore, the following problem is the one that must be solved:

Minimize: expected total cost (2.48)

Subject to:

1. a fixed set of routes,
2. satisfaction of customer demands,
3. P(primary error) $\geq (1 - \alpha)$, and
4. vehicle capacity is obeyed.

where $0 \leq \alpha \leq 1$.

Golden and Yee [29] extended this work to several other demand distributions and presented a more comprehensive view of vehicle routing.

In a later article, Yee and Golden [66] presented a dynamic programming approach to determine the driver operating strategies when demands on a route are probabilistic. Specifically, after delivery of goods to a demand point on a fixed route, which has already been determined by the Clarke and Wright procedure, the driver is faced with the decision of whether to return to the depot to replenish the supply. However, the optimal decision is based on whether the remaining supply of goods in the vehicle is greater or less than some critical value which must take into account the following criteria:

1. The probabilistic demands on the remaining portion of the route, and

2. The distances between the remaining customers.

Cook and Russell [18] have successfully treated a large routing problem with timing constraints and stochastic travel times and demands. The authors approach the problem by generating a deterministic solution using the MTOUR algorithm and then testing these routes via simulation to demonstrate that they are effective; however, the stochastic nature of the problem is not explicitly considered in the route generation stage. The basic procedure for the generation of travel times and pickup times is based on the development of the multiple regression equations for each random variable so that the point estimates can be calculated. The regression equation for the intra-city transit times is derived by employing the euclidean distance and average speed limit as the independent variables. The service time (pickup time) is considered to be a function of two independent variables: number of containers and the total capacity of the containers. Based on these assumptions, the second regression equation for pickup times is determined.

2.5 Interactive Heuristic Approach

Interactive vehicle routing is a general approach in which a high degree of human interaction is incorporated into the problem solving process [7]. It is a method of building routes which is under the control of the decision maker who uses an interactive computer program to indicate the results of decisions made in terms of cost, time, distance, or vehicle utilization.

Krolak et al. [41] proposed a man-machine approach which takes the following steps:

1. The decision maker defines the problem,

2. The computer organizes the data and then gives several alternative solutions using a sophisticated heuristic technique,

3. The decision maker creates another solution and the computer compares the solutions using a pictorial display, and

4. The decision maker attempts to modify the computer solution.

This process continues until the decision maker is satisfied with the solution.

Stacy [53] has developed an interactive vehicle routing algorithm which creates various logical stages in the trail of project design, data collection, validation, and staff training. Also, Waters [63] has developed an interactive vehicle routing algorithm which is able to introduce the concept to new or trainee schedulers. Some of the advantages and disadvantages of the interactive procedure as summarized by Turner and Vu [61] are given below:

Advantages of the Interactive Vehicle Routing problem:

1. Human interaction is allowed, yielding better solutions
2. The computer helps organize the data for the decision maker

Disadvantages of the Interactive Vehicle Routing problem:

1. It is time consuming for both the decision maker and computer
2. The solution is usually suboptimal
3. The concepts require trained or experienced personnel

Park [49] presented a heuristic algorithm to determine vehicle routes for the multiple-vehicle, single-depot case where conflicting multiple objective functions are treated explicitly. This heuristic approach is based on the ideas of Gillett and Miller [26], Clarke and

Wright [17], and Williams [64], which were discussed earlier. Park's heuristic approach implies different upper bounds for the constraints on vehicle travel distance and are based on the preemptive goal priority structure.

Allison [2] developed an interactive model to solve the Workload Balancing Vehicle Routing Problem (WBVRP) using multiple criteria analysis. This research is concerned with the VRP in order to minimize the total distance of the whole delivery system and the deviation in workload among the routes. The workload elements are defined to be (1) total distance on time spent driving, and (2) the total weight or amount of goods delivered. The WBVRP is a multiple criteria optimization problem and is concerned with the deterministic customer demand and travel time.

2.6 Summary

This chapter has presented a literature review of the VRP, multiple-depot VRP, and SVRP. As indicated, the single-depot, multiple-vehicle, node routing problem has attracted the attention of most researchers whereas little research has been conducted on the SVRP and multidepot, multiple-vehicle, node routing problem.

As previously discussed, solution techniques for the VRP are divided into two main categories: those which solve the problem optimally and those which solve the problem heuristically. Optimal seeking procedures are only practical for solving small-sized problems while heuristic techniques are the most promising tools for solving large-scale problems. For this reason, a great deal of attention has been given to the Clark and Wright [17] heuristic approach and its modifications and as well as to the Gillett and Miller [26] approach.

In this research, the heuristic methods were categorized into four groups: tour building heuristics, tour improvement methods, two-phase methods, and Lagrangian relaxation heuristic approaches. It should be noted that there are relatively few interactive approaches that solve the VRP and only two procedures that are capable of handling the VRP in a multiple objective environment [2, 49]. Moreover, each of the procedures described, with the exception of [2, 49], has a single objective cost, time, or distance minimization.

CHAPTER III

CHANCE-CONSTRAINED PROGRAMMING (CCP)

3.1 Introduction

When the parameters in a mathematical programming model are presumed to be random variables rather than constants, a stochastic programming problem must be solved. These problems involve risk if the probability distributions of the random variables are known, or involve uncertainty if the distribution of at least one random variable is unknown. The difficulties of dealing with risk and uncertainty in programming problems have been discussed in the literature since the 1950's.

Chance-Constrained Programming has been introduced into stochastic programming literature mainly through the exposition of Charnes and Cooper [10]. These authors suggest the E, V, and P models. In the E model the expected value of the objective function is to be maximized; in the V model the objective is to minimize a generalized mean square error; and in the P model the purpose is to maximize the probability that C'X does not exceed a given constant $C_o'X_o$. In this technique, a decision vector X has to be selected such that each constraint is satisfied at least α $(0 \leq \alpha \leq 1)$ percent of the time. The topic of CCP is perhaps best introduced by first exhibiting an ordinary LP problem in its general form as:

Minimize $Z = C'X$

Subject to: $AX \leq b,\ X \geq 0$

where A is an m x n matrix of constraints and C' is a 1 x n matrix while b is an m x 1 matrix. A chance-constrained formulation would replace the above problem with one of the following kind:

$$\text{Minimize} \quad Z = \sum_{j=1}^{n} C_j X_j \tag{3.1}$$

Subject to:

$$P\left(\sum_{j=1}^{n} a_{ij} X_j \leq b_i\right) \geq \alpha_i \qquad \text{for all } i = 1,\ldots,m \tag{3.2}$$

$$X_j \geq 0 \qquad \text{for all } j \tag{3.3}$$

where "P" means probability and $0 \leq \alpha_i \leq 1$. The parameters of this problem are the objective function coefficients C_j, the coefficients a_{ij}, and the right hand side values b_i. Practically, a_{ij}, C_j, and b_i are not necessarily constant, and in general, some or all of their elements are random variables. The vector α is a set of constants that are probability measures which determines the extent of the constraint violations.

The value of the objective function, Z, will depend upon the values of C_j, b_i, and a_{ij} when they are random variables having known distribution functions. The "E Model" [10] that optimizes the expected value of the objective function may be used only when the C_j are random variables. When one or more a_{ij} and/or one or more b_i are random variables, the "E" Model cannot be applied. In this case, the surrogate models of stochastic programming such as CCP and stochastic programming with recourse may be applied [33].

Problem (3.1)-(3.3) seek a solution vector X that satisfies the CC (3.2) and minimizes the value of Z. Many authors [20, 22, 28, 33, 62], as well as Charnes and Cooper [10], have offered methods to convert such stochastic models into their deterministic models (not necessarily linear) which can be solved by the existing mathematical programming techniques.

3.2 Development of Deterministic Equivalents

In this section, two cases where constraint requirements, b_i, and input-output coefficients, a_{ij} are random variables having known distribution functions will be discussed. To develop the equivalent deterministic form of chance-constrained inequality (3.2), consider in more detail a constraint

$$\sum_{j=1}^{n} a_{ij} X_j \leq b_i \qquad \text{for all } i = 1,\ldots,m \tag{3.4}$$

in the following two situations:

1. b_i are independently distributed random variables with mean μ_{b_i} and variance $\sigma^2_{b_i}$, and
2. a_{ij} are random variables with mean $\mu_{a_{ij}}$, and variance $\sigma^2_{a_{ij}}$, a_{ij} are distributed independently of a_{ik} $(j \neq k)$.

The random variables b_i and a_{ij} are assumed to be normal random variables. However, Sections 3.2.1 and 3.2.2 deal with the development of the equivalent deterministic form of the probabilistic constraints considering the above situations, respectively. A comprehensive survey for the development of the equivalent deterministic forms for other situations is discussed in [68].

3.2.1 Constraint Requirements Random Variable, b_i

In particular, constraint (3.4) is

$$P(\sum_{j=1}^{n} a_{ij} X_j \leq b_i) \geq \alpha_i \quad \text{for all } i = 1,2,\ldots,m \tag{3.5}$$

where $(1 - \alpha_i)$ denotes the allowable "risk" that a random variable will be chosen such that

$$\sum_{j=1}^{n} a_{ij} X_j \geq b_i.$$

The equivalent deterministic form of the constraint (3.5) for normally distributed random variables of b_i is

$$\sum_{j=1}^{n} a_{ij} X_j \geq \mu_{b_i} + \sigma_{b_i} K_{\alpha_i}.$$

Where $K_{\alpha i}$ is a standard normal value such that $\Phi(K_\alpha) = \alpha$ and Φ represents the cumulative distribution function for the standard normal [28, pp. 275]. Hence, by solving the problem

$$\text{Minimize:} \quad \sum_{j=1}^{n} C_j X_j \tag{3.6}$$

$$\text{Subject to:} \quad \sum_{j=1}^{n} a_{ij} X_j \geq \mu_{b_i} + \sigma_{b_i} K_{\alpha_i}, \; X_j \geq 0, \; i = 1,\ldots,m \tag{3.7}$$

one can obtain the optimal solution to problem (3.1)-(3.3).

3.2.2 Input-Output Coefficients

Random Variable, a_{ij}

To deal with one constraint at a time, drop the subscript i and let μ_j and σ_j^2 be the mean and variance of a_j. Then given X, the mean value of $\sum_j a_j X_j$ is $M = \sum_j \mu_j X_j$ and its standard deviation is $S = (\sum \sigma_j^2 X_j^2)^{\frac{1}{2}}$ (assuming M and S exist). Now if there exists a constant τ such that

$$P((\sum_j a_j X_j - M)/ S) \leq \tau) = \alpha \tag{3.8}$$

then constraint $P(\sum_j a_j X_j \leq b) \geq \alpha$ is equivalent to the nonstochastic constraint

$$M + \tau S \leq b. \tag{3.9}$$

Of course, M and S contain the unknown X_j. Constraint (3.9) is generally nonlinear in nature, as can be seen when it is written in the following form:

$$\sum_{j=1}^{n} \mu_{a\,ij} X_j + \tau \, (\sum_{j=1}^{n} \sigma_{a\,ij}^2 \, x_j^2)^{\frac{1}{2}} \leq b_i \text{ for all } i, \tag{3.10}$$

where $\tau = -K_\alpha$ and K_α is as previously defined [28]. The constraint (3.9) can be substituted for (3.2) if M and S exist and τ is independent of X_j. This will be the case if the distribution of $(\sum_j a_{ij} X_j - M)/S$ is the same as $(a_{ij} - \mu_{a\,ij})/\sigma_{a\,ij}$. This case is true when a_{ij} are normally distributed or random variables a_{ij} all have the same stable distribution with parameters U_{ij} and V_{ij}, respectively. Vajda [62, p. 84] claims that stable distributions have the common property of being completely determined by the specifications of two parameters U and V (not necessarily the mean and standard deviation) where U is real and $V > 0$. The convolution of any $n \times m$ distributions $F((a_{ij} - U_{ij})/V_{ij})$ for $i = 1,2,\ldots, m$ and $j = 1,2,\ldots, n$ is of the form $F((a - U)/V)$. Poisson, binomial, chi-square, and normal distributions belong to this family.

The deterministic constraint (3.10) can be written as:

$$\sum_{j=1}^{n} \mu_{a_{ij}} X_j - b_i \leq - \tau(\sum_{j=1}^{n} \sigma_{ij}^2 \quad x_{ij}^2)^{\frac{1}{2}} \tag{3.11}$$

When $\alpha > 0.5$ then τ is negative, which requires that one square both sides of the inequality to obtain

$$b_i^{\,2} + (\sum_{j=1}^{n} \mu_{a\,ij} X_j)^2 - 2b_i \sum_{j=1}^{n} \mu_{a_{ij}} \quad X_j - \tau^2 \sum_{j=1}^{n} \sigma_{ij}^2 \, X_j^2 \leq 0 \tag{3.12}$$

which is a quadratic constraint.

When the random variables a_{ij} or b_i are not normally distributed, the development outlined above does not apply. Goicoechea, Hansen, and Duckster [28, pp. 276-281] developed the equivalent deterministic forms of the probabilistic constraints which consist of random variables other than normal, such as exponential, uniform, and beta random variables.

3.3 Summary

The most common method for dealing with random variables in programming models is through certainty equivalents which can be achieved by transforming the CCP problems into nonstochastic problems. The equivalent deterministic forms of the probabilistic constraints are either linear, as shown in (3.7), or nonlinear, as shown in (3.10) or (3.12).

CHAPTER IV

MULTIPLE OBJECTIVE GOAL PROGRAMMING MODELS

4.1 Introduction

Goal Programming draws upon the highly developed and tested techniques of linear programming, yet provides a solution to a complex system of competing objectives. This technique can handle problems having a single goal with multiple subgoals as well as problems having multiple goals and subgoals [69]. The basic concept of GP involves incorporating some managerial goals into the constraints of the model.

Goal Programming technique was originally introduced by Charnes and Cooper [10] in early 1961 for a linear model. The GP has been extended into many areas, including the capital budgeting problem [38] and aggregate production and manpower planning [1]. Lee [42] applied goal programming to problems in production planning, financial decisions, academic planning, and medical care, to mention a few. A GP model is useful for the following three types of analysis [38]:

1. To determine the input (resource) requirements to achieve a set of goals

2. To determine the degree of attainment of defined goals with given resources and

3. To provide the optimum solution under varying inputs and priority structures.

In general, a goal programming problem can be categorized as:

1. Linear goal programming (LGP) problem,
2. Linear integer goal programming (LIGP) problem, and
3. Nonlinear goal programming (NGP) problem.

The LGP problem is discussed in section 4.2 and LIGP has been delayed until Chapter VI. The importance of NGP has been recognized by many authors including Griffith [32], Ignizio [38], and Lee and Wynne [43].

4.2 General Model

The general model of LGP can be stated as follows [38]:

$$\text{Minimize} \quad Z = \sum_{j=1}^{k} \sum_{i=1}^{m} P_j \left(W_{ij}^{-} n_i + W_{ij}^{+} p_i\right) \tag{4.1}$$

Subject to:

$$\sum_{r=1}^{n} a_{ir} X_r + n_i - p_i = b_i \qquad i = 1,2,\ldots,m \tag{4.2}$$

$$f_i\ (X) \begin{pmatrix} \leq \\ \geqq \end{pmatrix} b_i \qquad i = m + 1,\ldots,s \tag{4.3}$$

$$n_i\ p_i = 0 \qquad i = 1,2,\ldots,m \tag{4.4}$$

$$X_r \geq 0\ ,\ n_i\ ,\ p_i \geq 0, \qquad i = 1,2,\ldots,m \tag{4.5}$$

$$r = 1,2,\ldots,n$$

where

P_j = is the preemptive priority weight assigned to goal j

W_{ij}^{+}, W_{ij}^{-} = are numerical (differential) weights assigned to the deviational variables of goal i at a given priority level j

p_i = represents the positive deviations or surplus variables from goal j (overachievement)

n_i = represents the negative deviations or slack variables from goal j (underachievement)

b_i = is the i^{th} target level where $i = 1,2,\ldots,m$

a_{ir} = is the technological coefficient of X_r in goal i.

The sets of goal constraints are those with $i = 1,2,\ldots,m$ and the sets of rigid constraints are those with $i = m + 1,\ldots,s$.

There are three basic approaches to problems characterized by a priori set of goals: Preemptive Goal Programming, Archimedean (or Non-preemptive) Goal Programming, and Multigoal Programming. These three approaches are discussed in more detail below.

4.2.1 Preemptive Goal Programming

The objective function for preemptive goal programming is often written as [69]:

$$\sum_{i=1}^{k} P_i \; f_i \; (n_i, \; p_i) = P_1 \; f_1 \; (n_1, \; p_1) + \ldots + P_k \; f_k \; (n_k, \; p_k). \qquad (4.6)$$

The purpose of preemptive goal programming is the minimization of $f_i \; (n_i, \; p_i)$, one by one, in the order of their (preemptive) priorities. Functions f_i are typically linear functions of deviational variables; i.e., $f_i = n_i$, $f_i = p_i$, $f_i = n_i + p_i$, or $f_i = [W_i \; n_i + (1 - W_i) \; p_i]$, and so on. The summation above is redundant and meaningless; however, it is prevalent in the literature and thus cannot be ignored.

4.2.2 Archimedean Goal Programming

The objective function is to minimize [69]:

$$\sum_{i=1}^{k} W_i \, [f_i \, (n_i, \, p_i)]^r = W_1 \, [f_1 \, (n_1, \, p_1)]^r + \ldots + W_k \, [f_k \, (n_k, \, p_k)]^r. \qquad (4.7)$$

All the objective functions are considered simultaneously and their weights W_i are not preemptive. Powers r can take any value, but usually $r = 1,2$, or ∞.

4.2.3 Multigoal Programming

The purpose of Multigoal Programming is to minimize $[f_1 \, (n_1, \, p_1), \ldots, \, f_k \, (n_k, \, p_k)]$, as in Multiobjective Linear Programming [3, 36, 37, 69]. However, it is not necessary to write the objective function in terms of an aggregate preference function. Other variants of GP and multiobjective linear programming can be found in [28, 37, 43].

4.3 Solution Methods for Linear Goal Programming Problems

The most commonly used solution techniques for solving LGP problems are, partitioning goal programming [4], multiphase linear goal programming [28, 31], and interactive sequential goal programming [43].

Arthur and Ravindran [4] have modified the method of solution of GP problems with preemptive weights into a procedure called the Partitioning GP algorithm. In the partitioning procedure constraints should be categorized such that a nested series of GP problems can be formed:

$$SP_1 \subseteq SP_2 \subseteq \ldots SP_k \subseteq \ldots$$

In general, SP_k stands for the k^{th} subproblem which consists of those goal constraints assigned to the first k priority levels and the corresponding terms in the objective function of the k^{th} subproblem. The solution procedure starts with the smallest subproblem, SP_1, which consists of all goal constraints assigned to this priority, the system constraint, and the corresponding terms in the objective function. The main idea after obtaining the optimal solution of each subproblem is to examine the optimal tableau for alternate optimal solutions. If no alternative solutions exists, then the solution is optimal for the GP problem. In this case, the value of decision variables of the optimal tableau is substituted into the goal constraints of the lower priority levels (if any exist) to calculate their attainment levels. If alternate optimal solutions exist, the optimization process is continued after augmenting the next set of goal constraints and their objective function terms into the optimal tableau. However, the process of addition of goal constraints and objective function terms continues until no alternate optimum solution exists for one of the subproblems, or until all priority levels have been considered in the optimization process. The linear independency between each pair of individual variables guarantees no need for the dual simplex operation on the updated tableau at the beginning of each optimization process. Most important, when the optimal solution to the SP_{k-1} is obtained, before the addition of new goal constraints (for SP_k) into the optimal tableau, one should delete all nonbasic columns which have a negative value for $(Z_j - C_j)$ from the optimal tableau of SP_{k-1} for further consideration.

The multiphase (or modified simplex) algorithm is simply a refinement of the well-known two phase method. In this method, the basic simplex method of linear programming is utilized to minimize the deviational variables. The deviational variables are ranked according to preemptive priority factors so that during the solution process the goals are considered in order of their priorities. The weighting method is allowed to incorporate the cardinal values to goals at given priority levels [27, 44].

In most cases, the multiple objective problems cannot be optimized simultaneously because such problems involve making trade-off decisions to get the "best compromise" solution. However, Interactive Sequential GP [46] (ISGP) is a link between GP and interactive approaches which is based on the implicit assumption that the decision maker can adjust the desired goals through an interactive learning process based on the information in a set of solutions. Any iteration, say r, consists of two phases: calculation and evaluation. The Principal Solution and a set of Alternate Solutions are obtained in the calculation phase, and the evaluation phase consists of the decision maker's indication of his preference judgment about these solutions in the form of new desirable goal levels. With this new information, the process goes back to the calculation phase of the $(r + 1)^{th}$ iteration. Both linear and nonlinear problems can be solved by ISGP. Additional details concerning this procedure are given by Masud and Hwang [46].

4.4 Summary

The GP approach appears to be an appropriate solution technique in developing a model to attain multiple, competitive, and often conflicting objectives with varying priorities. However, GP is not the answer to all decision problems. In fact, there are a large number of problems that cannot be solved by this method, nor can this technique replace the subjective aspects of decision making. The application of GP for decision analysis does force the decision maker to think of goals and constraints in terms of their importance to the organization, and thus are an invaluable aid to the decision-making process.

CHAPTER V

DEVELOPMENT OF THE STOCHASTIC VEHICLE ROUTING PROBLEM

5.1 Development

This chapter is concerned with the development of the VRP within the framework of stochastic programming and addresses the goal programming formulation of the problem in which priorities of various goals are identified. The sensitivity of time and truck capacity upon the probability of route failures is analyzed and some necessary theorems are proven.

The SVRP examined in this research is concerned with the multiple-vehicle, single-depot node routing problem in which restrictions are placed on the total travel and unload times of each vehicle route. Alternatively, restriction can be imposed by the Decision Maker (DM) regarding total elapsed time of each vehicle route instead of specifying each type of time constraint individually. For example, the DM may specify that routes must require less than 10 hours time for both travel and unload times.

The time constraints arise in many real life problems such as industrial refuse collection and scheduled mail pick-up and delivery problem [18]. The importance of time constraints has been recognized by many authors including Cheshire et al. [11], Fisher and Jaikumer [23], Evans et al. [21], and Williams [64].

Some of the work that has been published in the literature deals with stochastic elements within the framework of linear, single objective, and heuristic approaches [18, 29, 30, 57]. As mentioned in Chapter I, this author considers two major stages for the stochastic VRP, the Route Construction Stage (RCS) and Route Improvement Stage (RIS). The following is a brief description of these two stages.

5.1.1 The Route Construction Stage

The RCS of the SVRP consists of problem formulation and partitioning a set of stations into feasible sets of vehicle routes. The presence of nonlinearities in the equivalent deterministic form of the SVRP generally make the problem more complex than similarly-sized VRPs. For this reason, only heuristic methods for solving SVRP are considered in this study.

The RCS of the SVRP consists of the following steps:

1. Problem formulation in which objective functions and probabilistic constraints are identified,
2. Transformation of the above stochastic problem into an equivalent deterministic form, and
3. Partitioning of a set of stations into feasible subsets using an appropriate heuristic approach.

The RIS of the problem consists of problem formulation and sequencing of stations on each vehicle route to meet the customer's and decision maker's requirements.

5.1.2 The Route Improvement Stage

This stage of the problem is important because the final results depend on the decision maker's policy and the way in which goals and their relevant priorities are listed. Obviously, the goal priority structure of all objectives must be carefully stated because the achievement of one goal may result in a very poor achievement of the remaining goals.

The RIS of the SVRP consists of the following steps:

1. Problem formulation in which goals and probabilistic constraints are identified,

2. Transformation of the above stochastic problem into an equivalent deterministic form, and

3. GP formulation of RIS in which priorities of various goals are identified.

Here, it should be noted that the mathematical formulation of the RIS, and in turn the derivation of its equivalent deterministic form, is easier to formulate than the RCS of the problem. Also, it is necessary to develop mathematically the objective functions of the RIS of the problem in terms of the decision variables. For these reasons, the problem formulation of the RIS of the problem is given in Sections 5.4 and 5.5 and the problem formulation of the RCS of the problem is delayed until Section 5.6.

5.2 Notations

The following notations are utilized in this research:

NS = number of stations on a vehicle route, excluding the central depot

TNS = the total number of stations to be served, excluding the central depot

C = the total cost of each vehicle route

C_{ij} = the travel cost of moving from station i to j, $C_{ii} = \infty$

d_i = the demand at station i (i = 1,2,...,TNS), is a random variable having a known distribution function

d_{ij} = the distance between station i and station j, and $d_{ii} = \infty$

D = a 1 x NS vector with components of d_i

I = set of stations on a vehicle route, including the central depot, 0 stands for central depot

J = I − {0}

M = the mean of travel time on a vehicle route

NV = number of vehicles

n(i) = a set of negative deviations for constraint (i)

p(i) = a set of positive deviations for constraint (i)

P(.) = stands for probability of (.)

Q = a vehicle capacity

$\bar{Q}$ = the artificial capacity of a vehicle

R = the variance-covariance (dispersion) matrix for customer demand

S = a set of feasible solutions for each vehicle route

S_{NV} = a set of feasible solutions for NV trucks

SS = the safety stock

t_i = the unload time at station i, (i = 1,2,...,TNS), is a random variable having a known distribution function

t_{ij} = the travel time from station i to station j, is a random variable having a known distribution function (i = 0,1,2,...,TNS) and (j = 0,1,...,TNS) and $t_{ii} = \infty$

E = a 1 x NS vector with components of t_i

T1 = the maximum total travel time allowed on each vehicle route

T2 = the maximum total unload time allowed on each vehicle route

TR_k = a predetermined maximum total travel time allowed for the k^{th} vehicle route

TT = the total time required to complete a vehicle route

UT_k = a predetermined maximum total unload time allowed for the k^{th} vehicle route

V = the variance-covariance (dispersion) matrix for travel time

W = the variance-covariance (dispersion) matrix for unload time

X_{ij} = decision variables, 1 if a truck goes from station i to station j, 0 otherwise

G = a 1 x NS vector with components X_{ij}

X_{ijk} = decision variables, 1 if the k^{th} truck goes from station i to station j, 0 otherwise

α,β,γ,η = the predetermined level of constraint violations, where $0 < \alpha \le 1$, $0 < \beta \le 1$, $0 < \gamma \le 1$, and $0 < \eta \le 1$

α_k,β_k,η_k = the predetermined level of constraint violation of the k^{th} truck, where $0 < \alpha_k \le 1$, $0 < \beta_k \le 1$, and $0 < \eta_k \le 1$

μ_{d_i} = the mean of the demand at station i

$\sigma^2_{d_i}$ = the variance of the demand at station i

μ_{t_i} = the mean of the unload time at station i

$\sigma^2_{t_i}$ = the variance of the unload time at station i

$\mu_{t_{ij}}$ = the mean of the travel time between station i and j

$\sigma^2_{t_{ij}}$ = the variance of the travel time between station i and j

Ψ = a constant value

U = a 1 x NS vector with components μ_{t_i}

H = a 1 x NS vector with components μ_{d_i}

$\bar{T}1$ = a target level for travel time for each vehicle route

$\bar{T}2$ = a target level for unload time for each vehicle route

T = a TNS $\times$ TNS matrix with components of t_{ij}

5.3 Assumptions

The following assumptions were considered in this model building:

1. The demand at each destination is a random variable having a known distribution function

2. The unload time at each destination is a random variable having a known distribution function

3. The travel time from one station to another is a random variable having a known distribution function

4. The commodity to be transported is homogeneous

5. All vehicles have the same capacity

6. The shortest distance between two stations is considered to be euclidian

7. The maximum allowable total travel time of each vehicle route is T1

8. The maximum allowable total unload time of each vehicle route is T2

9. $\alpha, \beta, \gamma, \eta, \alpha_k, \beta_k$, and η_k are predetermined.

5.4 Route Improvement Stage: Problem Formulation

5.4.1 Time Constraints

The formulation to be presented is based on the previous assumptions and notations. The basic form of the problem is typified by the situation in which deliveries are made from a central depot to the destinations by NV vehicles. All goods as well as the NV trucks are assumed to be available for delivery at an arbitrary time zero. This formulation allows different predetermined conditions on each vehicle route. T1 is considered to be the maximum value of the total travel time on each vehicle route for $100(1 - \alpha)\%$ of the time. On the other hand, the maximum value of the total unload time on each vehicle route is T2 for $100(1 - \beta)\%$ of the time. In general, when the travel time between any link and unload time at each station are deterministic, then the total elapsed time on the vehicle route is

$$\text{Total elapsed time} = \sum_{i=0}^{NS} (t_{i,i+1} + t_i). \tag{5.1}$$

where 0 stands for depot, $t_0 = 0$, and NS + 1 is defined to be 0.

The DM is generally interested in minimizing the total travel cost and total travel and unload times of each vehicle route to target levels

C, $\bar{T}1$, and $\bar{T}2$, respectively. Additionally, other criteria such as customer satisfaction may attract the attention of the DM in order to satisfy the customer's requirements. Hence, the multiple objective SVRP can be formulated having the following goals and constraints. However, the method of calculation of $\bar{T}1$ and $\bar{T}2$ are delayed until Section 5.7.

Problem A

Goals:

1. Minimize total travel cost or distance of each vehicle route

$$C = \sum_{i \in I} \sum_{\substack{j \in I \\ i \neq j}} C_{ij} \quad X_{ij} \tag{5.2}$$

2. Minimize total travel and total unload times of each vehicle route to the target levels which are set to be $\bar{T}1$ and $\bar{T}2$, respectively.

3. Maximize the dependency conditions such that station r follows station s.

Constraints:

$$P(\sum_{i \in I} \sum_{\substack{j \in I \\ i \neq j}} t_i \; X_{ij} \leq T2) \geq (1 - \beta) \tag{5.3}$$

$$P(\sum_{i \in I} \sum_{\substack{j \in I \\ i \neq j}} t_{ij} \; X_{ij} \leq T1) \geq (1 - \alpha) \tag{5.4}$$

$$G = [X_{ij}] \in S \tag{5.5}$$

where t_{ij} and t_i are independent random variables and assumed to be normally distributed with means $\mu_{t_{ij}}$ and μ_{t_i}, and variances $\sigma^2_{t_{ij}}$ and $\sigma^2_{t_i}$, respectively. The T1 and T2 introduced above indicate suitable upper limits on travel and unload times for each vehicle route.

5.4.2 Demand Constraint

In the field of VRP, one of the difficulties which occurs in the application of mathematical programming is that the demands at stations are not constants but are either fluctuating or of uncertain values. However, used on the previous assumptions and notations, a probabilistic demand can be handled by using the concept of probabilistic constraints. For example, suppose that η is the maximum allowable probability that a vehicle route will fail due to the total probabilistic demand exceeding the truck capacity, then:

$$P\left(\sum_{i \in I} \sum_{\substack{j \in I \\ i \neq j}} d_i \; X_{ij} \leq Q\right) \geq (1 - \eta) \qquad (5.6)$$

where d_i's are independent random variables representing demand at location i and assumed to be normally distributed with mean μ_{d_i} and variance $\sigma^2_{d_i}$, and Q is the truck capacity. The existence of the probabilistic customer demand forces the decision maker to minimize the safety stock (unused capacity) which is $Q - \bar{Q}$. However, the method of calculation of $\bar{Q}$ is delayed until Section 5.7. By incorporating this idea, Problem A can be modified to a more general form of a multicriteria SVRP, as presented in Problem B:

Problem B

Goal:

1. Minimize total travel cost or distance of each vehicle route

$$C = \sum_{i \epsilon I} \sum_{\substack{j \epsilon I \\ i \neq j}} C_{ij} X_{ij} \tag{5.7}$$

2. Minimize total travel and unload times of each vehicle route to the target levels which are set to be T1 and T2, respectively

3. Minimize the route safety stock

$$SS = Q - \overline{Q} \tag{5.8}$$

4. Maximize the dependency conditions such that station r follows station s

Constraints:

$$P(\sum_{i \epsilon I} \sum_{\substack{j \epsilon I \\ i \neq j}} t_i X_{ij} \leq T2) \geq (1 - \beta) \tag{5.9}$$

$$P(\sum_{i \epsilon I} \sum_{\substack{j \epsilon I \\ i \neq j}} t_{ij} X_{ij} \leq T1) \geq (1 - \alpha) \tag{5.10}$$

$$P(\sum_{i \epsilon I} \sum_{\substack{j \epsilon I \\ i \neq j}} d_i X_{ij} \leq Q) \geq (1 - \eta) \tag{5.11}$$

$$G = [X_{ij}] \epsilon S. \tag{5.12}$$

It is worth noting that when travel and unload times are probabilistic, then Problem A should be chosen for the purpose of the GP formulation of the RIS of the problem. On the other hand, when customer demand and travel and unload times are probabilistic, then Problem B should be used to construct the GP formulation of the RIS of the problem.

Problems A and B are used in Section 5.8 for the purpose of the GP formulation of the SVRP.

5.5 Development of the Deterministic Forms for the Set of Constraints of Problem B

It has been shown [39] that it is possible to deal effectively with random variables in the constraint set of a stochastic programming problem. When random variables appear in the constraint set, deterministic equivalents must be derived to replace the original chance-constrained inequalities. Therefore, this section is devoted to the development of the equivalent deterministic form of the constraints of Problem B. The following subsections consider each constraint separately:

1. Deterministic form for unload time constraints
2. Deterministic form for travel time constraints
3. Deterministic form for demand constraints

5.5.1 Deterministic Form for Unload Time Constraint

The first constraint of Problem B is called the unload time constraint which is written as

$$P(\sum_{i \in I} \sum_{\substack{j \in I \\ i \neq j}} t_i X_{ij} \leq T2) = (1 - \beta). \tag{5.13}$$

Now one can consider an NS component vector $E = (t_1, t_2, \ldots, t_{NS})$ as a multivariate normal with mean vector $U = (\mu_{t_1}, \mu_{t_2}, \ldots, \mu_{t_{NS}})$ and variance-covariance matrix

$$W = \begin{pmatrix} \sigma^2_{t_1} & & & & \\ & \ddots & & & \\ & & \sigma^2_{t_i} & & \\ & & & \ddots & \\ & & & & \sigma^2_{t_{NS}} \end{pmatrix} \tag{5.14}$$

where the t_i's are independent random variables corresponding to the unload time at each of the i stations. Let $Z = (Z_1, Z_2, \ldots, Z_{NS})$, be a vector of NS elements where each component of Z is

$$Z_i = \sum_{j \in I} X_{ij} \qquad \forall i \in I,\ i \neq j \tag{5.15}$$

According to multivariate statistical analysis [9, 34, 39], the linear combination Z'E is univariate normal with mean Z'U and variance Z'WZ. Therefore

$$P(Z'E \leq T2) = P((Z'E - Z'U)/(Z'WZ)^{1/2} \leq (T2 - Z'U)/(Z'WZ)^{1/2}) = (1 - \beta). \tag{5.16}$$

The above inequality exists if and only if

$$N((T2 - Z'U)/(Z'WZ)^{1/2}) = (1 - \beta)$$

or

$$(T2 - Z'U)/(Z'WZ)^{1/2} = N^{-1}(1 - \beta).$$

Finally, the deterministic form of (5.13) is

$$Z'U + N^{-1}(1 - \beta)(Z'WZ)^{1/2} = T2 \tag{5.17}$$

where $Z'U$ and $(Z'WZ)^{1/2}$ are the mean and standard deviation of the unload time on the vehicle route and $N^{-1}(1 - \beta)$ is the normalized deviate corresponding to the required probability

$$\beta = \frac{1}{(2\pi)^{1/2}} \int_{-\infty}^{N^{-1}(1-\beta)} \exp(-x^2/2)\, dx. \tag{5.18}$$

If the required probability $(1 - \beta) = 0.95$, then $N^{-1}(1 - \beta) = 1.645$ from the normal table [34, pp. 592-593]. The evaluated deterministic form given above is generally nonlinear in nature, which can be seen when it, (5.17), is written in the following form:

$$\sum_{i\in I} \sum_{\substack{j\in I \\ i\neq j}} \mu_{t_i} X_{ij} + N^{-1}(1 - \beta)\Big(\sum_{i\in I} \sum_{\substack{j\in I \\ i\neq j}} \sigma^2_{t_i} X^2_{ij}\Big)^{1/2} = T2. \tag{5.19}$$

5.5.2 Deterministic Form for

Travel Time Constraint

The second constraint of problem B (inequality (5.10)), called the travel time constraint, is written as

$$P\Big(\sum_{i\in I} \sum_{\substack{j\in I \\ i\neq j}} t_{ij} X_{ij} \leq T1\Big) = (1 - \alpha). \tag{5.20}$$

Suppose that $\mu_{t_{ij}}$ and $\sigma^2_{t_{ij}}$ are the mean and variance of t_{ij} where t_{ij}'s are independent random variables corresponding to the travel time from station i to station j and are assumed to be normally distributed. Then, one can consider a travel time vector $T = (t_{ij})\ \forall i,j,\ i \neq j$, as a multivariate normal with mean $M = (\mu_{t_{ij}})\ \forall i,j,\ i \neq j$, and variance-covariance matrix V. Let $G = [X_{ij}],\ \forall i,j,\ i \neq j$, be a vector of all elements X_{ij} which are arranged in the same order as the elements of $T = (t_{ij}),\ \forall i,j,\ i \neq j$. According to multivariate statistical analysis, the linear combination G'T is univariate normal with mean G'M and variance G'VG. Hence:

$$P(G'T \leq T1) = P((G'T - G'M)/(G'VG)^{\frac{1}{2}} \leq (T1 - G'M)/(G'VG)^{\frac{1}{2}}). \qquad (5.21)$$

Thus, by the definition of a cumulative distribution function, i.e., $N_X(x) = P(X \leq x)$ where X is a random variable, one can write

$$N((T1 - G'M)/(G'VG)^{\frac{1}{2}}) = (1 - \alpha).$$

Then, after similar calculations as previously considered

$$G'M + N^{-1}(1 - \alpha)\ (G'VG)^{\frac{1}{2}} = T1 \qquad (5.22)$$

where G'M and $(G'VG)^{\frac{1}{2}}$ are the mean and standard deviation of the travel time on a vehicle route and $N^{-1}(1 - \alpha)$ is the normal deviate as described in Section 5.5.1. Constraint (5.22) is generally nonlinear in terms of X_{ij} as shown below:

$$\sum_{i \in I} \sum_{\substack{j \in I \\ i \neq j}} \mu_{t_{ij}}\, X_{ij} + N^{-1}(1 - \alpha)\left(\sum_{i \in I} \sum_{\substack{j \in I \\ i \neq j}} \sigma^2_{t_{ij}}\, X^2_{ij}\right) = T1^{\frac{1}{2}}. \qquad (5.23)$$

5.5.3 Evaluation of the Total Time TT

This section determines the nature of the total elapsed time of each vehicle route in terms of the decision variables, X_{ij}, when decision maker needs to use TT = T1 + T2 as a criterion. Using equations (5.17) and (5.22), one can write:

$$TT = T1 + T2 = (X'M + Z'U) + [N^{-1}(1 - \alpha)(X'VX)^{\frac{1}{2}} + N^{-1}(1 - \beta)\ (Z'WZ)^{\frac{1}{2}}]. \tag{5.24}$$

If $\alpha < 0.5$ and $\beta < .5$, then $N^{-1}(1 - \alpha)$ and $N^{-1}(1 - \beta) > 0$. Therefore, the above constraint can be written as:

$$(TT - X'M - Z'U)^2 = (N^{-1}(1 - \alpha)\ (X'VX)^{\frac{1}{2}} + N^{-1}(1 - \beta)\ (Z'WZ)^{\frac{1}{2}})^2 \tag{5.25}$$

which is a quadratic function in terms of X_{ij}.

5.5.4 Deterministic Form for the Demand Constraint

This section is devoted to derivation of the deterministic form of the demand constraint where d_i's are independent random variables corresponding to the demand at station i and are assumed to be normally distributed. The demand constraint can be rewritten in the form

$$P(\sum_{i \in I} \sum_{\substack{j \in I \\ i \neq j}} d_i\ X_{ij} \leq Q) = (1 - \eta). \tag{5.26}$$

If μ_{d_i} and $\sigma^2_{d_i}$ are the mean and variance of d_i, then one can consider an NS component vector $D = (d_1, d_2, \ldots, d_{NS})$ as a multivariate normal with mean vector $H = (\mu_{d_1}, \mu_{d_2}, \ldots, \mu_{d_{NS}})$ and variance-covariance matrix R. Let $Y = (Y_1, Y_2, \ldots, Y_{NS})$ be a vector of NS elements where each component of Y is

$$Y_i = \sum_{j \in I} X_{ij} \qquad \forall i \in I,\ i \neq j. \tag{5.27}$$

Again, according to multivariate statistical analysis, the linear combination Y'D is univariate normal with mean Y'H and variance Y'RY. Hence,

$$P(Y'D \leq Q) = P((Y'D - Y'H)/(Y'RY)^{\frac{1}{2}} \leq (Q - Y'H)/(Y'RY)^{\frac{1}{2}})$$

or

$$((Q - Y'H)/(Y'RY)^{\frac{1}{2}}) = N^{-1}(1 - \eta). \tag{5.28}$$

Then

$$Q = Y'H + N^{-1}(1 - \eta)(Y'RY)^{\frac{1}{2}} \tag{5.29}$$

where Y'H and $(Y'RY)^{\frac{1}{2}}$ are the mean and standard deviation of the demand on the vehicle route and $N^{-1}(1 - \eta)$ is the normal deviate as previously described. This deterministic form is also nonlinear in nature, as can be seen when it is written in expanded form as

$$\sum_{i \in I} \sum_{\substack{j \in I \\ i \neq j}} \mu_{d_i} X_{ij} + N^{-1}(1 - \eta)(\sum_{i \in I} \sum_{\substack{j \in I \\ i \neq j}} \sigma_{d_i}^2 X_{ij}^2)^{\frac{1}{2}} = Q. \tag{5.30}$$

5.5.5 Safety Stock and Surplus in Term of the Decision Variables X_{ij}

Before the development of the safety stock and surplus in term of the decision variables begins, it is necessary to define the following terms. Golden and Stewart [30, pp. 253-254] have defined the primary and secondary error as follows:

Primary Error

A primary error occurs when a vehicle cannot satisfy the demands of the customers on the route to which it has been assigned.

Secondary Error

A secondary error occurs when a vehicle returns to the central depot after satisfying the demands on its route with more than 100(1 − a) percent of its original load, where $0 \leq a \leq 1$.

The primary error requires an additional trip to the central depot which causes additional cost and service delay. On the other hand, the existence of the secondary error is a waste of load and unload times and in some cases it is a waste of products (i.e., perishable goods).

By considering a delivery problem with one central depot, NS demand points, and a vehicle capacity Q with probabilistic demand d_i for the i^{th} customer, then by appealing to the Central Limit Theorem, one can argue that the total route demand, TD, is approximately normally distributed where

$$TD = d_1 + d_2 + \ldots + d_{NS}. \tag{5.31}$$

If d_i are poisson distributed with mean and variance μ_{d_i}, then the mean and variance of the total demand on the route are, respectively:

$$E(TD) = \mu_{d_1} + \mu_{d_2} + \ldots + \mu_{d_{NS}} \tag{5.32}$$

and

$$\sigma^2(TD) = E(TD). \tag{5.33}$$

The primary and secondary errors of each vehicle route are, respectively

$$P(TD \geq Q) = P(z \geq (TD - E(TD))/\sigma(TD)) \tag{5.34}$$

and

$$P(TD \leq aQ) = P(z \leq (aQ - E(TD))/\sigma(TD)) \text{ where } 0 < a \leq 1 \tag{5.35}$$

and z is a unit normal deviate.

One may treat the primary error as $P(TD \geq Q) \leq \eta$ where $0 < \eta < 0.5$ by incorporating the concept of an artificial capacity of a truck, $\bar{Q}$, where $\bar{Q} < Q$,

$$P(z \geq (Q - \bar{Q})/\sigma(TD)) = \eta \tag{5.36}$$

which has an equivalent deterministic form of

$$Q = \bar{Q} + N^{-1}(1 - \eta)\ \sigma(TD). \tag{5.37}$$

Notice that (5.37) is similar to (5.29). The quantity of $Q - \bar{Q}$ is called the route safety stock (SS) which is protection against the primary error. Since $(1 - \eta) > 0.5$, then $N^{-1}(1 - \eta)$ is positive. Using the SS as a criteria, one will have the following expression in terms of the decision variables X_{ij}:

$$SS = N^{-1}(1 - \eta)\Big(\sum_{i\epsilon I}\ \sum_{\substack{j\epsilon I\\ i\neq j}} \sigma^2_{d_i}\ X^2_{ij}\Big)^{\frac{1}{2}}. \tag{5.38}$$

The secondary error as defined in (5.35)

$$P(TD \leq aQ) = P(z \leq (aQ - E(TD))/\sigma(TD)) = \gamma$$

with the following equivalent deterministic form:

$$aQ = E(TD) + N^{-1}(\gamma)\sigma(TD). \qquad (5.39)$$

In this case, the quantity Q – aQ is the extra number of units of products carried on the vehicle routes if trucks are loaded up to their Q capacity. To minimize the carrying of these units on the vehicle route one may use the following nonlinear objective function:

$$Q - \sum_{\substack{i\epsilon I \\ i \neq j}} \sum_{j\epsilon I} \mu_{d_i} X_{ij} - N^{-1} (\gamma)(\sum_{\substack{i\epsilon I \\ i \neq j}} \sum_{j\epsilon I} \sigma_{d_i}^2 X_{ij}^2)^{\frac{1}{2}}. \qquad (5.40)$$

The GP formulation of the RIS of the problem is delayed until Section 5.8.

5.6 Route Construction Stage: Problem Formulation

The problem formulation of this stage is divided into two sections according to the type of criteria that is to be minimized. The criteria to be considered are:

1. Total cost (or distance) as presented in problem C, and
2. Total time as presented in problem D.

When cost (or distance) is considered as a criteria, the objective function is linear in terms of the decision variables X_{ijk}. On the other hand, when total time is considered to be minimized, the objective function becomes nonlinear in terms of the decision variables X_{ijk}.

5.6.1 Using Cost as a Criterion

Problem C

$$\text{Minimize:} \quad C = \sum_{k=1}^{NV} \sum_{i=0}^{TNS} \sum_{\substack{j=0 \\ i \neq j}}^{TNS} c_{ij} X_{ijk} \tag{5.41}$$

Subject to:

$$P\Big(\sum_{i=0}^{TNS} \sum_{\substack{j=0 \\ i \neq j}}^{TNS} t_{ij} X_{ijk} \leq TR_k\Big) \geq (1 - \alpha_k), \; k = 1, \ldots, NV \tag{5.42}$$

$$P\Big(\sum_{i=1}^{TNS} \sum_{\substack{j=0 \\ i \neq j}}^{TNS} t_i X_{ijk} \leq UT_K\Big) \geq (1 - \beta_k), \; k = 1, \ldots, NV \tag{5.43}$$

$$P\Big(\sum_{i=1}^{TNS} \sum_{\substack{j=0 \\ i \neq j}}^{TNS} d_i X_{ijk} \leq Q\Big) \geq (1 - \eta_k), \; k = 1, \ldots, NV \tag{5.44}$$

$$X = [X_{ijk}] \in S_{NV}. \tag{5.45}$$

5.6.2 Using Total Time as a Criterion

Problem D

$$\text{Minimize:} \quad \sum_{k=1}^{NV} [TR_k + UT_k] \tag{5.46}$$

Subject to:

$$P\Big(\sum_{i=0}^{TNS} \sum_{\substack{j=0 \\ i \neq j}}^{TNS} t_{ij} X_{ijk} \leq TR_k\Big) \geq (1 - \alpha_k), \; k = 1, \ldots, NV \tag{5.47}$$

$$P\Big(\sum_{i=1}^{TNS} \sum_{\substack{j=0 \\ i \neq j}}^{TNS} t_i X_{ijk} \leq UT_K\Big) \geq (1 - \beta_k), \; k = 1, \ldots, NV \tag{5.48}$$

$$P\left(\sum_{i=1}^{TNS} \sum_{\substack{j=0 \\ i \neq j}}^{TNS} d_i X_{ijk} \leq Q\right) \geq (1 - \eta_k), \; k = 1, \ldots, NV \tag{5.49}$$

$$X = [X_{ijk}] \in S_{NV} \tag{5.50}$$

where S_{NV} is the set of all feasible solutions to the NV travelling salesman problem, and α_k, β_k, and η_k (k = 1,..., NV) are the probability of constraint infeasibility on the K^{th} route by violating the predetermined levels TR_k, UT_k, and Q, respectively. The process of transformation of the above probabilistic constraints to their equivalent deterministic forms are similar to those shown previously. Without loss of generality and for the sake of space, the deterministic forms of Problems C and D are shown in following section.

However, the equivalent deterministic form of each situation is completely different and, hence, each requires a different solution technique.

5.6.3 Equivalent Deterministic Forms of Problems C and D of the RCS of the Problem

The equivalent deterministic form of Problem C is

Problem E

$$\text{Minimize:} \quad C = \sum_{k=1}^{NV} \sum_{i=0}^{TNS} \sum_{\substack{j=0 \\ i \neq j}}^{TNS} c_{ij} X_{ijk} \tag{5.51}$$

Subject to:

$$\sum_{\substack{i=0 \\ i\neq j}}^{TNS} \sum_{j=0}^{TNS} \mu_{t_{ij}} X_{ijk} + N^{-1}(1-\alpha_k)\Big(\sum_{\substack{i=0 \\ i\neq j}}^{TNS} \sum_{j=0}^{TNS} \sigma^2_{t_{ij}} X^2_{ijk}\Big)^{1/2} \le TR_k \qquad (5.52)$$

$$\sum_{\substack{i=1 \\ i\neq j}}^{TNS} \sum_{j=0}^{TNS} \mu_{t_i} X_{ijk} + N^{-1}(1-\beta_k)\Big(\sum_{\substack{i=1 \\ i\neq j}}^{TNS} \sum_{j=0}^{TNS} \sigma^2_{t_i} X^2_{ijk}\Big)^{1/2} \le UT_k \qquad (5.53)$$

$$\sum_{\substack{i=1 \\ i\neq j}}^{TNS} \sum_{j=0}^{TNS} \mu_{d_i} X_{ijk} + N^{-1}(1-\eta_k)\Big(\sum_{\substack{i=1 \\ i\neq j}}^{TNS} \sum_{j=0}^{TNS} \sigma^2_{d_i} X^2_{ijk}\Big)^{1/2} \le Q \qquad (5.54)$$

$$X = [X_{ijk}] \in S_{NV} \text{ and } k = 1,2,\ldots,NV. \qquad (5.55)$$

Similarly, the equivalent deterministic form of Problem D is

Problem F

Minimize:

$$\sum_{k=1}^{NV} \Big\{\Big[\sum_{\substack{i=0 \\ i\neq j}}^{TNS} \sum_{j=0}^{TNS} \mu_{t_{ij}} X_{ijk} + \sum_{\substack{i=1 \\ i\neq j}}^{TNS} \sum_{j=0}^{TNS} \mu_{t_i} X_{ijk}\Big] + \Big[N^{-1}(1-\alpha_k)\Big(\sum_{\substack{i=0 \\ i\neq j}}^{TNS} \sum_{j=0}^{TNS} \sigma^2_{t_{ij}} X^2_{ijk}\Big)^{1/2}\Big]$$

$$+ \Big[N^{-1}(1-\beta_k)\Big(\sum_{\substack{i=1 \\ i\neq j}}^{TNS} \sum_{j=0}^{TNS} \sigma^2_{t_i} X^2_{ijk}\Big)^{1/2}\Big]\Big\} \qquad (5.56)$$

Subject to:

$$\sum_{\substack{i=1 \\ i\neq j}}^{TNS} \sum_{j=0}^{TNS} \mu_{d_i} X_{ijk} + N^{-1}(1-\eta_k)\Big(\sum_{\substack{i=1 \\ i\neq j}}^{TNS} \sum_{j=0}^{TNS} \sigma^2_{d_i} X^2_{ijk}\Big)^{1/2} \le Q \qquad (5.57)$$

$$X = [X_{ijk}] \in S_{NV}. \qquad k = 1,2,\ldots, NV \qquad (5.58)$$

The deterministic forms of problems C and D will be called "E" and "F" type problems.

The most important characteristic of the "E" type problem is that while the decision maker desires to minimize total expected cost or distance of the whole delivery system, he can also restrict the travel and unload times of each single vehicle route.

On the other hand, the "F" type problem deals with the minimization of total elapsed time of the whole delivery system with no restriction on the travel and unload times of each vehicle route.

It is the decision maker's responsibility to determine which of these two problems are suitable for the delivery system. These two problems offer two benefits. One benefit is that they support the decision maker, quantitatively, in attempting to make a good decision. The decision maker may be willing to change some or all upper bounds, TR_k, UT_k, of each vehicle route when a previous solution was not favorable. The second benefit is that the "F" type problem allows the decision maker to minimize total elapsed time of the whole delivery system without time restriction put on each single vehicle route.

Since normal distributions are symmetrical about their mean, the objective function (5.59) is identical with the expected value reformulation of (5.56) when $\alpha_k = 0.5$ and $\beta_k = 0.5$ for all $k \epsilon (1,2,...,NV)$, which yields:

$$\text{Minimize:} \quad E(\sum_{i=0}^{TNS} \sum_{\substack{j=0 \\ i \neq j}}^{TNS} [t_{ij} X_{ijk} + \sum_{i=1}^{TNS} \sum_{\substack{j=0 \\ i \neq j}}^{TNS} t_i X_{ijk}]) \qquad (5.59)$$

from which one obtains the expression

$$\text{Minimize:} \quad \sum_{k=1}^{NV} [\sum_{\substack{i=0 \\ i \neq j}}^{TNS} \sum_{j=0}^{TNS} \mu_{t_{ij}} X_{ijk} + \sum_{\substack{i=1 \\ i \neq j}}^{TNS} \sum_{j=0}^{TNS} \mu_{t_i} X_{ijk}]. \tag{5.60}$$

when $\alpha_k = \beta_k = \eta_k = 0.5$ for all $k \in (1,2,\ldots,NV)$ then problem (5.56) – (5.58) is converted to the following problem.

$$\text{Minimize} \quad \sum_{k=1}^{NV} [\sum_{\substack{i=0 \\ i \neq j}}^{TNS} \sum_{j=0}^{TNS} \mu_{t_{ij}} X_{ijk} + \sum_{\substack{i=1 \\ i \neq j}}^{TNS} \sum_{j=0}^{TNS} \mu_{t_i} X_{ijk}] \tag{5.61}$$

$$\text{Subject to:} \quad \sum_{\substack{i=1 \\ i \neq j}}^{TNS} \sum_{j=0}^{TNS} \mu_{d_i} X_{ijk} \leq Q \qquad k = 1,\ldots,NV$$

$$X = [X_{ijk}] \in S_{NV}.$$

When $0.5 < (1 - \alpha_k)$, $(1 - \beta_k)$, $(1 - \eta_k) < 1$ for all $k \in (1,2,\ldots,NV)$, which is reasonable to assume, $N^{-1}(1 - \alpha_k)$, $N^{-1}(1 - \beta_k)$, and $N^{-1}(1 - \eta_k)$ are all greater than zero, then (5.56) and (5.58) are convex, since $(X'VX)^{1/2}$ is a convex function. Therefore, problem (5.56) – (5.58) is a convex programming problem.

The corresponding deterministic form of the previous problem, shown in (5.56) – (5.58), is shown below with $q_k = N^{-1}(1 - \alpha_k)$ and $f_k = N^{-1}(1 - \beta_k)$ and $e_k = N^{-1}(1 - \eta_k)$ for $k = (1,2,\ldots,NV)$:

$$\text{Maximize:} \quad -\sum_{k=1}^{NV} [\sum_{\substack{i=0 \\ i \neq j}}^{TNS} \sum_{j=0}^{TNS} \mu_{t_{ij}} X_{ijk} + \sum_{\substack{i=1 \\ i \neq j}}^{TNS} \sum_{j=0}^{TNS} \mu_{t_i} X_{ijk}]$$

$$-\sum_{k=1}^{NV} [q_k (\sum_{i=0}^{TNS} \sum_{\substack{j=0 \\ i \neq j}}^{TNS} \sigma_{t_{ij}}^2 X_{ijk}^2)^{½} + f_k (\sum_{i=1}^{TNS} \sum_{\substack{j=0 \\ i \neq j}}^{TNS} \sigma_{t_i}^2 X_{ijk}^2)^{½}] \tag{5.62}$$

$$\text{Subject to: } \sum_{i=1}^{TNS} \sum_{\substack{j=0 \\ i \neq j}}^{TNS} \mu_{d_i} X_{ijk} + e_k (\sum_{i=1}^{TNS} \sum_{\substack{j=0 \\ i \neq j}}^{TNS} \sigma_{d_i}^2 X_{ijk}^2)^{½} \leq Q,$$

$$k = 1,\ldots,NV \tag{5.63}$$

$$AX \leq b \tag{5.64}$$

where constraint set $AX \leq b$ is equivalent to constraints (2.8), (2.9), (2.11), and (2.12) where N = TNS and depot is node 0.

Next, consider the following quadratic programming problem which is (5.62) with R_{1_k} and R_{2_k} inserted as shown:

$$\text{Maximize: } -\sum_{k=1}^{NV} [\sum_{i=0}^{TNS} \sum_{\substack{j=0 \\ i \neq j}}^{TNS} \mu_{t_{ij}} X_{ijk} + \sum_{i=1}^{TNS} \sum_{\substack{j=0 \\ i \neq j}}^{TNS} \mu_{t_i} X_{ijk}] \tag{5.65}$$

$$-\sum_{k=1}^{NV} [(q_k/2R_{1_k})(\sum_{i=0}^{TNS} \sum_{\substack{j=0 \\ i \neq j}}^{TNS} \sigma_{t_{ij}}^2 X_{ijk}^2) + (f_k/2R_{2_k})(\sum_{i=1}^{TNS} \sum_{\substack{j=0 \\ i \neq j}}^{TNS} \sigma_{t_i}^2 X_{ijk}^2)]$$

$$\text{Subject to: } \sum_{i=1}^{TNS} \sum_{\substack{j=0 \\ i \neq j}}^{TNS} \mu_{d_i} X_{ijk} + e_k (\sum_{i=1}^{TNS} \sum_{\substack{j=0 \\ i \neq j}}^{TNS} \sigma_{d_i}^2 X_{ijk}^2)^{½} \leq Q,$$

$$k = 1,2,\ldots,NV \tag{5.66}$$

$$AX \leq b. \tag{5.67}$$

R_{1_k} and R_{2_k} are the positive parameters defined as follows

$$R_{1_k} = (\sum_{i=0}^{TNS} \sum_{\substack{j=0 \\ i \neq j}}^{TNS} \sigma_{t_{ij}}^2 X_{ijk}^2)^{½} \tag{5.68}$$

$$R_{2_k} = \left(\sum_{i=1}^{TNS} \sum_{\substack{j=0 \\ i \neq j}}^{TNS} \sigma_{t_i}^2 X_{ijk}^2\right)^{1/2} \tag{5.69}$$

for all $k \in (1,2,\ldots,NV)$. The following theorem provides the optimum solution for problem (5.56) – (5.58).

Theorem (5.1)

If an optimal solution $\hat{X}(R_{1_k}, R_{2_k})$ of problem (5.65) – (5.67) satisfies the conditions R_{1_k} and R_{2_k} as shown in (5.68) and (5.69), then $\hat{X}(R_{1_k}, R_{2_k})$ is also the optimal solution vector of problem (5.62) – (5.64).

Proof

For problem (5.62) – (5.64) the Lagrangian function is

$$FI(X,\lambda,\mu) = -\sum_{k=1}^{NV} \left[\sum_{i=0}^{TNS} \sum_{\substack{j=0 \\ i \neq j}}^{TNS} \mu_{t\,ij} X_{ijk} + \sum_{i=1}^{TNS} \sum_{\substack{j=0 \\ i \neq j}}^{TNS} \mu_{t_i} X_{ijk}\right] -$$

$$\sum_{k=1}^{NV} \left[q_k \left(\sum_{i=0}^{TNS} \sum_{\substack{j=0 \\ i \neq j}}^{TNS} \sigma_{t_{ij}}^2 X_{ijk}^2\right)^{1/2} + f_k \left(\sum_{i=1}^{TNS} \sum_{\substack{j=0 \\ i \neq j}}^{TNS} \sigma_{t_i}^2 X_{ijk}^2\right)^{1/2}\right]$$

$$+ \sum_{k=1}^{NV} \lambda_k \left[Q - \sum_{i=1}^{TNS} \sum_{\substack{j=0 \\ i \neq j}}^{TNS} \mu_{d_i} X_{ijk} - e_k \left(\sum_{i=1}^{TNS} \sum_{\substack{j=0 \\ i \neq j}}^{TNS} \sigma_{d_i}^2 X_{ijk}^2\right)^{1/2}\right] + \mu(b - AX). \tag{5.70}$$

The vector $\hat{X} = [X_{ijk}]$ for all i,j,k is an optimal solution for (5.62) – (5.64), if $\hat{X}$ and $\hat{\lambda} = (\lambda_1,\ldots,\lambda_k)$ and $\hat{\mu}$ satisfy the following conditions:

$$\partial FI/\partial X_{ijk} = -\sum_{k=1}^{NV} [\sum_{\substack{i=0 \\ i\neq j}}^{TNS} \sum_{j=0}^{TNS} \mu_{t_{ij}} + \sum_{\substack{i=1 \\ i\neq j}}^{TNS} \sum_{j=0}^{TNS} \mu_{t_i}]$$

$$-\sum_{k=1}^{NV} q_k (\sum_{\substack{i=0 \\ i\neq j}}^{TNS} \sum_{j=0}^{TNS} \sigma^2_{t_{ij}} X_{ijk})/(\sum_{\substack{i=0 \\ i\neq j}}^{TNS} \sum_{j=0}^{TNS} \sigma^2_{t_{ij}} X^2_{ijk})^{\frac{1}{2}}$$

$$-\sum_{k=1}^{NV} f_k (\sum_{\substack{i=1 \\ i\neq j}}^{TNS} \sum_{j=0}^{TNS} \sigma^2_{t_i} X_{ijk})/(\sum_{\substack{i=0 \\ i\neq j}}^{TNS} \sum_{j=0}^{TNS} \sigma^2_{t_{ij}} X^2_{ijk})^{\frac{1}{2}}$$

$$+\sum_{k=1}^{NV} \lambda_k (\sum_{\substack{i=1 \\ i\neq j}}^{TNS} \sum_{j=0}^{TNS} \mu_{d_i} - e_k(\sum_{\substack{i=1 \\ i\neq j}}^{TNS} \sum_{j=0}^{TNS} \sigma_{d_i} X^2_{ijk})/(\sum_{\substack{i=1 \\ i\neq j}}^{TNS} \sum_{j=0}^{TNS} \sigma^2_{d_i} X^2_{ijk})^{\frac{1}{2}})$$

$$-A\mu \quad \begin{cases} \leq 0 & \text{for } X_{ijk} = 0 \\ = 0 & \text{for } X_{ijk} > 0 \end{cases} \qquad (5.71)$$

$$\partial FI/\partial \lambda_k = Q - \sum_{\substack{i=1 \\ i\neq j}}^{TNS} \sum_{j=0}^{TNS} \mu_{d_i} X_{ijk} - e_k (\sum_{\substack{i=1 \\ i\neq j}}^{TNS} \sum_{j=0}^{TNS} \sigma^2_{d_i} X^2_{ijk})^{\frac{1}{2}}$$

$$\begin{cases} \geq 0 \text{ for all } \lambda_k = 0 \\ = 0 \text{ for } \lambda_k > 0 \end{cases}$$

$$k = 1,2,\ldots,NV \qquad (5.72)$$

$$\partial FI/\partial \mu = +(b - AX) \quad \begin{cases} \geq 0 \text{ for } \mu = 0 \\ = 0 \text{ for } \mu > 0 \end{cases} \qquad (5.73)$$

Similarly, the Kuhn-Tucker conditions of the problem (5.65) – (5.67) can be written as shown below. If $\hat{X} = [X_{ijk}]$ $\forall$ i,j,k and $\hat{X}$, $\hat{\lambda}$, $\hat{\mu}$ satisfy the following conditions, then it is the optimum solution of problems (5.65) – (5.67).

$$FII(X,\lambda,\mu) = -\sum_{k=1}^{NV} \Big[\sum_{i=1}^{TNS} \sum_{\substack{j=0\\ i\neq j}}^{TNS} \mu_{t_{ij}} X_{ijk} + \sum_{i=1}^{TNS} \sum_{\substack{j=0\\ i\neq j}}^{TNS} \mu_{t_i} X_{ijk}\Big]$$

$$-\sum_{k=1}^{NV} \Big[(q_k/2R_{1_k})\Big(\sum_{i=0}^{TNS} \sum_{\substack{j=0\\ i\neq j}}^{TNS} \sigma^2_{t_{ij}} X^2_{ijk}\Big)+(f_k/2R_{2_k})\Big(\sum_{i=1}^{TNS} \sum_{\substack{j=0\\ i\neq j}}^{TNS} \sigma^2_{t_i} X^2_{ijk}\Big)\Big]$$

$$-\sum_{k=1}^{NV} \lambda_k \Big[Q - \sum_{i=1}^{TNS} \sum_{\substack{j=0\\ i\neq j}}^{TNS} \mu_{d_i} X_{ijk} - e_k \Big(\sum_{i=1}^{TNS} \sum_{\substack{j=0\\ i\neq j}}^{TNS} \sigma^2_{d_i} X^2_{ijk}\Big)^{\frac{1}{2}}\Big] + \mu(b-AX) \qquad (5.74)$$

$$\partial FII/\partial X_{ijk} = -\sum_{k=1}^{NV} \Big[\sum_{i=0}^{TNS} \sum_{\substack{j=0\\ i\neq j}}^{TNS} \mu_{t_{ij}} + \sum_{i=1}^{TNS} \sum_{\substack{j=0\\ i\neq j}}^{TNS} \mu_{t_i}\Big]$$

$$-\sum_{k=1}^{NV} (q_k/R_{1_k})\Big(\sum_{i=0}^{TNS} \sum_{\substack{j=0\\ i\neq j}}^{TNS} \sigma^2_{t_{ij}} X_{ijk}\Big) - \sum_{k=1}^{NV} (f_k/R_{2_k})\Big(\sum_{i=1}^{TNS} \sum_{\substack{j=0\\ i\neq j}}^{TNS} \sigma^2_{t_i} X_{ijk}\Big)$$

$$+\sum_{i=1}^{NV} \lambda_k \Big(\sum_{i=0}^{TNS} \sum_{\substack{j=0\\ i\neq j}}^{TNS} \mu_{d_i} - e_k \Big(\sum_{i=0}^{TNS} \sum_{\substack{j=0\\ i\neq j}}^{TNS} \sigma^2_{d_i} X^2_{ijk}\Big)/\Big(\sum_{i=0}^{TNS} \sum_{\substack{j=0\\ i\neq j}}^{TNS} \sigma^2_{d_i} X^2_{ijk}\Big)^{\frac{1}{2}}\Big)$$

$$-A\mu \begin{cases} \leq 0 \text{ for } X_{ijk} = 0 \\ = 0 \text{ for } X_{ijk} > 0 \end{cases} \qquad (5.75)$$

$$\partial FII/\partial\lambda_k = Q - \sum_{\substack{i=1 \\ i\neq j}}^{TNS} \sum_{j=0}^{TNS} \mu_{d_i} X_{ijk} - e_k\left(\sum_{\substack{i=1 \\ i\neq j}}^{TNS} \sum_{j=0}^{TNS} \sigma^2_{d_i} X^2_{ijk}\right)^{\frac{1}{2}} \begin{cases} \geq 0, & \lambda_k = 0 \\ = 0, & \lambda_k > 0 \end{cases} \quad (5.76)$$

$$\partial FII/\partial\mu = (b - AX) \quad \begin{cases} \geq 0, & \mu = 0 \\ = 0, & \mu > 0 \end{cases} \quad (5.77)$$

Hence, if equations (5.68) and (5.69) exist, then the conditions of both problems are identical. Therefore, $\hat{X}(R_{1_k}, R_{2_k})$ is also an optimal solution of (5.62) – (5.64). Conversely, for any optimal solution $\hat{X}$ of (5.62) – (5.64), if R_{1_k}, R_{2_k} is set such that it satisfies (5.68) and (5.69), $\hat{X}$ also satisfies the condition (5.65) – (5.67).

This theorem indicates that an optimal solution to the "F" type problems exists. Since this problem is nonlinear and decision variables are in 0-1 form, seeking the optimum solution by exact procedure is not efficient. Hence, heuristic approaches are considered for solving "E" and "F" type problems.

5.7 Distributions Other Than Normal

The deterministic constraints (5.19), (5.23), and (5.30) can be replaced with some other constraints in an easier form, if the time and demand distributions are of the same special forms. There are several distributions that satisfy the following condition:

$$\sigma_i^2 = \Psi\mu_i \quad (5.78)$$

This means that the variance is some constant multiple of the mean of that distribution. Distributions such as Poisson and chi-square satisfy the above condition. The value of Ψ for these distributions are

1. Poisson $\quad \sigma_i^2 = \mu_i \quad \Psi = 1$

2. Chi-square $\quad \mu_i = v$

   $\sigma_i^2 = 2v \quad \Psi = 2$

The following theorem shows the existence of a set of deterministic linear time and demand constraints which are equivalent to the non-linear set of the time and demand constraints of the RIS problem.

Theorem 5.2

Under the following conditions:

1. The probability distributions of t_{ij} are independent and stable [62, p. 84], and $\sigma^2_{t_{ij}} = \Psi\mu_{t_{ij}}$

2. The probability distributions of t_i are independent and stable, and $\sigma^2_{t_i} = \Psi\mu_{t_i}$

3. The probability distributions of d_i are independent and stable, and $\sigma^2_{d_i} = \Psi\mu_{di}$,

then there exist values $\overline{T1}$, $\overline{T2}$, and $\overline{Q}$ such that

$$\sum_{i\epsilon I} \sum_{\substack{j\epsilon I \\ i\neq j}} \mu_{t_{ij}} X_{ij} = \overline{T1} \tag{5.79}$$

$$\sum_{i\epsilon I}\sum_{\substack{j\epsilon I\\ i\neq j}} \mu_{t_i} X_{ij} = \bar{T}2 \qquad (5.80)$$

$$\sum_{i\epsilon I}\sum_{\substack{j\epsilon I\\ i\neq j}} \mu_{d_i} X_{ij} = \bar{Q} \qquad (5.81)$$

which are equivalent to the deterministic constraint (5.23), (5.19), and (5.30), respectively.

Proof

The proof is developed only for (5.79). One can prove similarly for (5.80) and (5.81). Since decision variable X_{ij} is either zero or 1, then,

$$X_{ij} = X_{ij}^2.$$

Therefore

$$\sigma_{t_{ij}}^2 X_{ij} = \sigma_{t_{ij}}^2 X_{ij}^2$$

$$\sum_{i\epsilon I}\sum_{\substack{j\epsilon I\\ i\neq j}} \sigma_{t\,ij}^2 X_{ij} = \sum_{i\epsilon I}\sum_{\substack{j\epsilon I\\ i\neq j}} \sigma_{t\,ij}^2 X_{ij}^2$$

or

$$[\sum_{i\epsilon I}\sum_{\substack{j\epsilon I\\ i\neq j}} \sigma_{t_{ij}}^2 X_{ij}]^{\frac{1}{2}} = [\sum_{i\epsilon I}\sum_{\substack{j\epsilon I\\ i\neq j}} \sigma_{t_{ij}}^2 X_{ij}^2]^{\frac{1}{2}} = [\sum_{i\epsilon I}\sum_{\substack{j\epsilon I\\ i\neq j}} \Psi\mu_{t_{ij}} X_{ij}]^{\frac{1}{2}} \qquad (5.82)$$

$$= \Psi^{\frac{1}{2}}[\sum_{i\epsilon I}\sum_{\substack{j\epsilon I\\ i\neq j}} \mu_{t_{ij}} X_{ij}]^{\frac{1}{2}}.$$

Substitute (5.82) in equality (5.23), then

$$\sum_{i\epsilon I}\sum_{\substack{j\epsilon I\\ i\neq j}} \mu_{t_{ij}} X_{ij} + N^{-1}(1-\alpha)\Psi^{\frac{1}{2}} [\sum_{i\epsilon I}\sum_{\substack{j\epsilon I\\ i\neq j}} \sigma_{t_{ij}} X_{ij}]^{\frac{1}{2}} = T1. \quad (5.83)$$

Let

$$v = [\sum_{i\epsilon I}\sum_{\substack{j\epsilon I\\ i\neq j}} \mu_{t_{ij}} X_{ij}]^{\frac{1}{2}} \text{ and } N^{-1}(1-\alpha) = \phi$$

then

$$T1 = v^2 + \phi\Psi^{\frac{1}{2}} v$$

$$v^2 + \phi\Psi^{\frac{1}{2}}v - T1 = 0.$$

Solve for v

$$v = [-\phi\Psi^{\frac{1}{2}} + (\phi^2\Psi + 4T1)^{\frac{1}{2}}]/2.$$

However,

$$v^2 = [(-\phi\Psi^{\frac{1}{2}} + (\phi^2\Psi + 4T1)^{\frac{1}{2}})/2]^2 = \overline{T}1. \quad (5.84)$$

Hence,

$$v^2 = \{[\sum_{i\epsilon I}\sum_{\substack{j\epsilon I\\ i\neq j}} \mu_{t_{ij}} X_{ij}]^{\frac{1}{2}}\}^2 = \overline{T}1,$$

or

$$\sum_{i\epsilon I}\sum_{\substack{j\epsilon I\\ i\neq j}} \mu_{t_{ij}} X_{ij} = \overline{T}1. \quad (5.85)$$

A similar analysis yields

$$\sum_{i\epsilon I}\sum_{\substack{j\epsilon I\\ i\neq j}} \mu_{t_i} X_{ij} = \overline{T}2. \qquad \text{and} \quad (5.86)$$

$$\sum_{i \epsilon I} \sum_{\substack{j \epsilon I \\ i \neq j}} \mu_{d_i} X_{ij} = \bar{Q}. \qquad (5.87)$$

5.8 Goal Programming Formulation

The GP approach is based on a priority structure of the established goals. In other words, the technique provides a solution according to the policy of the decision maker. The decision maker is thus required to determine the priority of the desired attainment of each goal and rank them in ordinal sequence for decision analysis.

The purpose of this section is to demonstrate the application of GP to decision problems in the area of SVRP. The model presented here has a priority of goals as follows:

1. All routes are feasible
2. Minimize the total cost of each vehicle route
3. Minimize the total travel and unload times
4. Minimize the route safety stock
5. Maximize the customer's satisfaction through the emergency service for the k^{th} customer
6. Meet the dependency conditions such that station r follows station s

The Linear Integer Goal Programming (LIGP) formulation of Problem B is provided by utilizing the results of Theorem (5.2). Constraints (5.92) and (5.93) will change to nonlinear forms when random variables other than poisson and chi-square are utilized.

System constraints:

1. Route feasibility

$$\sum_{\substack{j \in I \\ i \neq j}} X_{ij} + n(1) - p(1) = 1 \text{ for all } i \in I \tag{5.88}$$

$$\sum_{\substack{i \in I \\ i \neq j}} X_{ij} + n(2) - p(2) = 1 \text{ for all } j \in I \tag{5.89}$$

$$Z_i - Z_j + N\, X_{ij} + n(3) - p(3) = N - 1 \tag{5.90}$$

$$\forall\ i,j \in I,\ i \neq j \text{ and } i,j \neq 0.$$

Goal constraints:

2. Total cost of each vehicle route

$$\sum_{i \in I} \sum_{\substack{j \in I \\ i \neq j}} C_{ij}\, X_{ij} + n(4) - p(4) = C \tag{5.91}$$

3. Total elapsed time of each vehicle route

$$\sum_{i \in I} \sum_{\substack{j \in I \\ i \neq j}} \mu_{t\,ij}\, X_{ij} + \sum_{i \in I} \sum_{\substack{j \in I \\ i \neq j}} \mu_{t_i}\, X_{ij} + n(5) - p(5) = \bar{T}1 + \bar{T}2. \tag{5.92}$$

When the minimization of the total travel and unload times of each vehicle route to target levels $\bar{T}1$ and $\bar{T}2$ are required, the goal constraint (5.92) can be divided into the following constraints, respectively:

$$\sum_{i \in I} \sum_{\substack{j \in I \\ i \neq j}} \mu_{t_{ij}}\, X_{ij} + n'(5) - p'(5) = \bar{T}1 \text{ and}$$

$$n'(5) \in n(5)$$
$$p'(5) \in p(5)$$

$$\sum_{i\epsilon I} \sum_{\substack{j\epsilon I \\ i\neq j}} \mu_{t_i} X_{ij} + n''(5) - p''(5) = \bar{T}2$$

$$n''(5) \in n(5)$$
$$p''(5) \in p(5)$$

4. Route safety stock

$$\sum_{i\epsilon I} \sum_{\substack{j\epsilon I \\ i\neq j}} \mu_{d_i} X_{ij} + n(6) - p(6) = [(Q - \bar{Q})/N^{-1}(1 - \eta)]^2 = \text{a constant} \tag{5.93}$$

5. Emergency service for the k^{th} customer

$$X_{0k} + n(7) - p(7) = 1 \tag{5.94}$$

where all X_{ij}'s are either 0 or 1.

6. Meet the dependency conditions such that station r follows station s

$$X_{rs} + n(8) - p(8) = 1 \tag{5.95}$$

The system constraints, (5.88) – (5.90), are based on the logic and philosophy of the VRP in that only one station must follow station i on a given route. These constraints can be achieved by minimizing

$$P_1 \; [n(1) + p(1) + n(2) + p(2) + p(3)]$$

where n(.) and p(.) indicate the vectors of the underachievement and overachievement for the set of system constraints and P_1 indicates the first goal priority. The goal constraints, (5.91) – (5.95), can be achieved through the minimization of $P_2[p(4)]$, $P_3[p(5)]$, $P_4[p(6)]$, $P_5[n(7) + p(7)]$, and $P_6[n(8) + p(8)]$, respectively.

The trade-offs among the goals (2) – (5) can be easily made. For instance, if the goal of route safety stock is more important than the total cost, total time of each vehicle route, and customer's satisfaction, it is necessary to minimize $P_2[p(6)]$ after minimizing $P_1[.]$. In this problem, it is assumed that neither underachievement nor overachievement of the fifth and sixth goals are desirable. Therefore, the variables n(1), p(1), n(2), p(2), p(3), n(7), p(7), n(8), and p(8) are to be minimized. However, it is assumed that the total cost and total elapsed time must be less than predetermined levels C and $\bar{T}1 + \bar{T}2$, respectively. Thus, only p(4) and p(5) are to be minimized for these two goals. Since the route safety stock cannot exceed the value of $[(Q - \bar{Q})/N^{-1}(1 - \eta)]^2$, then p(6) is to be minimized.

5.9 Sensitivity of Time and Truck Capacity upon the Probability of Route Failures

In most cases, some of the problem data are not known exactly, but are estimated as accurately as possible. The probability of route failures α, β, and η might not exactly be known for the decision maker because travel and unload times, and customer demands are random variables. Therefore, the probability of route failures forces one to analyze the sensitivity of time and truck capacity.

<u>Theorem 5.3</u>

If α, the probability of route failure, increases (up to 0.5), the value of $\bar{T}1$ will increase provided that T1 is a fixed value.

Proof:

Let us reconsider equation (5.84) in the form of (5.96), where $\phi = Z = Z_{1-\alpha} = N^{-1}(1 - \alpha)$ and T1 is fixed value:

$$\bar{T}1 = (2Z^2\ \Psi + 4T1 - 2Z\Psi^{\frac{1}{2}}\ (Z^2\ \Psi + 4T1)^{\frac{1}{2}})/4 \qquad (5.96)$$

$$\partial\bar{T}1/\partial\alpha = \partial\bar{T}1/\partial Z \cdot \partial Z/\partial\alpha \cdot \qquad (5.97)$$

$\partial Z/\partial\alpha < 0$, and this is because when α increases, then $(1 - \alpha)$ and $Z_{(1\ -\ \alpha)}$ decrease. However, after some calculations, one has

$$\partial\bar{T}1/\partial Z = (Z\Psi(Z^2\Psi + 4T1)^{\frac{1}{2}} - 3/2Z^2\Psi^{3/2} - 2T1\Psi^{\frac{1}{2}})/(Z^2\Psi + 4T1)^{\frac{1}{2}}.$$

Notice that when a and b are two positive numbers, the following inequality exists:

$$(a + b)^{\frac{1}{2}} < (a)^{\frac{1}{2}} + (b)^{\frac{1}{2}}. \qquad (5.98)$$

Therefore,

$$Z\Psi(Z^2\Psi + 4T1)^{\frac{1}{2}} < Z\Psi(Z^2\Psi)^{\frac{1}{2}} + Z\Psi(4T1)^{\frac{1}{2}} = Z^2\Psi^{3/2} + 2Z\Psi T1^{\frac{1}{2}}. \qquad (5.99)$$

Hence, due to inequality (5.99),

$$\partial\bar{T}1/\partial Z < (Z^2\Psi^{3/2} + 2Z\Psi T1^{\frac{1}{2}} - 3/2\ Z^2\Psi^{3/2} - 2T1\Psi^{\frac{1}{2}})/(Z^2\Psi + 4T1)^{\frac{1}{2}}$$

$$= (- 1/2\Psi^{3/2}\ Z^2 + 2Z\Psi T1^{\frac{1}{2}} - 2T1\Psi^{\frac{1}{2}})/(Z^2\Psi + 4T1)^{\frac{1}{2}}. \qquad (5.100)$$

The numerator on the right hand side of (5.100) is

$$- [2^{-\frac{1}{2}}\ Z\Psi^{3/4} - (2T1\Psi^{\frac{1}{2}})^{\frac{1}{2}}]^2$$

$$\partial\bar{T}1/\partial Z < - [(\frac{1}{\sqrt{2}})Z\Psi^{3/4} - \sqrt{2T1\Psi^{\frac{1}{2}}}]^2/(Z^2\Psi + 4T1)^{\frac{1}{2}} < 0. \qquad (5.101)$$

Hence

$$\partial\bar{T}1/\partial\alpha = \partial\bar{T}1/\partial Z \cdot \partial Z/\partial\alpha > 0. \quad \text{Q.E.D.}$$

Corollary 1 to Theorem 5.3

If β, the probability of route failure, increases (up to 0.5) then the value of $\bar{T}2$ will increase provided that T2 is a fixed value.

Corollary 2 to Theorem 5.3

If η, the probability of route failure, increases (up to 0.5) then the value of $\bar{Q}$ will increase provided that Q is a fixed value.

Theorem 5.4

Suppose that α_k and β_k, the probability of route failures, are set such that $d\beta_k/d\alpha_k > 0$, then by increasing α_k and β_k (up to 0.5), the total elapsed time of the k^{th} route will decrease.

Proof:

Let $Z_1 = Z(1 - \alpha_k) = N^{-1}(1 - \alpha_k)$ and $Z_2 = Z(1 - \beta_k) = N^{-1}(1 - \beta_k)$.

The total elapsed time of the k^{th} route is

$$T_k = \sum_{i\in I} \sum_{\substack{j\in I \\ i\neq j}} \mu_{t_{ij}} X_{ijk} + \sum_{i\in I} \sum_{\substack{j\in I \\ i\neq j}} \mu_{t_i} X_{ijk} + Z_1\Big(\sum_{i\in I} \sum_{\substack{j\in I \\ i\neq j}} \sigma^2_{t_{ij}} X^2_{ijk}\Big)^{\frac{1}{2}}$$

$$+ Z_2\Big(\sum_{i\in I} \sum_{\substack{j\in I \\ i\neq j}} \sigma^2_{t_i} X^2_{ijk}\Big)^{\frac{1}{2}}. \qquad (5.102)$$

Now, it is necessary to show that $\partial T_k/\partial\alpha_k < 0$ where

$$\partial T_k/\partial\alpha_k = \partial T_k/\partial Z_1 \cdot \partial Z_1/\partial\alpha_k + \partial T_k/\partial Z_2 \cdot \partial Z_2/\partial\beta_k \cdot d\beta_k/d\alpha_k. \qquad (5.103)$$

$\partial Z_1/\partial\alpha_k$ and $\partial Z_2/\partial\beta_k$ are both less than zero because by increasing α_k and β_k, $(1 - \alpha_k)$ and $(1 - \beta_k)$ decrease and consequently $Z_1 = Z(1 - \alpha_k)$ and $Z_2 = Z(1 - \beta_k)$ decrease. However,

$$\partial T_k/\partial Z_1 = \Big(\sum_{i\epsilon I}\sum_{\substack{j\epsilon I\\ i\neq j}} \sigma^2_{t_{ij}}\; X^2_{ijk}\Big)^{\frac{1}{2}} > 0 \tag{5.104}$$

$$\partial T_k/\partial Z_2 = \Big(\sum_{i\epsilon I}\sum_{\substack{j\epsilon I\\ i\neq j}} \sigma^2_{t_i}\; X^2_{ijk}\Big)^{\frac{1}{2}} > 0 \tag{5.105}$$

$\partial T_k/\partial\alpha_k = (+)\ (-) + (+)\ (-)\ (+) = (-) + (-) < 0.$ Q.E.D.

Theorem 5.5

If the condition of Theorem (5.4) exists for all NV truck routes, then the total elapsed time of the whole system decreases.

Proof:

According to Theorem (5.4) the total elapsed time of each vehicle route decreases and thus it can be concluded that the total elapsed time of the whole delivery system will decrease since NV remains unchanged.

Theorem 5.6

For a SVRP having only probabilistic customer demands, if η, the probability of route failure, increases (up to 0.5), then the total travel distance of the whole delivery system will decrease.

Proof:

To prove this theorem, the following, which is a mathematical model for a SVRP having only probabilistic customer demand and no time restrictions, is considered:

$$\text{Minimize:} \quad D = \sum_{k=1}^{NV} \sum_{i=0}^{TNS} \sum_{\substack{j=0 \\ i \neq j}}^{TNS} d_{ij} X_{ijk} \tag{5.106}$$

Subject to:

$$\sum_{i=1}^{TNS} \sum_{\substack{j=0 \\ i \neq j}}^{TNS} \mu_{d_i} X_{ijk} + N^{-1}(1 - \eta_k)\left(\sum_{i=1}^{TNS} \sum_{\substack{j=0 \\ i \neq j}}^{TNS} \sigma_{d_i}^2 X_{ijk}^2\right)^{\frac{1}{2}} \leq Q \tag{5.107}$$

$$X = [X_{ijk}] \in S. \tag{5.108}$$

By setting

$$Z = N^{-1}(1 - \eta_k) \text{ and } Y = \left(\sum_{i=1}^{TNS} \sum_{\substack{j=0 \\ i \neq j}}^{TNS} \sigma_{di}^2 X_{ijk}^2\right)^{\frac{1}{2}} > 0$$

it will be noticed that $\partial Z/\partial \eta < 0$ because by increasing η_k, $(1 - \eta_k)$ decreases and consequently $N^{-1}(1 - \eta_k)$ decreases. Hence, $Z \cdot Y$ decreases and consequently, $\bar{Q} = Q - Z \cdot Y$ will increase. However, $\partial D/\partial \eta = \partial D/\partial \bar{Q}\ \partial \bar{Q}/\partial Z\ \partial Z/\partial \eta$. It is obvious that $\partial \bar{Q}/\partial Z = - Y < 0$. Hence, it remains to show that $\partial D/\partial \bar{Q} < 0$, this is because $\partial \bar{Q}/\partial Z \cdot \partial Z/\partial \eta > 0$. Now, it is only necessary to prove that in the deterministic VRP where customer demands are equal to their demand's mean, by increasing the artificial capacity of truck the travelled distance will decrease. If the transportation cost depends linearly on the weight of goods delivered and the distance travelled, then the following equation can be used:

$$C_{ij} = U_{ij} \, W_{ij} \, d_{ij} \tag{5.109}$$

where

U_{ij} = cost per unit weight per unit distance from node i to node j,

W_{ij} = weight transported from node i to node j,

d_{ij} = the distance from node i to node j,

r_j = number of times that weight W_{ij} can be fitted in $\bar{Q}$.

However,

$$d_{ij} = \frac{C_{ij}}{U_{ij} \, W_{ij}} \tag{5.110}$$

and $\bar{Q} = r_j \, W_{ij}$

or $W_{ij} = \dfrac{\bar{Q}}{r_j}$

Since $X_{ij} = \begin{cases} 1 \\ 0 \end{cases}$, then

$$D = \sum_{k=1}^{NV} \sum_{i=0}^{TNS} \sum_{\substack{j=0 \\ i \neq j}}^{TNS} d_{ij} \, X_{ijk} = \sum_{k=1}^{NV} \sum_{i=0}^{TNS} \sum_{\substack{j=0 \\ i \neq j}}^{TNS} d_{ij}$$

or

$$D = \sum_{k=1}^{NV} \sum_{i=0}^{TNS} \sum_{\substack{j=0 \\ i \neq j}}^{TNS} \frac{C_{ij} \, r_j}{U_{ij} \, Q}.$$

Hence,

$$\frac{\partial D}{\partial Q} = \frac{-1}{(Q)^2} \sum_{k=1}^{NV} \sum_{i=0}^{TNS} \sum_{\substack{j=0 \\ i \neq j}}^{TNS} \frac{C_{ij} \, r_j}{U_{ij}} < 0.$$

This result indicates that $\partial D/\partial \bar{Q} < 0$, which proves this theorem. Q.E.D.

The following theorem is concerned with the number of constructed vehicle routes for the SVRP having probabilistic customer demands. It indicates that when number of constructed vehicle routes increases, then the total demand to be served by the vehicles will increase. However, this situation will not happen when customer demands are deterministic.

Theorem 5.7

The larger number of routes is equivalent to the larger total demand to be served by all vehicles.

Proof:

Suppose that μ_i and σ_i^2 are the mean and variance of demand point i. It is clear that

$$\sum_{i=1}^{TNS} \mu_i = \mu_1 + \mu_2 + \ldots + \mu_{TNS}. \quad (5.112)$$

But

$$(\sum_{i=1}^{TNS} \sigma_i^2)^{\frac{1}{2}} < \sigma_1^{\frac{1}{2}} + \sigma_2^{\frac{1}{2}} + \ldots + \sigma_{TNS}^{\frac{1}{2}}. \quad (5.113)$$

If only one vehicle can be used to deliver all customer demands, then the following inequality is needed:

$$\sum_{i=1}^{TNS} \mu_i + N^{-1}(1 - \eta) (\sum_{i=1}^{TNS} \sigma_i^2)^{\frac{1}{2}} \leq Q. \quad (5.114)$$

If two vehicles are used instead of one vehicle to deliver the customer demands, the following inequalities are needed;

$$\sum_{i=1}^{m<TNS} \mu_i + N^{-1}(1 - \eta)(\sum_{i=1}^{m<TNS} \sigma_i^2)^{\frac{1}{2}} \leq Q \quad (5.115)$$

and

$$\sum_{i=m+1}^{TNS} \mu_i + N^{-1}(1 - \eta)(\sum_{i=m+1}^{TNS} \sigma_i^2)^{\frac{1}{2}} \leq Q. \qquad (5.116)$$

Hence, after the addition of both sides of inequalities (5.115) and (5.116), the result is the following inequality:

$$\sum_{i=1}^{TNS} \mu_i + N^{-1}(1 - \eta)[(\sum_{i=1}^{m<TNS} \sigma_i^2)^{\frac{1}{2}} + (\sum_{i=m+1}^{TNS} \sigma_i^2)^{\frac{1}{2}}] \leq 2Q \qquad (5.117)$$

The left-hand side of inequality (5.117) represents the total generated demands to be served by two vehicles. Using the concept of inequality (5.113), inequality (5.117) can be written in the following form

$$\sum_{i=1}^{TNS} \mu_i + N^{-1}(1 - \eta)(\sum_{i=1}^{TNS} \sigma_i^2)^{\frac{1}{2}} \leq \sum_{i=1}^{TNS} \mu_i + N^{-1}(1 - \eta). \qquad (5.118)$$

$$[(\sum_{i=1}^{m} \sigma_i^2)^{\frac{1}{2}} + (\sum_{i=m+1}^{TNS} \sigma_i^2)^{\frac{1}{2}}] < 2Q.$$

The inequality (5.118) indicates that the total generated demand using two vehicles is larger than using one vehicle. However, one can extend the previous discussion for NV vehicles which are needed to satisfy all customer demands.

5.10 Summary

This chapter has presented the development of a multiple objective goal programming formulation of SVRP in which customer demand and travel and unload times are considered to be random variables having known distribution functions. The mathematical formulations of the problem were directly related to the model developments for the RCS and RIS of the problem. The equivalent deterministic forms of Problems "C" and "D" were developed and presented by "E" and "F" type problems, respectively.

The existence of the optimum solution for the RCS of the problem is shown through Theory (5.1). The existence of a set of deterministic linear time constraints, which are equivalent to the nonlinear set of time constraints of the problem for distributions such as poisson and chi-square, is proved through Theory (5.2). The effects of route failure probabilities on the total elapsed time of the whole delivery system were proved through Theories (5.3) – (5.5). Theory (5.6) illustrates that the total travelled distance decreases when route failure probability increases. Additionally, Theory (5.7) is provided to demonstrate that the larger number of vehicle routes is equivalent to the larger customer demands.

The remainder of this research is divided into five chapters. Chapter VI is devoted to the development of the LIGP technique. Chapter VII demonstrates the development of an appropriate heuristic algorithm for obtaining favorable vehicle routes for "E" and "F" type problems. These heuristic approaches are new modifications of Clarke and Wright's algorithm. Chapter IX discusses the details of two interactive computer programs for the SVRP and LIGP techniques. Chapter X gives the conclusions and recommendations for future research in the area of SVRP.

CHAPTER VI

LINEAR INTEGER GOAL PROGRAMMING TECHNIQUE

6.1 Introduction

Linear Programming (LP) is a well known mathematical technique for optimizing a single objective function such as profit, total elapsed time, or total travelled distance, subject to stated constraints. The LP technique is employed in decision making situations in many real world problems. Due to the existence of conflicting objectives in many decision making situations, the area of multiple objective decision making has received a great deal of attention in recent years. In such decision making situations, the overall desire is that all objectives or goals be simultaneously met to as large an extent as possible.

One well known procedure which treats this problem is the Goal Programming (GP) technique. The GP programming technique assumes that the variables take continuous values within the feasible region. A Linear Integer Goal Programming (LIGP) problem is a goal programming problem in which the constraints and objective functions are linear, but the variables in the final solution are required to be integers.

Two distinct GP techniques, Preemptive Goal Programming (PREGP) and Partitioning Goal Programming (PARGP), are employed as a basis of algorithm routings. The PREGP and PARGP procedures are based on the simplex method of LP and Arthur and Ravindran's technique for solving linear goal programming problems. Arthur and Ravindran [4] have devised a PARGP technique which consists of solving a series of linear programming subproblems with the solution to the higher priorities used as the initial solution of the lower priority problem. The major advantage of PARGP, relative to PREGP, is that one deals with the fewer constraints, fewer variables, and only one objective function at each stage of the the problem.

The methodology for solving linear goal programs cannot be used to solve the linear integer goal program. Therefore, the PREGP and PARGP are extended to accomplish the handling of integer variables. However, two approaches are used for the integer algorithm developments of goal programs. The first approach deals with the development of the "cutting planes" (new objectives) that are added to the problem formulation when the continuous solution of the original problem has been obtained by the algorithm. The second approach concerns the development of the branch and bound algorithm for linear integer goal programs. However, the developed linear integer goal programs for GP techniques of PREGP and PARGP are called LIPREGP and LIPARGP, respectively.

6.2 Cutting Plane Method for Integer Goal Programming

The Cutting Plane (CP) method is a procedure which is used in literature [42] to solve the integer GP problems. The CP algorithms were originally developed in 1958 by Ralph Gomory [40, 56] for general Integer Linear Programming (ILP). The main difference between the Gomory procedure for Linear Integer Programming (LIP) and the CP of goal programming is the method in which these procedures handle the multidimensional priority weights.

The following sections are devoted to the development of the CP method for the All-Integer and Mixed-Integer LGP problems. The All-Integer, where all variables are required to be integer valued, will be discussed first, followed by a discussion of the Mixed-Integer case.

6.2.1 Development of All-Integer Cut

Suppose that the i^{th} constraint is the selected source row and it appears in the final tableau of the related GP as

$$\sum_{j=1}^{n} a_{ij}\, x_j = b_i \qquad (i = \text{source row}) \tag{6.1}$$

where for simplicity x_j denotes any variable, whether decision or deviation; a_{ij} is the coefficient of variable x_j in the source row, and b_i is the right-hand-side value of the source row. Indicate the integer part of a as [a], then, since $[a_{ij}] \leq a_{ij}$ and $x_j \geq 0$, one can write

$$\sum_{j=1}^{n} [a_{ij}]\, x_j \leq b_i. \tag{6.2}$$

Any integer vector x which satisfies (6.1) will also satisfy (6.2). For such an x, the left-hand side of (6.2) is an integer. Hence, integer vector x must also satisfy the following constraint

$$\sum_{j=1}^{n} [a_{ij}]\, x_j \leq [b_i] \tag{6.3}$$

On the assumption that a_{ij} and b_i are not integer valued, one can write

$$[a_{ij}] + f_{ij} = a_{ij} \tag{6.4}$$

and

$$[b_i] + f_i = b_i \tag{6.5}$$

where $0 \le f_{ij} < 1$ and $0 \le f_i < 1$.

After substituting (6.4) and (6.5) in (6.3), then

$$\sum_{j=1}^{n} (-f_{ij})\, x_j \le -f_i \tag{6.6}$$

Inequality (6.6) can further be changed into

$$\sum_{j=1}^{n} (-f_{ij})\, x_j + n_i - p_i = -f_i \tag{6.7}$$

Equation (6.7) is the cutting plane constraint to be added to the final tableau of the GP problem with the noninteger variables. In order to take care of the infeasibility resulting from the addition of constraint (6.7) into the final tableau of the GP problem, the dual simplex method may be employed to solve the new problem. Alternatively, the cutting plane (6.7) can be arranged as in (6.8):

$$\sum_{j=1}^{n} f_{ij}\, x_i + n_i - p_i = f_i. \tag{6.8}$$

In this case, the regular primal procedure is utilized.

In order to satisfy the cutting plane constraints (6.7) and (6.8), p_i and n_i should be minimized at the first priority level, respectively, and all other priorities are downgraded one level lower than their original assignment.

6.2.2 Development of the Mixed-Integer Cut

A Mixed-Integer Linear Goal Programming (MILGP) problem [40] can be developed in a manner similar to the pure IGP problem as previously described. In this case, only certain variables are to be integer-valued. The remaining ones take on feasible values on the continuous scale. Suppose that x_j is a variable which is required to be integral, then the i^{th} (source row) constraint can be written as

$$x_j + \sum_{j \in \text{nonbasic}} f'_{ij}\, x_j = b_i \tag{6.9}$$

By considering

$$b_i = [b_i] + f_i \tag{6.10}$$

then

$$x_j + \sum_{j \in \text{nonbasic}} f'_{ij}\, x_j = [b_i] + f_i \tag{6.11}$$

or

$$\sum_{j \in \text{nonbasic}} f'_{ij}\, x_j + n_i - p_i = ([b_i] - x_j) + f_i \tag{6.12}$$

where, the index i indicates the source row, f_i is the fractional part of the right-hand side value of the source row, and f'_{ij} is defined as follows:

$$f'_{ij} = \begin{cases} a_{ij} & \text{if } a_{ij} \geq 0 \text{ and } x_j \text{ is a continuous variable} \\ (f_i/(f_i-1))a_{ij} & \text{if } a_{ij} < 0 \text{ and } x_j \text{ is a continuous variable} \\ f_{ij} & \text{if } f_{ij} < f_i \text{ and } x_j \text{ is an integer variable} \\ (f_i/(1-f_i))(1-f_{ij}) & \text{if } f_{ij} > f_i \text{ and } x_j \text{ is an integer variable} \end{cases} \tag{6.13}$$

Note that f'_{ij} is the nonnegative fractional part of a_{ij}. When the cutting plane method (6.12) is chosen to be used, then p_i is to be minimized at the first priority level. All other priorities are to be downgraded one level lower than their original assignment. However, using either technique, the process of adding cutting plane methods and solving the new problem is repeated until an integer solution is reached. This process is described in more detail below.

6.3 Algorithm for LIGP using Cutting Plane Method

The proposed algorithm can be summarized as follows:

Step 1: Solve the initial GP problem by dropping the integerality requirements. If the solution to this problem is integer, stop. Otherwise, go to Step 2.

Step 2: Generate the cutting plane constraint as shown in (6.8) or (6.12), depending on the type of problem (pure or mixed integer). A most promising technique for choosing the source row is to choose the constraint in the final simplex tableau which gives the largest f_i.

Step 3: Solve the new problem with the augmented cutting plane. Use the regular method of the GP procedure. If the solution to this problem is integer, stop. Otherwise, go to step 2.

Example 1

The following problem was taken from A. A. Abduelmagd [1] for illustration of this procedure. This problem is solved by the preemptive LIGP procedure using the cutting plane method where an integer solution to variables x_1, x_2, X_3, x_4, x_5, and x_6 is required.

$$\text{Minimize} \quad P_1(n_1 + p_1 + n_2 + p_2) + P_2(n_3) + P_3(n_4)$$

Subject to:

$$8x_1 + x_2 + 3x_3 + 2x_4 + 3x_5 - 3x_6 + n_1 - p_1 = 17$$
$$3x_1 + 2x_3 + x_4 + x_5 - x_6 + n_2 - p_2 = 5$$
$$5x_1 + x_3 + 2x_4 + x_5 - 4x_6 + n_3 - p_3 = 8$$
$$12x_1 + x_2 + 2x_3 + 5x_4 + 4x_5 - 6x_6 + n_4 - p_4 = 30$$
$$x_i \geq 0,\ n_i,\ p_i \geq 0 \text{ and } x_i \text{ are integer.}$$

A continuous solution to this problem is obtained after five simplex iterations have been performed. The solution is

$$x_1 = 0.40,\ x_2 = 7.0,\ x_4 = 4.60,\ x_6 = 0.80$$

where the remainder of variables are zero and all three priorities have been achieved. Since only six variables out of 14 variables (number of decision variables plus deviations) were required to be integer, the mixed integer procedure was employed to obtain an optimal integer solution for this problem. After 23 more iterations and seven cuts, the following integer solution was obtained:

$$x_1 = 0.0,\ x_2 = 6.0,\ x_3 = 0.0,\ x_4 = 4.0,\ x_5 = 1.0,\ x_6 = 0.0$$

where all priority levels were achieved.

6.4 Branch and Bound Method for Integer Goal Programming

The technique of Branch and Bound was originally introduced by Land and Doing [40, 56]. Due to the inefficiency of Branch and Bound for computer coding, a modification of the algorithm was developed by Dakin [40, 44, 56]. This technique, unlike the CP methods, can be applied directly to both the pure and mixed integer LGP problems. In order to apply the Dakin algorithm to a LGP, one starts to solve the problem by a general LGP with the integer requirements ignored. If the result of this GP happens to be an integer solution according to the original integer requirements of the problem, then the optimal solution has been achieved. If the optimal solution is not an integer solution, then a noninteger variable should be selected from the list of the required integer variables. After such a variable, x_j, is selected, one can write a range of the following form for that variable:

$$[b_j] \leq x_j \leq [b_j] + 1 \tag{6.14}$$

where $[b_j]$ represents the largest integer that is less than the value of b_j. Since x_j is required to be integer, the given range by (6.14) is infeasible for this variable. However, to avoid any solution in this range, the following conditions can be utilized as two objectives

$$x_j \leq [b_j]$$
$$x_j \geq [b_j] + 1$$

Each of these objectives is a presentation of a new problem which is branched from the previous problem. The GP problem which is associated with each new branch consists of the GP problem of the previous problem from which this branch emanates and one of these two new objectives. The Branch and Bound method can be summarized in the following five steps:

Step 1 (Initial Solution): Solve the problem by a general method of LGP by treating all variables (decision variables and deviations) as continuous. Check the optimal solution, if this solution satisfies the integer requirements of the problem, then the optimal solution has been obtained, otherwise, go to Step 2.

Step 2 (Branching Variable Selection): Select a variable from the set of variables which are constrainted to be integer and its solution value is not integer. Using this variable, develop two new objectives as follows:

$$x_j \leq [b_i] \tag{6.15}$$

and

$$x_j \geq [b_i]+1 \tag{6.16}$$

where x_j is the basic variable located in the ith row and b_i is the right-hand-side value of this row (or the solution value of x_j).

The objectives of (6.15) and (6.16) can be written in terms of nonbasic variables of the optimal tableau from which x_j was chosen. More clearly, objective (6.15) can be written as

$$x_j = b_i - \sum_{j=1}^{n} a_{ij}\ x_j \leq [b_i] \tag{6.17}$$

or

$$b_i - [b_i] \leq \sum_{j=1}^{n} a_{ij} x_j \quad (6.18)$$

By setting $f_i = b_i - [b_i]$, (6.18) can be written as

$$f_i \leq \sum_{j=1}^{n} a_{ij} x_j$$

Therefore,

$$\sum_{j=1}^{n} a_{ij} x_j + n_i - p_i = f_i \quad (6.19)$$

where n_i is to be minimized at the first priority level. Similarly, the objective $x_j \geq [b_i] + 1$ can be written as

$$\sum_{j=1}^{n} (-a_{ij}) x_j + n_i - p_i = (1 - f_i) \quad (6.20)$$

where n_i should be minimized again at the priority level one. Furthermore, it should be noted that each of the equations (6.19) and (6.20) will be treated separately as a new constraint and a new objective function in the goal programming formulation of new problems which emanate from the previous problem.

Step 3 (Formation of New Nodes): Add these new constraints to the goal programming problem by the node under consideration in Step 2. One subproblem is formed by augmenting constraint (6.19) and the other by augmenting constraint (6.20). Solve each of these subproblems as a linear goal programming using the simplex method to obtain two new solutions to these problems. Determine the degree of goal attainments (the optimal value of the objective function) for each subproblem separately.

Step 4 (Test for Terminal Node): Each of the nodes formed in Step 3 may be a terminal node for one of the following reasons:

1. The problem represented by the node may have no feasible solution.

2. The value of x_j, $j \in I$ are all integers (I indicates the set of all required integer variables).

In both cases, the node under consideration should be terminated. In the second case, the value of the objective function should be compared with current best available value. By defining vector $\bar{R}_i = (r_1, r_2, \ldots)$ as the value of priority levels at node i, then for any two solutions, say $\bar{R}_k$ and $\bar{R}_m$, $\bar{R}_k$ is preferred to $\bar{R}_m$ if a priority level of $\bar{R}_k$ is lower in value than the corresponding priority level in $\bar{R}_m$ and all preceding priority levels are equal in both $\bar{R}_k$ and $\bar{R}_m$ [38, pp. 130]. The priority level $\bar{R}_k$ is preferred to priority level $\bar{R}_r$ if $\bar{R}_k = (0, 100, 16, 300, 0)$ and $\bar{R}_r = (0, 100, 19, 401, 5)$.

Step 5 (Node Selection): If both nodes at Step 4 were terminated then select the next node from the list of nodes which are in the waiting list for further branching. If exactly one node in Step 4 was terminated, then use the nonterminal node and go to Step 2.

If both nodes in Step 4 were nonterminal, then choose the more promising one. A node with the smallest value of the objective function is considered to be more promising. However, the other node should be added to the list of waiting nodes for further branching which are considered later. Figure 1 depicts the branch and bound procedure for LIGP.

The selection of variables from which to generate new constraints is obviously one of the most important steps to be taken in the solution process of LIGP problem. The i^{th} constraint where the basic variable x_j assumes a noninteger value is considered to be the source row. The easiest way of selecting the source row is to pick up a basic variable with the largest fractional part. It is important to note that this rule is not an absolute one but it works well.

Example 2

Let us reconsider the problem which is given in example 1. Now an integer solution to this problem by the LIPARGP technique along with the branch and bound procedure is required.

A continuous solution to this problem was obtained after 5 iterations which is give below:

$X_1 = 0.40 \qquad X_2 = 7.0 \qquad X_4 = 4.6 \qquad X_6 = 0.80$

where all priorities are achieved and the remainder of variables are zero. The integer procedure has started on the sixth iteration and has stopped on the sixteenth iteration with the result given in Figure 2. Hence, the optimal integer solution to this problem by branch and bound technique is $X_1 = 0$, $X_2 = 7$, $X_3 = 0$, $X_4 = 5$, $X_5 = 1$, $X_6 = 1$ where priorities 1 and 3 have been achieved and the underachievement of priority 2 is equal to 1.

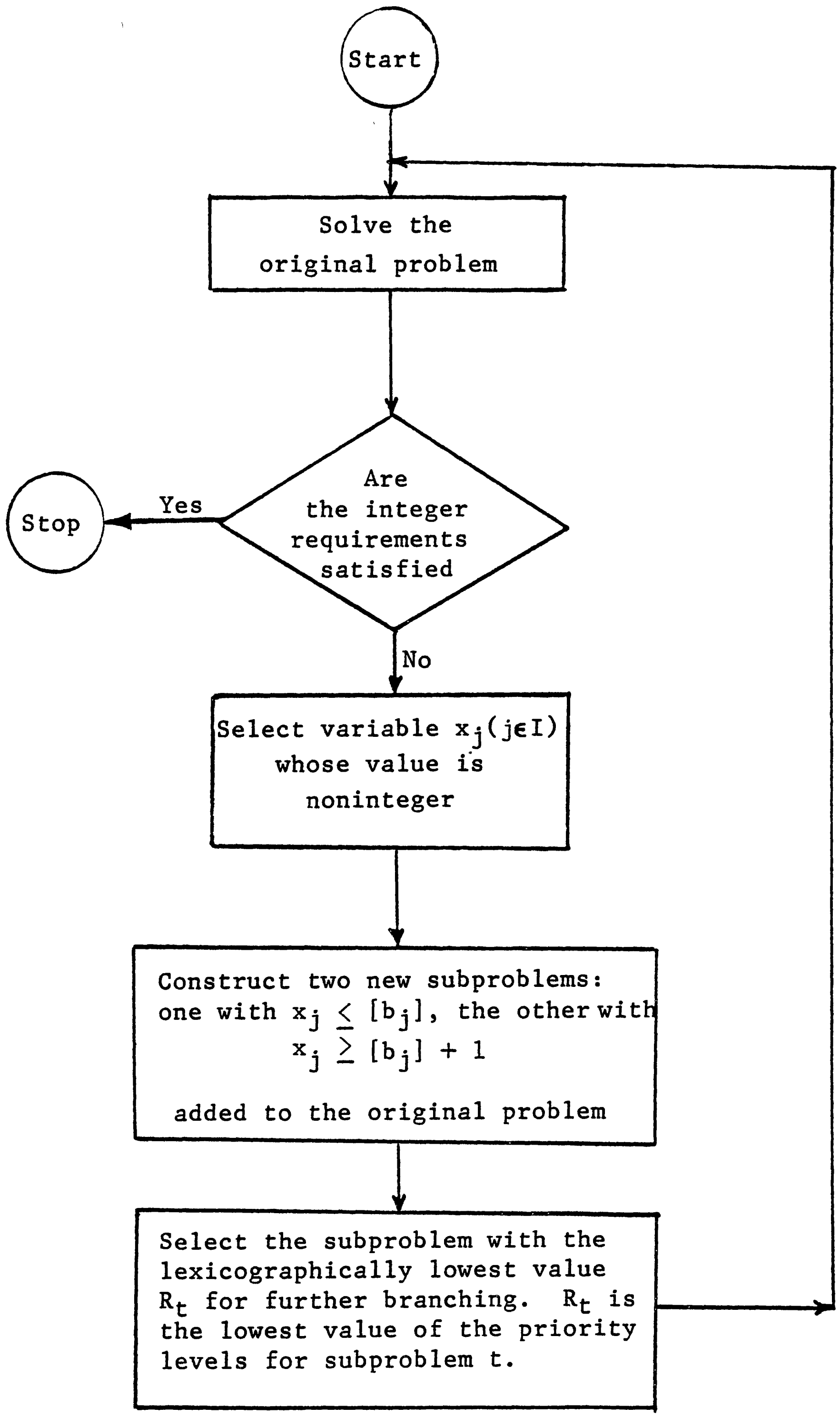

Figure 1. Flowchart for the Branch and Bound Procedure for LIGP

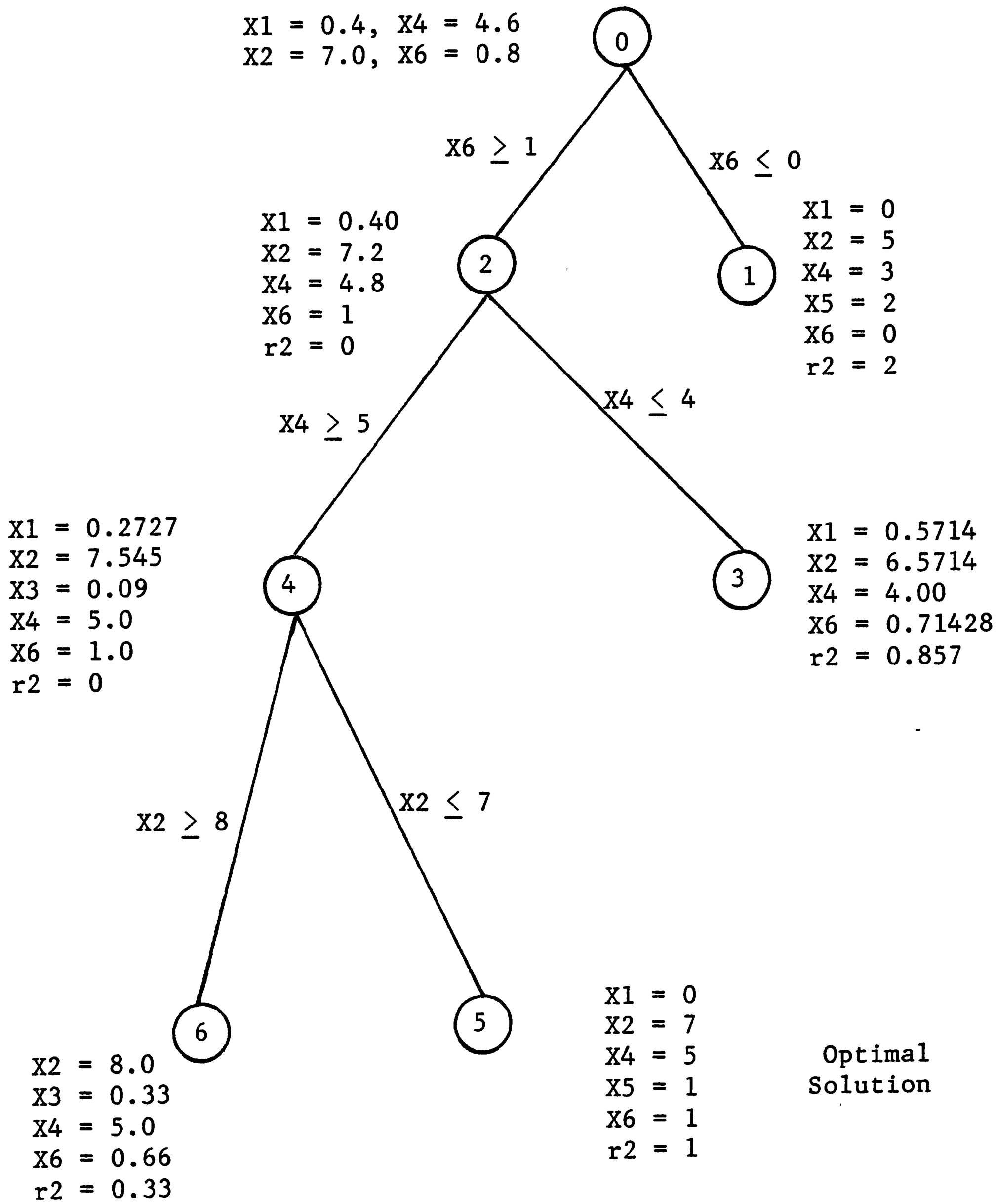

Figure 2. A Tree Diagram Presentation of the Solution of Example 2

Example 3

The problem which is presented in Table I [44] is a 10 variable problem with 15 constraints and 6 priority levels. Therefore, the total number of variables (decision and deviation) are 40. First, an attempt was made to find only an integer solution to this problem using the branch and bound method and LIPREGP. Therefore, an integer solution to this problem after 11.10 seconds and 115 iterations was obtained. The solution is

$$X_1 = 2,\ X_2 = 2,\ X_3 = 1,\ X_4 = 0,\ X_5 = 0,$$
$$X_6 = 1,\ X_7 = 0,\ X_8 = 0,\ X_9 = 1,\ X_{10} = 0$$

where all priorities have been achieved at the levels of

$$r_1 = r_2 = r_3 = r_4 = r_5 = 0 \text{ and } r_6 = 320.54.$$

Next, an attempt was made for the evaluation of 0-1 integer solution to the above problem. Therefore, constraints of the following type: $x_i + n_i - p_i = 1,\ \forall i = 1,2,\ldots,10$ and an absolute priority level $P_0 = \sum_{i=1}^{10} p_{i+m}$ were added into the original problem. Now this problem is composed of 60 variables, 25 constraints, and 7 priorities. The absolute priority level was considered as the first priority and the original priorities of the problem were downgraded by one level. This problem was solved by the LIPREGP using branch and bound method. Three integer solutions of the following were obtained after 68 iterations and 11.37 seconds. The first set of 0-1 integer solutions are:

(I) $X_1 = 1,\ X_2 = 1,\ X_3 = 0,\ X_4 = 0,\ X_5 = 1,$
$X_6 = 1,\ X_7 = 0,\ X_8 = 0,\ X_9 = 1,\ X_{10} = 0$

TABLE I

A 10-VARIABLE TEST PROBLEM FOR EXAMPLE 3*

Minimize: $Z = P_1(n_{14} + p_{15}) + P_2(p_1 + p_2 + p_3) + P_3 n_4 +$

$P_4(n_5 + n_6 + n_7) + P_5(n_8 + n_9 + n_{10}) +$

$P_6(p_{11} + p_{12} + p_{13})$

Subject to:

1	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8	X_9	X_{10}	b_i
1	10	20	100	75	200						250
2	50			150	90	100	40	10	10		259
3	10	5	1			10	4			75	300
4	256	315	173	160	680	475	95	65	120	115	1800
5	60	75	20	250	10	5					275
6	100	150	25	170	250	100	5	5		20	600
7	100	150	100	115	250	50	70	25	50	100	590
8	46	56	15		185	6					187
9	74	111	16	126	199	65				1	242
10	75	111	74	79	186	32	27	14	29	74	267
11	31	43	29	37	21	17	6	33	18	9	81
12	33	41	29	41	27	17	6	31	19	27	84
13	35	40	31	42	27	24	13	21	19	41	84
14									1	1	1
15									1	1	1

*The data of this test problem is taken from Lewis [44, p. 83].

where priorities 1 through 6 have been achieved at the level of 0, 0, 0, 125, 0, and 163.2, respectively. The second set of 0-1 integer solution is

(II) $X_1 = 1,\ X_2 = 1,\ X_3 = 0,\ X_4 = 1,\ X_5 = 0,$
$X_6 = 1,\ X_7 = 1,\ X_8 = 1,\ X_9 = 0,\ X_{10} = 1$

where priorities 1 through 6 have been achieved at the level of 0, 95, 309, 45, 78 and 343, respectively. Finally, the third set of 0-1 integer solution is

(III) $X_1 = 1,\ X_2 = 1,\ X_3 = 1,\ X_4 = 0,\ X_5 = 1,$
$X_6 = 1,\ X_7 = 0,\ X_8 = 0,\ X_9 = 1,\ X_{10} = 0$

where priorities 1 through 6 have been achieved at the level of 0, 83, 0, 105, 0, and 254, respectively.

Obviously, the optimal integer solution to this problem is the first set of integer solutions using the method of preference as previously described. Lewis [44, p. 82] solved this problem by the zero-one goal programming code. The same optimal integer solution was obtained by the 0-1 GP code after 12 seconds and 490 solution combinations.

6.5 Summary

The purpose of this chapter was to develop the linear integer goal programming techniques which could efficiently be used in the goal oriented problems. Two goal programming techniques, PREGP and PARGP, were used as the basis of the algorithm routings. The Cutting Plane method and Branch and Bound techniques were employed for solving the integer goal programming problems. Two cases of all integer variables and mixed integer variables were discussed for the Cutting Plane methods. Similar situations were investigated for the IGP based on the Branch and Bound procedure. The applicability of IGP methods were demonstrated through the solution of three example problems.

CHAPTER VII

DEVELOPMENT OF THE HEURISTIC ALGORITHM FOR STOCHASTIC VRP

7.1 Introduction

This chapter is concerned with the description and evaluation of the appropriate solution procedures for the "E" and "F" type problems. The proposed approaches for solving the "E" and "F" type problems are based on the Clarke and Wright "saving" approach to construct feasible vehicle routes which in turn satisfy the probabilistic customer demands at each station and the probabilistic travel and unload times constraints.

7.2 Clarke and Wright Algorithm

The Clarke and Wright algorithm [17] (saving approach) is the most widely known of the heuristics developed for solving delivery problems. In the saving approach, it is assumed that every two distinct demand points i and j are supplied individually by two vehicles (Figure 3).

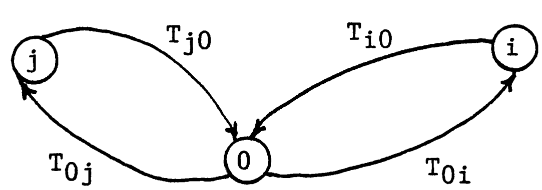

Figure 3. Initial Set-up

Figure 3 illustrates an initial set-up where one vehicle is assigned to one demand point. However, if instead of two vehicles, one uses only one vehicle (Figure 4), the saving in travelled time (cost or distance) is:

$$S_{ij} = 2T_{0i} + 2T_{0j} - (T_{0i} + T_{0j} + T_{ij}) = T_{0i} + T_{0j} - T_{ij} \qquad (7.1).$$

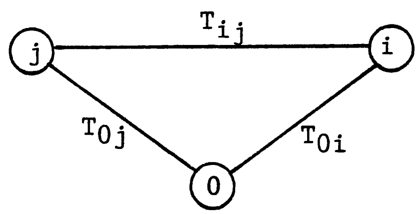

Figure 4. NODES i and j linked by using the Savings Approach concept.

The user calculates the "savings" associated with all pairs of locations to be serviced and then sorts these savings in decreasing order beginning with the list of pairs with positive values. Starting at the top of the list, the demand points are combined provided that the resulting tour is feasible and truck capacity is not violated. Using this method, increasingly larger and better tours are formed until the list of savings is exhausted. The chief deficiency of this method is that once an arc is added to a route, it is never removed.

7.3 Development of the Heuristic Approach For "E" Type Problem

The "E" type problem, as developed in Chapter V, is reproduced here.

$$\text{Minimize} \sum_{k=1}^{NV} \sum_{i=0}^{TNS} \sum_{j=0}^{TNS} c_{ij}\, x_{ijk} \tag{7.2}$$

Subject to:

$$\sum_{i=0}^{TNS} \sum_{j=0}^{TNS} \mu_{t_{ij}}\, x_{ijk} + N^{-1}(1-\alpha_k)\left(\sum_{i=0}^{TNS} \sum_{j=0}^{TNS} \sigma^2_{t\,ij}\, x^2_{ijk}\right)^{\frac{1}{2}} \le TR_k \tag{7.3}$$

$$\sum_{i=1}^{TNS} \sum_{j=0}^{TNS} \mu_{t_i}\, x_{ijk} + N^{-1}(1-\beta_k)\left(\sum_{i=1}^{TNS} \sum_{j=0}^{TNS} \sigma^2_{t\,i}\, x^2_{ijk}\right)^{\frac{1}{2}} \le UT_k \tag{7.4}$$

$$\sum_{i=1}^{TNS} \sum_{j=0}^{TNS} \mu_{d_i}\, x_{ijk} + N^{-1}(1-\eta_k)\left(\sum_{i=1}^{TNS} \sum_{j=0}^{TNS} \sigma^2_{d\,i}\, x^2_{ijk}\right)^{\frac{1}{2}} \le Q \tag{7.5}$$

$$X = [X_{ijk}] \in S \qquad \text{for } k = 1,2,\ldots,NV \tag{7.6}$$

To adapt the Clarke and Wright algorithm, the following rule previously described in Section 7.2 must be employed,

$$S_{ij} = C_{0i} + C_{0j} - C_{ij}$$

where C_{ij} is the cost of moving from station i to station j. In this problem, constraints (7.3), (7.4), and (7.5) should be checked for feasibility before the addition of any new node to an existing route or before combining two routes together. However, the procedure for solving the "E" type problem is completely identical to the Clarke and Wright algorithm with the exception that some additional checks must be made for new constraints. The "E" type problem algorithm uses the objective function $\sum \sum \sum C_{ij} X_{ijk}$ and nonlinear constraints (7.3), (7.4), and (7.5) in contrast to the deterministic form of the Clarke and

Wright algorithm where only linear constraints must be checked for feasibility. These evaluations make the procedure more complex, and consequently more memory allocation and computer time will be required.

The calculations in all three constraints, (7.3), (7.4), and (7.5), are carried out by using values $\tau_1 = N^{-1}(1 - \alpha)$, $\tau_2 = N^{-1}(1 - \beta)$, and $\tau_3 = N^{-1}(1 - \eta)$ rather than using α, β, and η, respectively. The evaluation of τ_1, τ_2, and τ_3 is not a very hard job. For instance, for normal distribution which has been assumed in equations (7.3), (7.4), and (7.5), τ_1, τ_2, and τ_3 will be the standard normal deviate z.

The flowchart shown in Figure 5 outlines the procedural steps for the method developed for the "E" type problem.

7.4 Development of Heuristic Approach For "F" Type Problem

The "F" type problem, as shown in Chapter V, is reproduced here.

$$\text{Minimize} \sum_{k=1}^{NV} \{[\sum_{i=0}^{TNS} \sum_{j=0}^{TNS} \mu_{t_{ij}} X_{ijk} + \sum_{i=1}^{TNS} \sum_{j=0}^{TNS} \mu_{t_i} X_{ijk} + N^{-1}(1 - \alpha_k) \tag{7.7}$$

$$(\sum_{i=0}^{TNS} \sum_{j=0}^{TNS} \sigma^2_{t_{ij}} X^2_{ijk})^{\frac{1}{2}} + N^{-1}(1 - \beta_k)(\sum_{i=1}^{TNS} \sum_{j=0}^{TNS} \sigma^2_{t_i} X^2_{ijk})^{\frac{1}{2}}]\}$$

Subject to:

$$\sum_{i=1}^{TNS} \sum_{j=0}^{TNS} \mu_{d_i} X_{ijk} + N^{-1}(1 - \eta_k)(\sum_{i=1}^{TNS} \sum_{j=0}^{TNS} \sigma^2_{d_i} X^2_{ijk})^{\frac{1}{2}} \le Q \tag{7.8}$$

$$X = [X_{ijk}] \in S. \qquad k = 1,\ldots,NV \tag{7.9}$$

As mentioned before, it is almost impossible to solve a large scale problem such as the "F" type problem by the exact procedure. Therefore,

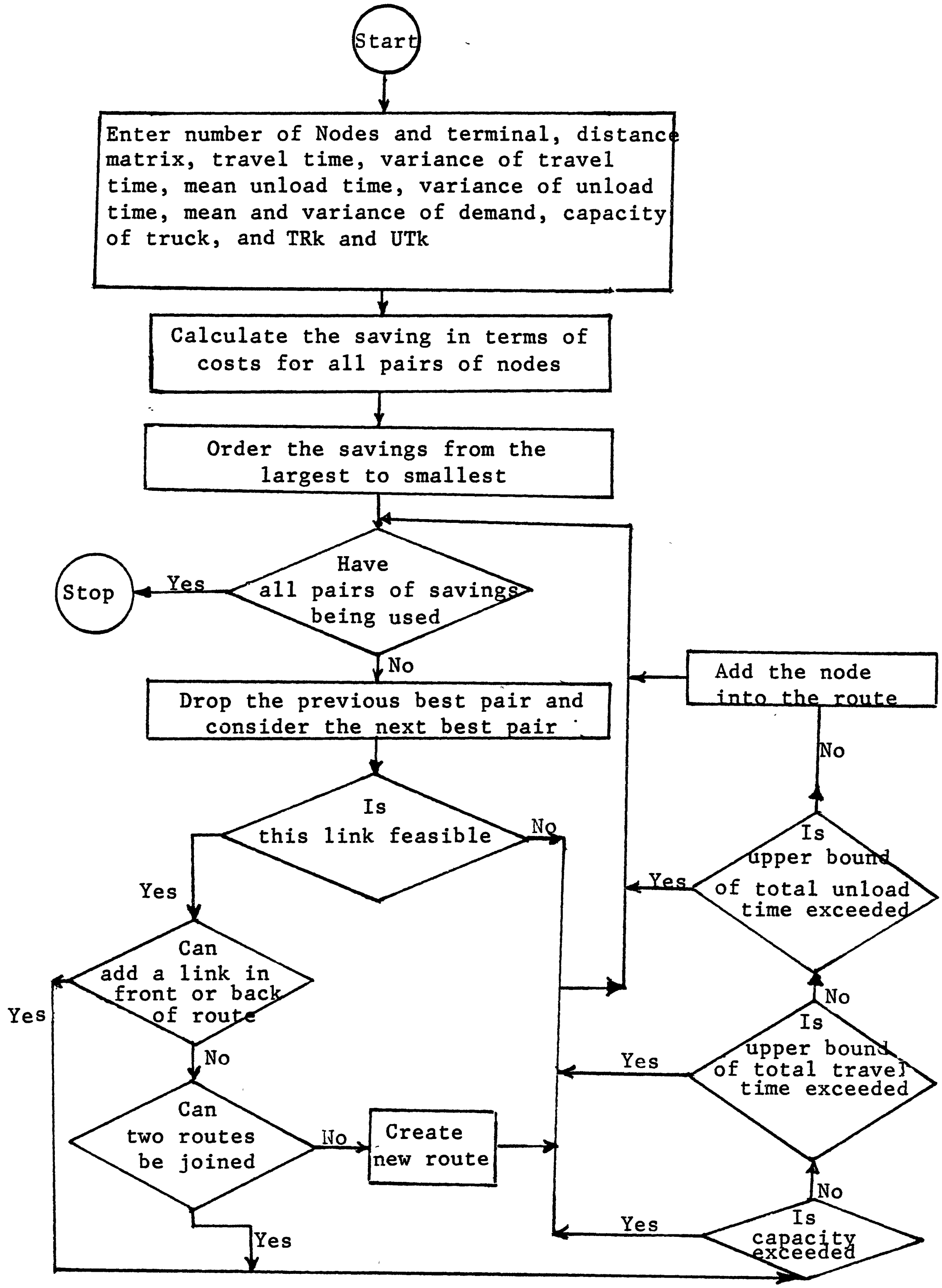

Figure 5. Algorithmic Flowchart of Procedural Steps for "E" Type Problem

heuristic approach is considered for solving this problem. In the "F" type problem the objective is to minimize total elapsed, travel, and unload times where these times are random variables. Two modifications of the Clarke and Wright algorithm as they pertain to the "F" type problem, are shown in Sections 7.4.1 and 7.4.2.

7.4.1 Algorithm (I)

To adapt the Clarke and Wright algorithm to handle the random travel and unload times when the minimization of the total elapsed time of the whole delivery system is the criterion, the saving function must be modified. A saving function of the following form can be used for evaluation of savings when travel time is random variable having a known distribution function:

$$S_{ij} = \gamma(\mu_{t_{0i}} + \mu_{t_{0j}} - \mu_{t_{ij}}) + (1 - \gamma)(\sigma^2_{t_{0i}} + \sigma^2_{t_{oj}} + \sigma^2_{t_{ij}})^{\frac{1}{2}} \quad (7.10)$$

provided that $\mu_{t_{0i}} + \mu_{t_{0j}} - \mu_{t_{ij}} > 0$ and $0 < \gamma \leq 1$. The saving function shown in (7.10) can be developed very easily. Consider Figures 3 and 4 which have been used for the Clarke and Wright saving algorithm: by considering the fact that the travel time between demand points i and j are random variables with means of $\mu_{t_{0i}}$, $\mu_{t_{0j}}$, and $\mu_{t_{ij}}$, and variances $\sigma^2_{t_{0i}}$, $\sigma^2_{t_{0j}}$, and $\sigma^2_{t_{ij}}$. Also, if one vehicle is used instead of using two, the saving in terms of these random variables is $ST = t_{0i} + t_{0j} - t_{ij}$, where t_{0i}, t_{0j}, and t_{ij} are assumed to be independent random variables. Therefore,

$$E(ST) = E(t_{0i} + t_{0j} - t_{ij}) = E(t_{0i}) + E(t_{0j}) - E(t_{ij})$$

or

$$E(ST) = \mu_{t0i} + \mu_{t0j} - \mu_{t\,ij} \tag{7.11}$$

and

$$Var(ST) = \sigma^2_{t0i} + \sigma^2_{t0j} + \sigma^2_{t\,ij}. \tag{7.12}$$

The total saving in terms of mean and standard deviation of random variable ST can be related in one expression with more emphasis in mean of saving than on the standard deviation of saving, as shown by Equation (7.10). In this equation, if $\gamma = 1$, then all emphasis is placed on the mean of saving ST which involves the basic concept of Clarke and Wright's algorithm. On the other hand, if $0 < \gamma < 1$, then a combination of mean and standard deviation of saving ST will be used.

7.4.2 Algorithm (II)

To adapt the Clarke and Wright algorithm, the following saving function can be used:

$$F_{saving} = \mu_{saving} + (\bar{\sigma}^2/(\delta*(\sigma^2_{saving})^{\frac{1}{2}})) \tag{7.13}$$

where $\mu_{saving} = E(ST)$, $\sigma^2_{saving} = var(ST)$, M is the total number of $\sigma^2_{t\,ij}$, $\delta > 0$ and

$$\bar{\sigma}^2 = (\sum_{ij} \sigma^2_{t\,ij}/M). \tag{7.14}$$

Maximizing function (7.13) provides a station which can be added to the vehicle route. When δ approaches to infinitive, then $F_{saving} = \mu_{saving}$, which is the basic concept of the Clarke and Wright algorithm. On the other hand, when $\delta \rightarrow 0$, then a great emphasis is placed on the standard deviation rather than on the mean. Equation (7.14) indicates that $\bar{\sigma}^2$ is a constant value for each specific problem. Hence, F_{saving} for each pair of demand points depends upon the value of saving in mean and on the amount of standard deviation between these two stations. Since great emphasis will more often be placed on the mean of saving rather than on the variance of saving, one can logically design a larger coefficient for saving in mean than on the variance. However, one may employ both algorithms to solve a SVRP and then accept the solution with the lowest total travel time for the whole system. The flowchart shown in Figure 6 outlines the procedural steps for the method of solution for the "F" type problem.

To compare saving function (7.10) and (7.13), consider the following new notations for simplicity: $x = \mu_{saving}$, $y = (\sigma^2_{saving})^{1/2}$, $z = S_{ij}$, $c = \bar{\sigma}^2$ = constant and $z' = F_{saving}$. Hence, the saving functions (7.10) and (7.13) can be written in terms of new notations as:

$$z = \gamma x + (1 - \gamma)y, \text{ and}$$

$$z' = x + \frac{c}{\delta * y}$$

respectively. To obtain a relationship between γ and δ, let $z = z'$. Therefore,

$$\gamma x + (1 - \gamma)y = x + \frac{c}{\delta * y} \qquad (7.15)$$

or

$$\gamma = 1 + \frac{c}{\delta * y * (x-y)} \qquad (7.16)$$

Equation (7.16) indicates that γ approaches 1 when $\delta \rightarrow \infty$ regardless of the values of x, y, and c. However, one can expect to obtain similar results by algorithms (I) and (II) of the "F" type problem when

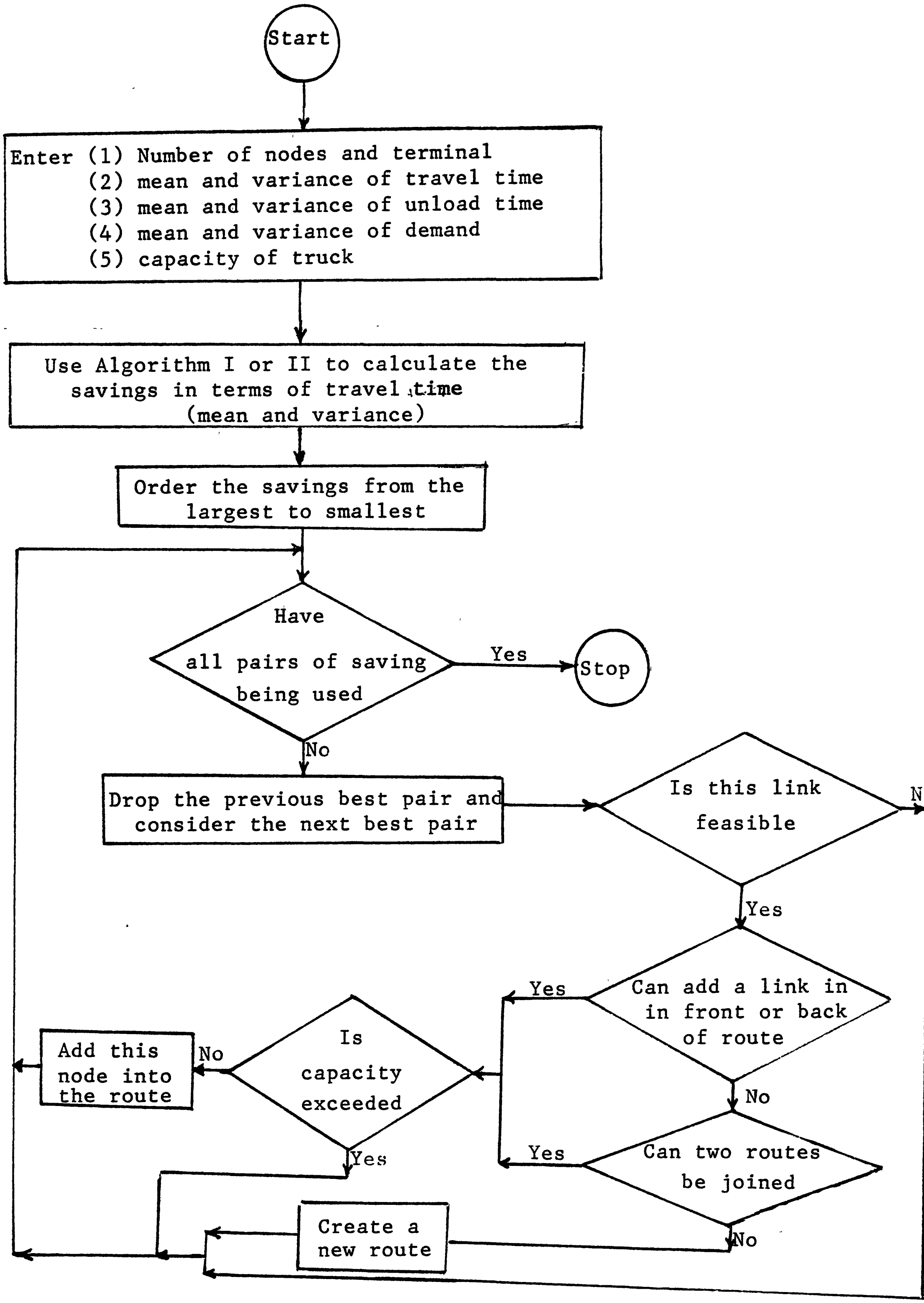

Figure 6. Algorithmic Flowchart of Procedural Steps for "F" Type Problem

γ and δ accept large values in ranges zero to one and zero to infinitive, respectively.

A flowchart summarizing the general structure of the analysis and solution of the SVRP is shown in Figure 7. The analysis begins with the determination of the type of the objective function which is to be minimized as was shown by problem C or D. The next step is the determination of the equivalent deterministic forms of problems C or D as were presented by the "E" or "F" type problem. The "E" or "F" type problem is used for the purpose of constructing routes. The routes are determined using the appropriate heuristic approach of "E" or "F" type problem.

If the decision maker is willing to accept the vehicle routes constructed by the RCS of the problem without change, then the procedure halts. Otherwise, the DM should consider the output information from the RCS of the problem and provide a set of goals for the RIS. Two GP models for the RIS of the problem are shown by problems A and B.

Prior to the formulation of each problem (each vehicle route is called a problem) as a 0-1 integer GP problem, values $\bar{T}1$ and $\bar{T}2$ for problems A and $\bar{Q}$ for problem B should be evaluated. This 0-1 integer CP problem can be solved using either LIPARGP or LIPREGP techniques. The route improvement stage should be applied to those vehicle routes that do not satisfy the customer and decision maker's requirements after RCS of the problem has been applied.

7.5 Example Problem

The algorithm for the multiple objective GP model of the SVRP is illustrated by a simple example problem. The following small problem is involved with a single depot and 15 locations to be served by vehicles.

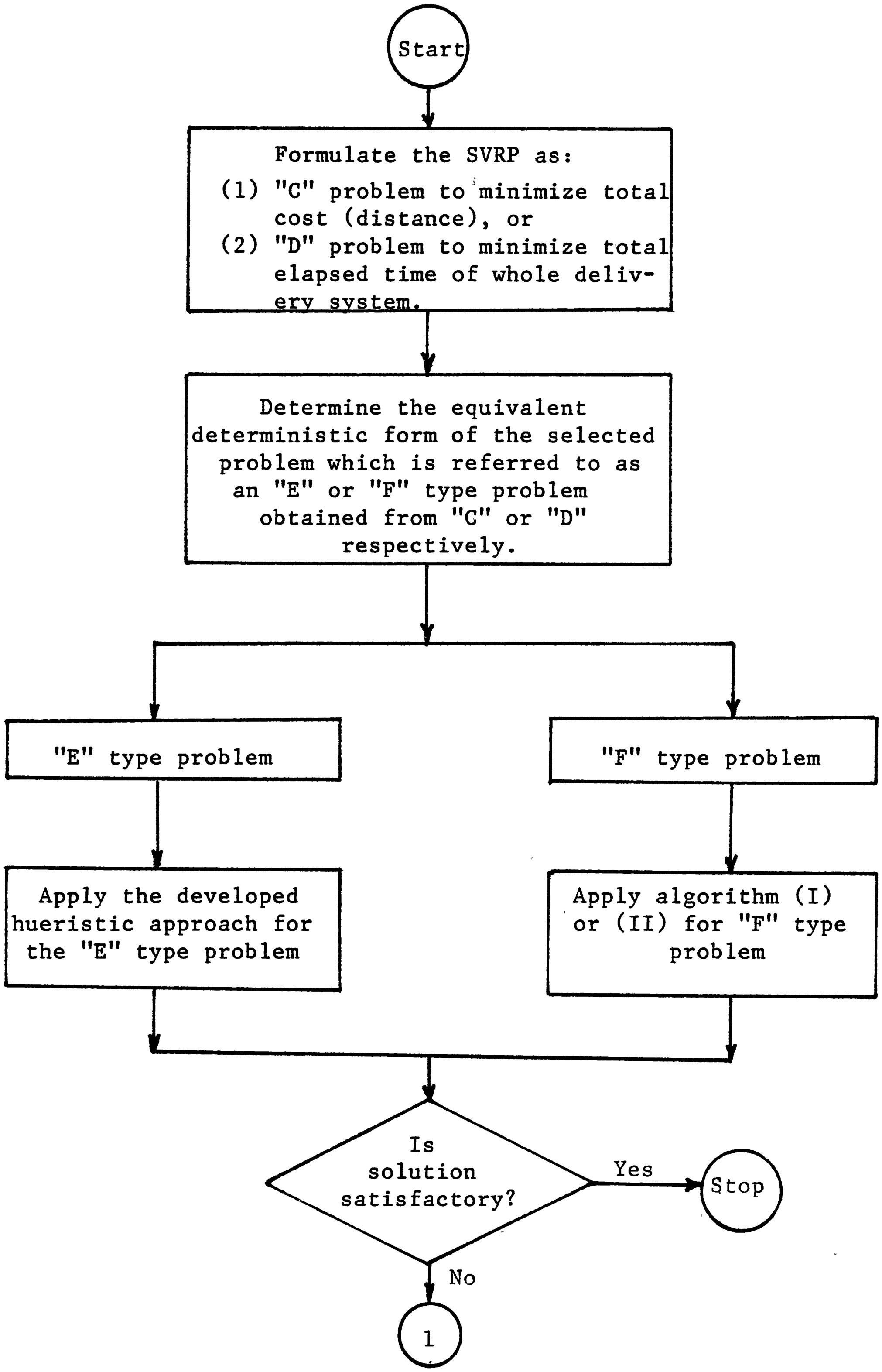

Figure 7. General Structure of the Analysis and Solution of SVRP

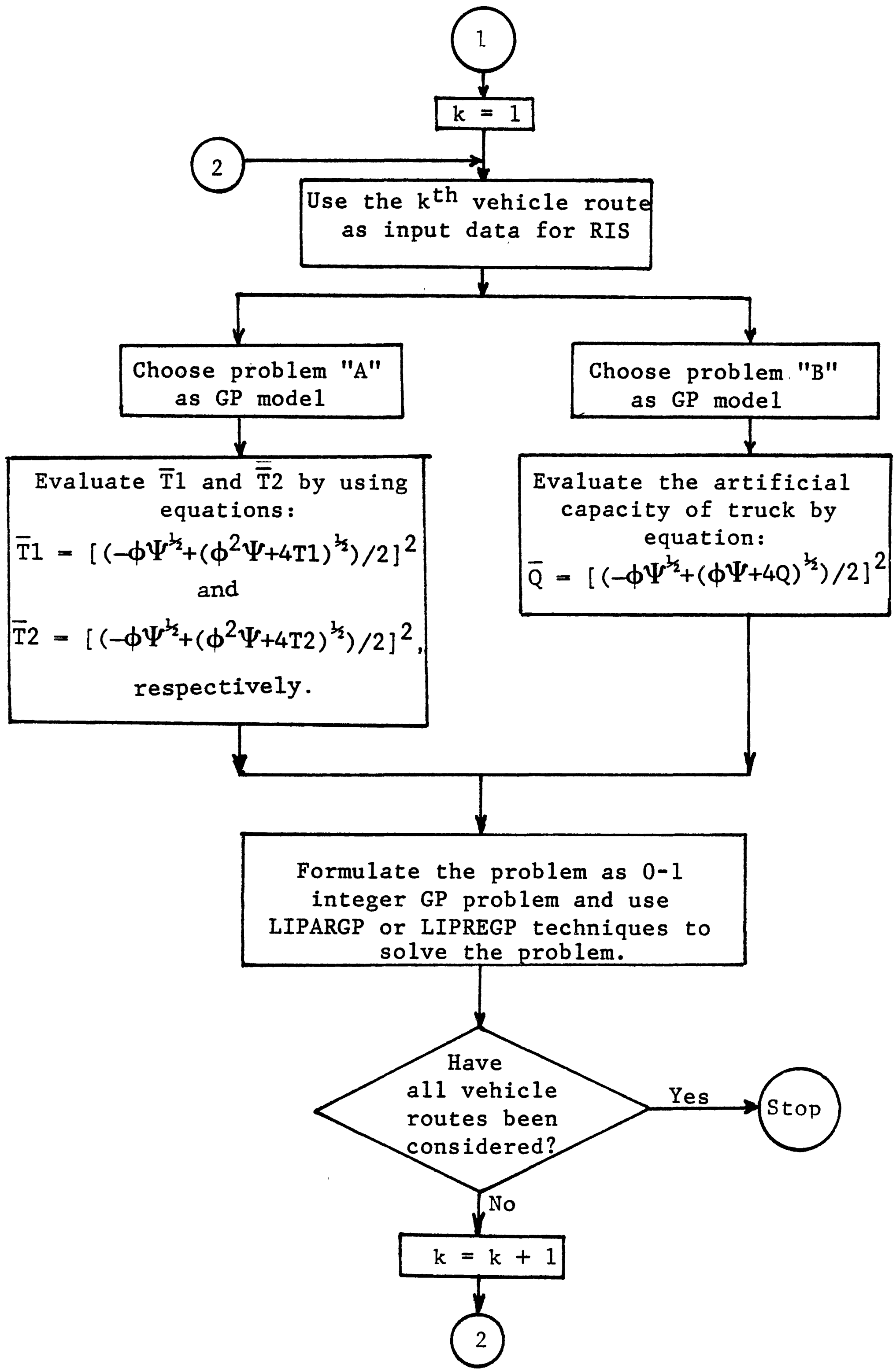

Figure 7. Continued

TABLE II

SUMMARY OF DISTANCES BETWEEN LOCATION IN MILES

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	00															
1	60	00														
2	27	63	00													
3	27	36	45	00												
4	87	75	63	90	00											
5	48	42	33	48	52	00										
6	102	45	99	81	78	66	00									
7	90	54	78	81	29	45	39	00								
8	72	39	57	60	39	27	45	21	00							
9	51	63	27	60	36	21	57	48	42	00						
10	24	36	30	18	72	30	78	69	48	42	00					
11	48	39	69	21	105	66	81	93	72	81	39	00				
12	63	21	75	36	96	60	57	75	60	81	45	24	00			
13	45	15	48	27	69	30	57	54	36	48	21	39	30	00		
14	66	60	45	69	21	21	72	39	27	21	51	87	78	48	00	
15	81	27	78	63	66	48	21	33	27	66	57	66	45	36	48	00

Source: Skitt, R. A. and Levary, R. R. "Vehicle Routing Via Column Generation." European Journal of Operational Research, Vol. 21 (1985), p. 72.

TABLE III

MEAN TRAVEL TIME BETWEEN LOCATIONS IN MINUTES*

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	00															
1	78	00														
2	38	92	00													
3	36	47	59	00												
4	111	104	80	115	00											
5	62	56	40	60	62	00										
6	146	66	132	99	146	66	00									
7	125	69	95	101	125	69	56	00								
8	96	58	73	75	96	58	66	25	00							
9	61	80	35	77	61	80	82	62	52	00						
10	35	43	41	25	35	43	95	98	68	59	00					
11	65	51	88	36	65	51	106	133	90	103	56	00				
12	78	30	90	49	79	30	71	98	82	103	66	34	00			
13	58	19	62	38	58	19	73	79	47	61	27	51	39	00		
14	90	73	55	96	90	73	100	51	37	29	61	119	94	64	00	
15	108	34	93	90	108	34	28	40	37	79	68	94	56	46	68	00

Source: Skitt, R. A. and Levary, R. K. "Vehicle Routing Via Column Generation." European Journal of Operational Research, Vol. 21 (1985) p. 73.

*The original data has been multiplied by 60 and rounded off to the nearest integer value.

TABLE IV

SUMMARY OF DEMAND AND UNLOAD TIME
FOR EACH DEMAND POINT

	Demand		Unload Time (minutes)	
Demand Point	Mean	Variance	Mean	Variance
1	30	30	15	15
2	29	29	15	15
3	16	16	8	8
4	12	12	6	6
5	37	37	18	18
6	17	17	9	9
7	6	6	4	4
8	22	22	11	11
9	20	20	9	9
10	33	33	16	16
11	11	11	7	7
12	37	37	18	18
13	10	10	11	11
14	14	14	7	7
15	25	25	15	15

Tables II, III, and IV summarize the necessary data for whole stations. Tables II and III give the summary of distance between locations in miles and travel time between locations in minutes, respectively. Table IV gives the mean and variance of demand and unload time at each demand point. Additional conditions are as follows:

1. Total unload time for each vehicle route is restricted to 120 minutes,
2. Total travelled time for each route is limited to 480 minutes,
3. Capacity of each truck is 80 units,
4. $\alpha_k = 0.1$, $\beta_k = \eta_k = 0.05$, and
5. The DM's requirements are given in details in Section 7.5.2. This is because the DM can set the goal priority levels based on the results of the RCS of the problem.

The process of solution for this example problem is divided into four parts. In the first part, this problem is treated as an "E" type problem where minimization of the total travelled distance of the whole delivery system is considered as a criterion. The second part treats this problem as a "F" type problem considering the minimization of the total elapsed time of the whole delivery system as a criterion. The third part deals with the utilization of the developed LIGP technique to improve the sequence of stations on the constructed vehicle routes to meet the DM's requirements. The fourth part of the solution process is concerned with the sensitivity analysis of the results as were theoretically investigated in Chapter V.

The first and second parts of the solution process are illustrated in Section 7.5.1 while the third and fourth parts are discussed in Sections 7.5.2 and 7.6, respectively.

7.5.1 Route Construction Stage

If the above conditions along with all other assumptions being employed in this research apply to this example problem, then the "E" and "F" type problems can be solved by applying the proposed algorithms in order to determine the most satisfactory solution.

The solution to the "E" type problem, as formulated in Chapter V, is obtained via the computer program where constructed vehicle routes are {0, 4, 7, 6, 15, 0}, {0, 9, 14, 8, 13, 0}, {0, 1, 12, 0}, {0, 10, 3, 11, 0}, and {0, 2, 5, 0}. The total distance, travel time, unload time, and customer demand of each vehicle route are shown in Table V. The total travelled distance is 810 miles and total travel and unload times are 1,240 and 215 minutes, respectively.

The next attempt was to solve the example problem (7.5) using the concept of the "F" type problem, as described in Chapter V. Two algorithms, (I) and (II), were employed with $\gamma = 0.90$ and $\delta = 0.50$ for these procedures, respectively. The constructed vehicle routes by algorithm (I) are {0, 14, 7, 6, 15, 0}, {0, 1, 12, 0}, {0, 11, 4, 5, 0}, {0, 9, 8, 13, 0}, {0, 2, 0}, and {0, 3, 10, 0} while the constructed vehicle routes by algorithm (II) are {0, 14, 7, 6, 15, 0}, {0, 1, 12, 0}, {0, 9, 4, 10, 0}, {0, 2, 8, 13, 0}, and {0, 5, 11, 3, 0}. The details of the results in the case of the "F" type problem using algorithm (I) are given in Table VI. The results obtained in the case of "F" type problem using algorithm (II) are given in Table VII. The results show that the number of constructed vehicle routes in the case of the "F" type problem using algorithm (II) is smaller than that obtained from the algorithm (I). Therefore, one can verify Theorem (5.7) by comparing the total

TABLE V

SUMMARY OF RESULTS FOR "E" TYPE PROBLEM USING $\alpha_k = 0.1$, $\beta_k = \eta_k = 0.05$ for all k

Route Number	Distance (Mile)	Travel Time (minute)	Unload Time (minute)	Demand
1	267	454	43	72
2	180	251	48	79
3	144	203	42	80
4	111	177	40	72
5	108	155	42	79
Total	810	1,240	215	382

demand, travel, and unload times that are generated by algorithms (I) and (II) of the "F" type problem.

The results obtained from this simple example by the heuristic approaches of the "E" and "F" type problems can be compared using tables V, VI, and VII. The number of constructed vehicle routes using the "E" type problem is equal to that obtained by the algorithm (II) of the "F" type problem. However, the number of constructed vehicle routes by algorithm (I) is larger than that obtained by algorithm (II). Tables VI and VII indicate that the total travel and unload times and total generated demand by algorithms (I) and (II) are not equal. This is mainly because of the difference in the number of vehicle routes. On the other hand, the total travel and unload times and total demands obtained by algorithm (I) of the "F" type problem is larger than that obtained by the heuristic approach of the "E" type problem and algorithm (II) of the "F" type problem.

TABLE VI

SUMMARY OF RESULTS FOR THE "F" TYPE PROBLEM USING ALGORITHM (I) WHERE $\alpha_k = 0.1$, $\beta_k = \eta_k = 0.05$ ($\gamma = 0.9$)

Route	Travel Time (minute)	Unload Time (minute)	Demand
1	356	44	74
2	203	42	80
3	274	40	72
4	236	40	63
5	108	32	60
6	87	21	37
Total	1,264	219	386

TABLE VII

SUMMARY OF RESULTS FOR THE "F" TYPE PROBLEM USING ALGORITHM (II) WHERE $\alpha_k = 0.10$, $\beta_k = \eta_k = 0.05$ ($\delta = 0.5$)

Route Number	Travel Time (minute)	Unload Time (minute)	Demand
1	356	44	74
2	203	42	80
3	209	40	78
4	234	47	73
5	202	42	77
Total	1,204	215	382

7.5.2 Route Improvement Stage

When the arrangement of stations on one route do not exactly or even partially meet the decision maker's needs, then the route improvement technique should be employed. This stage of the problem is required to sequence the stations on each route for the purpose of meeting the customer's and decision maker's criteria. The stations on each vehicle route are sequenced by using the LIGP technique. Prior to the utilization of this technique, the decision maker should consider the following information from the RCS:

1. constructed routes,
2. total demand of each route,
3. total expected cost, time or distance for the whole delivery system,
4. unload and travel times for each route, and
5. number of required vehicles.

In order to demonstrate the application of the route improvement stage of the problem that involves multiple conflicting goals, routes number 1 and 2 from the set of routes of the "E" type problem have been selected. The set of goals and priorities assigned to these routes are described in Sections 7.5.2.1 and 7.5.2.2, respectively. It is assumed that the sequence of stations on routes 3, 4, and 5 meet the decision maker's criteria.

7.5.2.1 Goals and Priorities for Route 1 of "E" Type Problem

Of course the primary objective is to reconstruct a new feasible route. Therefore, route feasibility has been given the first priority. Other priorities are as listed below (Problem B from Section 5.4.2):

P_2 - To minimize total travelled distance to 267 miles

P_3 - To minimize total travel time to 450 minutes and total unload time to 50 minutes

P_4 - To minimize the route safety stock

P_5 - To meet the dependency condition for station 7 which is to be served after station 6.

Before the formulation of Multiple Objective GP begins, one should use equation (5.84) from Theorem 5.2 in order to calculate the artificial capacity of truck, $\bar{Q}$:

$$\bar{Q} = [(-\phi\Psi^{1/2} + (\phi^2\Psi + 4Q)^{1/2})/2]^2 \qquad (7.17)$$

where $\phi = N^{-1}(1 - \eta_k)$ and Ψ is as previously defined. By substituting $Q = 80$ units, $\Psi = 1$, and $\phi = 1.645$ [34, pp. 592-593] in the above equation, an artificial truck capacity of 66 units will be obtained.

The problem can be formulated as:

$$\text{Min} \quad P_1 \left[\sum_{i=1}^{10} (n_i + p_i) + \sum_{i=11}^{22} p_i \right] + P_2 (p_{23}) +$$

$$P_3 (p_{24} + p_{27}) + P_4 (p_{25}) + P_5 (p_{26} + n_{26})$$

Subject to:

$$X_{0,4} + X_{0,7} + X_{0,6} + X_{0,15} + n_1 - p_1 = 1$$

$$X_{4,0} + X_{4,7} + X_{4,6} + X_{4,15} + n_2 - p_2 = 1$$

$$X_{7,0} + X_{7,4} + X_{7,6} + X_{7,15} + n_3 - p_3 = 1$$

$$X_{6,0} + X_{6,4} + X_{6,7} + X_{6,15} + n_4 - p_4 = 1$$

$$X_{15,0} + X_{15,6} + X_{15,7} + X_{15,4} + n_5 - p_5 = 1$$

$$X_{4,0} + X_{7,0} + X_{6,0} + X_{15,0} + n_6 - p_6 = 1$$

$$X_{0,4} + X_{7,4} + X_{6,4} + X_{15,4} + n_7 - p_7 = 1$$

$$X_{0,7} + X_{4,7} + X_{6,7} + X_{15,7} + n_8 - p_8 = 1$$

$$X_{0,6} + X_{4,6} + X_{7,6} + X_{15,6} + n_9 - p_9 = 1$$

$$X_{0,15} + X_{4,15} + X_{7,15} + X_{6,15} + n_{10} - p_{10} = 1$$

$$Z_1 - Z_2 + 5X_{4,7} + n_{11} - p_{11} = 4$$

$$Z_1 - Z_3 + 5X_{4,6} + n_{12} - p_{12} = 4$$

$$Z_1 - Z_4 + 5X_{4,15} + n_{13} - p_{13} = 4$$

$$Z_2 - Z_1 + 5X_{7,4} + n_{14} - p_{14} = 4$$

$$Z_2 - Z_3 + 5X_{7,6} + n_{15} - p_{15} = 4$$

$$Z_2 - Z_4 + 5X_{7,15} + n_{16} - p_{16} = 4$$

$$Z_3 - Z_1 + 5X_{6,4} + n_{17} - p_{17} = 4$$

$$Z_3 - Z_2 + 5X_{6,7} + n_{18} - p_{18} = 4$$

$$Z_3 - Z_4 + 5X_{6,15} + n_{19} - p_{19} = 4$$

$$Z_4 - Z_1 + 5X_{15,4} + n_{20} - p_{20} = 4$$

$$Z_4 - Z_2 + 5X_{15,7} + n_{21} - p_{21} = 4$$

$$Z_4 - Z_3 + 5X_{15,6} + n_{22} - p_{22} = 4$$

$$87X_{0,4} + 90X_{0,7} + 102X_{0,6} + 81X_{0,15} + 87X_{4,0} + 39X_{4,7} +$$

$$78X_{4,6} + 66X_{4,15} + 90X_{7,0} + 39X_{7,4} + 39X_{7,6} + 33X_{7,15} +$$

$$102X_{6,0} + 78X_{6,4} + 39X_{6,7} + 21X_{6,15} + 81X_{15,0} + 66X_{15,4} + 33X_{15,7} + 21X_{15,6} + n_{23} - p_{23} = 267$$

$$6X_{0,4} + 6X_{7,4} + 6X_{6,4} + 6X_{15,4} + 4X_{0,7} + 4X_{4,7} + 4X_{6,7} + 4X_{15,7} + 9X_{0,6} + 9X_{4,6} + 9X_{7,6} + 9X_{15,6} + 15X_{0,15} + 15X_{4,15} + 15X_{7,15} + 15X_{6,15} + n_{24} - p_{24} = 50$$

$$12X_{0,4} + 12X_{7,4} + 12X_{6,4} + 12X_{15,4} + 6X_{0,7} + 6X_{4,7} + 6X_{6,7} + 6X_{15,7} + 17X_{0,6} + 17X_{4,6} + 17X_{7,6} + 17X_{15,6} + 25X_{0,15} + 25X_{4,15} + 25X_{7,15} + 25X_{6,15} + n_{25} - p_{25} = 66$$

$$X_{6,7} + n_{26} - p_{26} = 1$$

$$111X_{0,4} + 125X_{0,7} + 146X_{0,6} + 108X_{0,15} + 111X_{4,0} + 125X_{4,7} + 146X_{4,6} + 108X_{4,15} + 125X_{7,0} + 125X_{7,4} + 56X_{7,6} + 40X_{7,15} + 146X_{6,0} + 146X_{6,4} + 56X_{6,7} + 28X_{6,15} + 108X_{15,0} + 108X_{15,4} + 40X_{15,7} + 28X_{15,6} + n_{27} - p_{27} = 450$$

An attempt was made to evaluate the 0-1 integer solution of the above problem. Therefore, constraints of the following type

$$X_i + n_i - p_i = 1 \qquad \forall_i, \ i = 1, \ldots, 20,$$

and an absolute priority level of

$$P_0 = \sum_{i=1}^{20} p_{i+20}$$

were added to the original problem. The resulting problem consisted of 47 constraints, 118 variables (decision and deviational variables), and 6 priority levels. The absolute priority level was considered as the first priority and the original priorities of the problem were downgraded by one level. This problem was solved by the LIPARGP technique using the branch and bound method. Two new integer solutions, illustrated in the form of routes {0, 15, 6, 7, 4, 0} and {0, 6, 7, 15, 4, 0}, were obtained. Other computer results are illustrated in Table VIII. Route {0, 15, 6, 7, 4, 0} which satisfies the decision maker's criteria is the final solution to this problem.

TABLE VIII

SUMMARY OF RESULTS FOR ROUTE 1 OF "E" TYPE PROBLEM AFTER EMPLOYMENT OF THE RIS OF THE PROBLEM

Solution Number	Sequence of locations on each new route	Distance	Unload Time (minute)	Travel Time (minute)
1	{0, 15, 6, 7, 4, 0}	267	34	428
2	{0, 6, 7, 15, 4, 0}	327	34	461

7.5.2.2 Goals and Priorities for Route 2 of "E" Type Problem

As expected, the route feasibility is given the first priority level where other priorities are (Problem A from Section 5.4.1):

P_2 - To minimize total travelled distance of each vehicle route to 180 miles,

P_3 - To minimize the unload time of vehicle route to $\bar{T}_2$ minutes and travel time of vehicle route to $\bar{T}_1$ minutes

P_4 - To meet the dependency conditions such that Station 8 follows Station 9.

The third priority level requires the minimization of travel and unload times to the levels of $\bar{T}_1$ and $\bar{T}_2$, respectively; hence, prior to the formulation of this Multiple Objective GP, $\bar{T}_1$ and $\bar{T}_2$ must be calculated. Equation (7.17) can be used again for the calculation of $\bar{T}_1$. Q and ϕ are substituted by 480 and 1.285, respectively. Similarly, in the evaluation of $\bar{T}_2$, Q and ϕ can be substituted with 120 and 1.645, respectively. The overall formulation of the problem for Route 2 of the "E" type problem is:

$$\text{Min} \quad P_1 \left[\sum_{i=1}^{10} (n_i + p_i) + \sum_{i=11}^{22} (p_i) \right] + P_2 \, (p_{23}) +$$

$$P_3 \, (p_{24} + p_{25}) + P_4 \, (n_{26} + p_{26})$$

Subject to:

$$X_{0,13} + X_{0,8} + X_{0,14} + X_{0,9} + n_1 - p_1 = 1$$
$$X_{13,0} + X_{13,8} + X_{13,14} + X_{13,9} + n_2 - p_2 = 1$$
$$X_{8,0} + X_{8,13} + X_{8,14} + X_{8,9} + n_3 - p_3 = 1$$
$$X_{14,0} + X_{14,13} + X_{14,8} + X_{14,9} + n_4 - p_4 = 1$$
$$X_{9,0} + X_{9,13} + X_{9,8} + X_{9,14} + n_5 - p_5 = 1$$

$$X_{13,0} + X_{8,0} + X_{14,0} + X_{9,0} + n_6 - p_6 = 1$$
$$X_{0,13} + X_{8,13} + X_{14,13} + X_{9,13} + n_7 - p_7 = 1$$
$$X_{0,8} + X_{13,8} + X_{14,8} + X_{9,8} + n_8 - p_8 = 1$$
$$X_{0,14} + X_{13,14} + X_{8,14} + X_{9,14} + n_9 - p_9 = 1$$
$$X_{0,9} + X_{13,9} + X_{8,9} + X_{14,9} + n_{10} - p_{10} = 1$$

$$Z_1 - Z_2 + 5X_{13,8} + n_{11} - p_{11} = 4$$
$$Z_1 - Z_3 + 5X_{13,14} + n_{12} - p_{12} = 4$$

$Z_1 - Z_4 + 5X_{13,9} + n_{13} - p_{13} = 4$

$Z_2 - Z_1 + 5X_{8,13} + n_{14} - p_{14} = 4$

$Z_2 - Z_3 + 5X_{8,14} + n_{15} - p_{15} = 4$

$Z_2 - Z_4 + 5X_{8,9} + n_{16} - p_{16} = 4$

$Z_3 - Z_1 + 5X_{14,13} + n_{17} - p_{17} = 4$

$Z_3 - Z_2 + 5X_{14,8} + n_{18} - p_{18} = 4$

$Z_3 - Z_4 + 5X_{14,9} + n_{19} - p_{19} = 4$

$Z_4 - Z_1 + 5X_{9,13} + n_{20} - p_{20} = 4$

$Z_4 - Z_2 + 5X_{9,8} + n_{21} - p_{21} = 4$

$Z_4 - Z_3 + 5X_{9,14} + n_{22} - p_{22} = 4$

$45X_{0,13} + 72X_{0,8} + 66X_{0,14} + 51X_{0,9} + 45X_{13,0} +$
$36X_{13,8} + 48X_{13,14} + 48X_{13,9} + 72X_{8,0} + 36X_{8,13} +$
$27X_{8,14} + 42X_{8,9} + 15X_{14,0} + 48X_{14,13} + 27X_{14,8} +$
$21X_{14,9} + 51X_{9,0} + 48X_{9,13} + 42X_{9,8} + 21X_{9,14} +$
$n_{23} - p_{23} = 180$

$11X_{0,13} + 11X_{8,13} + 11X_{14,13} + 11X_{9,13} + 11X_{0,8} +$
$11X_{13,8} + 11X_{14,8} + 11X_{9,8} + 7X_{0,14} + 7X_{13,14} +$
$7X_{8,14} + 7X_{9,14} + 9X_{0,9} + 9X_{13,9} + 9X_{8,9} + 9X_{14,9} +$
$n_{24} - p_{24} = 111$

$58X_{0,13} + 96X_{0,8} + 90X_{0,14} + 61X_{0,9} + 58X_{13,0} +$
$47X_{13,8} + 64X_{13,14} + 61X_{13,9} + 96X_{8,0} + 47X_{8,13} +$
$37X_{8,14} + 52X_{8,9} + 90X_{14,0} + 64X_{14,13} + 37X_{14,8} +$
$29X_{14,9} + 61X_{9,0} + 61X_{9,13} + 52X_{9,8} + 29X_{9,14} +$
$n_{25} - p_{25} = 452$

$X_{9,8} + n_{26} - p_{26} = 1$

A similar procedure for calculation of 0-1 integer solution for this problem was used. After the addition of all necessary constraints and the absolute priority level of P_0, the total number of constraints, variables, and priorities became 46, 116, and 5, respectively. This problem was solved by LIPARGP technique and a new solution illustrated in forms of route {0, 14, 9, 8, 13, 0} was obtained.

The characteristics of this vehicle route are:

total travelled distance = 210 miles

total unload time = 38 minutes

total travel time = 276 minutes

These results indicate that all decision maker's criteria, except the total travelled distance, have been achieved. However, it can be conclude that an increase of 30 miles in the total travelled distance has been sacrificed for the achievement of other goals set by the decision maker.

The final solution to this example problem, according to the decision maker's criteria and customer's requirements, is summarized in Table IX.

7.6 Sensitivity of Elapsed Time Upon The Probability of Route Failures

Thus far, the basic concepts of the SVRP and the derivation of solution methods have been the main objective of this research. However, an important part of any solution process is the analysis of the parameter changes after the final solution has been determined. This technique is defined as the sensitivity analysis of the procedure. The

TABLE IX

THE SUMMARY OF FINAL RESULTS OF EXAMPLE PROBLEM BASED ON THE "E" TYPE PROBLEM ANALYSIS

Route Number	Distance (mile)	Travel Time (minute)	Unload Time (minute)	Demand
1	267	428	34	72
2	210	276	38	79
3	144	203	42	80
4	111	177	40	72
5	108	155	42	79
Total	840	1,239	196	382

degree of uncertainty in real world problems such as demands, travel and unload times, shipment, and costs has increased the utilization of sensitivity analysis in the decision making environments. Obviously, forecasting techniques can be used to predict the future values of the important parameters of the problem when the final solution is relatively sensitive to these factors.

The purpose of this section is to introduce some ideas concerning the analysis of elapsed time in the SVRP due to route failure probabilities. To illustrate this idea, it is necessary to review the example problem presented in the previous section. In the example, the route failure probabilities were considered to be $\alpha_k = 0.1$, $\beta_k = \eta_k = 0.05$. Now, what will be the number of vehicle routes, travel time, unload time, and travelled distance using different values of route failure probabilities by employing the "E" and "F" type problems?

The data and all necessary information of the example problem has been used in the solution process of the "E" type problem. The results are shown in Tables X and XI. Table X gives the summary of results when $\beta_k = \eta_k$ and α_k accept different values. On the other hand, Table XI illustrates the results for the "E" type problem when η_k is fixed and $\alpha_k = \beta_k$ accepts different probability levels. The number of vehicle routes for this example using the "E" type problem under these probability levels is 5.

After solving the "E" type problem, the next objective was to solve the example problem (7.5) using the "F" type problem solution procedure where $\gamma = 0.90$. Table XII provides the travel time, unload time, and total elapsed time of each vehicle route developed by algorithm (I), where α_k accepts different values and $\beta_k = \eta_k$ are fixed. Table XIII provides the travel time, unload time, and total elapsed time of each vehicle route developed by algorithm (I) for fixed η_k and various probability levels for $\alpha_k = \beta_k$ where $\gamma = 0.90$.

The example problem (7.5) was solved by the "F" type problem using the algorithm (II) where $\delta = 0.50$. These results are shown in Tables XIV and XV. However, Tables X, XII, and XIV indicate that if α_k increases, then the travel time of each vehicle route decreases. For instance, Table XI shows that by increasing α_k from 0.05 to 0.1, then the travel time of routes 1 and 2 decrease from 462 to 454 and 257 to 251, respectively. Tables XI, XIII, and XV support the results of Theorem 5.5, which has been proved in Chapter V. This means, for example, that if α_k and β_k increase such that $\alpha_k = \beta_k$, then the travel and unload times of each vehicle route will decrease.

TABLE X

SUMMARY OF RESULTS FOR THE "E" TYPE PROBLEM
FOR $\beta_k = \eta_k = 0.05$

Route Number	α_k	Travel Time (minutes)	Unload Time (minutes)
1	0.05	462	43
2		257	48
3		208	42
4		181	40
5		159	42
1	0.10	454	43
2		251	48
3		203	42
4		177	40
5		155	42
1	0.30	438	43
2		239	48
3		193	42
4		167	40
5		146	42

TABLE XI

SUMMARY OF RESULTS FOR THE "E" TYPE PROBLEM
FOR η_k = 0.05

Route Number	α_k	β_k	Travel Time (minutes)	Unload Time (minutes)
1	0.05	0.05	462	43
2			257	48
3			208	42
4			181	40
5			159	42
1	0.1	0.1	454	41
2			251	45
3			203	40
4			177	38
5			155	40
1	0.30	0.30	438	37
2			239	41
3			193	36
4			167	33
5			146	36

TABLE XII

SUMMARY OF RESULTS FOR THE "F" TYPE PROBLEM USING ALGORITHM (I) WHERE $\beta_k = \eta_k = 0.05$ AND $\gamma = 0.90$

Route Number	α_k	Travel Time (minutes)	Unload Time (minutes)	Total Elapsed Time of Each Route (minutes)
1	0.025	368	44	412
2		212	42	254
3		285	40	325
4		246	40	286
5		115	32	147
6		93	21	114
1	0.10	356	44	400
2		203	42	245
3		274	40	314
4		236	40	276
5		108	32	140
6		87	21	108
1	0.30	342	44	386
2		193	42	235
3		262	40	302
4		225	40	265
5		101	32	133
6		80	21	101

TABLE XIII

SUMMARY OF RESULTS FOR THE "F" TYPE PROBLEM USING ALGORITHM (II) WHERE $\eta_k = 0.05$ AND $\gamma = 0.90$

Route Number	α_k	β_k	Travel Time (minutes)	Unload Time (minutes)	Total Elapsed Time of Each Route (minutes)
1	0.025	0.025	368	46	414
2			212	44	256
3			285	41	326
4			246	41	287
5			115	33	148
6			93	22	115
1	0.10	0.10	356	42	398
2			203	40	243
3			274	38	312
4			236	38	274
5			108	30	138
6			87	19	106
1	0.30	0.30	342	38	380
2			193	36	229
3			262	33	295
4			225	33	258
5			101	26	127
6			80	17	97

TABLE XIV

SUMMARY OF RESULTS FOR THE "F" TYPE PROBLEM USING ALGORITHM (II) WHERE $\beta_k = \eta_k = 0.05$ AND $\delta = 0.50$

Route Number	α_k	Travel Time (minutes)	Unload Time (minutes)	Total Elapsed Time of Each Route (minutes)
1	0.025	368	44	412
2		212	42	254
3		219	40	259
4		244	47	291
5		211	42	253
1	0.10	356	44	400
2		203	42	245
3		209	40	249
4		234	47	281
5		202	42	244
1	0.30	342	44	386
2		193	42	235
3		199	40	239
4		223	47	270
5		192	42	234

TABLE XV

SUMMARY OF RESULTS FOR THE "F" TYPE PROBLEM USING ALGORITHM (II) WHERE $\eta_k = 0.05$ AND $\delta = 0.5$

Route Number	α_k	β_k	Travel Time (minutes)	Unload Time (minutes)	Total Elapsed Time (minutes)
1	0.025	0.025	368	46	414
2			212	44	256
3			219	41	260
4			244	48	292
5			211	44	255
1	0.10	0.10	356	42	398
2			203	40	243
3			209	38	247
4			234	44	278
5			202	40	242
1	0.30	0.30	342	38	380
2			193	36	229
3			199	33	232
4			223	40	263
5			192	36	228

7.7 Summary

A heuristic algorithm based on the concept of Clarke and Wright's saving approach for the "E" type problem has been developed. The development of the heuristic approach for the "F" type problem was related to two new algorithms, (I) and (II). Two functions, one for each algorithm, have been developed and used as the basis of the saving evaluations in the "F" type problem. Two algorithms which consist the saving rules are used for partitioning a set of stations into feasible subsets using the concept of the Clarke and Wright procedure. Algorithms (I) and (II), which were developed in this chapter, have the capability of evaluating the savings for the SVRP where travel times are random variables. Hence, any SVRP with probabilistic customer demand, travel time, and unload time can be solved by employing the proposed heuristic approaches.

A simple example problem (7.5) is employed to illustrate the algorithm procedure. The process of solution of this example were divided into four sections. In the first part, the problem was treated as an "E" type problem, and in the second part as a "F" type problem, as discussed in Chapter V. The appropriate heuristic approaches of these types of problems were employed to design the vehicle routes. The third part of the analysis is related to the utilization of the developed LIGP technique for improving the sequence of stations on the constructed vehicle routes by the RCS of the problem for the "E" type problem. Finally, the example problem (7.5) was analyzed through the sensitivity analysis, as theoretically investigated in Chapter V. The results of this example problem fully support the theoretical background of the sensitivity analysis in relation to the route failure probabilities.

The most important characteristic of the developed algorithms for "E" and "F" type problems is to take into account the decision maker's and the customer's requirements. These procedures allow the decision maker to investigate and make good trade-off decisions concerning any possible criteria in the problem's environment.

CHAPTER VIII

ANALYSIS OF RESULTS

8.1 Introduction

The objective of this chapter is to analyze the results obtained from the "E" and "F" type problems developed in Chapter VII and the SVRP having only probabilistic customer demands. These results will be used to validate the new procedure for the SVRP. Three numerical examples demonstrate the performance of these algorithms. The validity of these procedures is evaluated by comparing the results with those of the existing saving methods for the SVRP having only probabilistic customer demands. A saving method developed by Stewart [54] is selected for the purpose of comparison of the results for the SVRP with only probabilistic customer demand. However, the lack of research in the area of SVRP with probabilistic customer demand and travel and unload times has made a comparison of results for this type of problem impossible. Therefore, the computational results obtained by Algorithms (I) and (II) of the "F" type problem, as described in Chapter VII, are only compared to each other.

Before entering into the analysis of the test problems, it is important to discuss difficulties which may arise due to the utilization of CCP in the SVRP. Specifically, the major difficulty with CCP is the determination of appropriate probability levels for constraints. Obviously, a reasonable approach is to provide a specific range for each probability level for the important constraints and several probability levels for other constraints, then determine the corresponding results. The reason for considering a specific range for probability levels is two-fold. First, a specific range will prevent the problem from becoming too large. Second, the decision maker might not be interested in the whole range of the probability level which is from 0 to 1. The general assumption, of course, is that the manager of a delivery system is able to determine the value of these probability levels because of his or her familiarity, experience, and utilization of the CCP in the SVRP.

Another point that needs to be mentioned concerns the utilization of LIGP in the route improvement stage of the problem. The goal program developed in this research can only solve the linear and linear integer GP problems, For this reason, whenever a GP problem with nonlinear constraints appears to be a feasible option, a nonlinear integer goal program regarding the minimization of the priority levels must be employed. In this case, the GP model should be able to solve a nonlinear problem with 0-1 type decision variables.

In the examples given in this chapter for comparison purposes, one or more of the random variables are considered to be normally distributed. This is done for ease of comparison of the new model with the other models.

8.2 Validity of the New Model

To validate the new model, the results are first verified through hand computations to assure that the results satisfy all specified conditions. Specifically, such verification consists of determining that (1) the total demand of each route and truck capacity agree, (2) each customer is served by only one truck, (3) the total travel time of each vehicle route satisfies the predetermined travel time level, and (4) there is no disagreement with the total unload time of each route and its predetermined level.

Next, the results obtained from this model are compared with those obtained from the saving method developed by Stewart [54]. Stewart's model is selected as a basis for comparison because the new model has similar characteristics provided that customer demands are probabilistic and travel and unload times are deterministic.

8.3 Comparison with the Stewart Model

In this section, two test problems proposed by Stewart [54] are used for the purpose of comparison. The detailed data for these problems are reproduced in Appendix C. The first test problem consists of fifty demand points where customer demands are considered to be normally distributed. The second test problem consists of 75 demand points with normally distributed customer demands. The objective of these two problems is the minimization of the total travelled distance between the stations.

Table XVI compares the results of Clarke and Wright's algorithm for the CCP problem, where η = 0.01, 0.025, 0.10, and 0.15 are considered to be the probability of route failure for customer demand. These results are based on the fifty node problem with truck capacity of 160 units and where customer demands are considered to be normally distributed. The proposed model produced routes requiring the same number as those derived by the Stewart algorithm. The total travel distance of both algorithms are almost identical. Most likely, the small differences between the generated total travelled distance by these two procedures is due to round-off errors in integer calculations.

Table XVII summarizes the computational results of Clark and Wright's algorithm for the CCP problem of 75 demand points (second test problem) for η values of 0.025, 0.05, and 0.10, where the capacity of each truck is considered to be 140 units. The customer demands were assumed to be normally distributed. The detailed data for this problem is shown in Appendix C. The results indicate that the same number of vehicle routes and nearly identical travel distances are obtained by both procedures. It is expected that the route distance will decrease with the increase in the η probability level.

8.4 Validity of the Developed Heuristic Approaches

In this section the validity of the developed heuristic approaches for solving the "E" and "F" type problems is proven. A numerical problem (third test problem) is furnished in order to demonstrate some important points when "E" and "F" type problems are used. The model is solved with the total distance ("E" type problem) and total elapsed time ("F" type problem) as two separate objective functions. The data for this problem is randomly generated [51] with the following characteristics:

1. this problem is an extension of Steward's 50-node problem,
2. customer demands are normally distributed,
3. unload times are poisson distributed such that mean unload time is equal to mean customer demand,

TABLE XVI

COMPARISON OF RESULTS OF THE 50 DEMAND POINTS WITH THE STEWART ALGORITHM FOR THE CHANCE-CONSTRAINED VRP WITH NORMALLY DISTRIBUTED CUSTOMER DEMAND

		Stewart		Proposed Procedure	
Problem Number	η	Distance (Mile)	Number of Routes	Distance (Mile)	Number of Routes
1	0.010	606	7	607	7
2	0.025	596	6	590	6
3	0.100	621	6	622	6
4	0.150	623	6	623	6

Truck Capacity = 160 units

TABLE XVII

COMPARISON OF RESULTS FOR 75 DEMAND POINTS WITH THE STEWART ALGORITHM FOR CHANCE-CONSTRAINED VRP WITH NORMALLY DISTRIBUTED CUSTOMER DEMAND

		Stewart		Proposed Procedure	
Problem Number	η	Distance (Mile)	Number of Routes	Distance (Mile)	Number of Routes
1	0.025	975	13	973	13
2	0.050	---	--	948	12
3	0.100	923	12	923	12

Truck Capacity = 140 units

4. mean travel time between stations i and j is considered to be a linear function of distance between stations i and j. It is evaluated through the following equation, mean of $t_{ij} = 1.2\ d_{ij} + 3$, and

5. the standard deviation, $\sigma_{t_{ij}}$, of travel times was randomly generated using a uniform random number generator so that $\sigma_{t_{ij}}$ fall between zero and 1/4 of the mean travel time of t_{ij}.

In this case, the customer demands and travel and unload times are assumed to be independent of each other. The detailed data for characteristics 1 through 5 are shown in Appendix C.

The following conditions for solving this problem are shown in Table XVIII:

1. truck capacity with values of 140, 160, and 200 units,
2. unload time with values 60, 90, and 120 minutes for each vehicle route in order to solve the "E" type problem,
3. travel time with values 420, 390, and 360 minutes per route for solving the "E" type problem,
4. route failure probabilities of α_k, β_k, and η_k can accept ranges $0 < \alpha_k < 0.20$, $0 < \beta_k < 0.10$, and $0 < \eta_k < 0.10$,
5. in Algorithm (I) of "F" type problem, γ can accept values $0.70 < \gamma < 0.99$, and
6. in Algorithm (II) of "F" type problem, δ can accept values $0.50 < \delta < 4.0$.

The amount of truck capacity, maximum value of travel and unload times per each vehicle route, value of probability levels, and other factors such as γ and δ are chosen arbitrarily. A set of nine subproblems considering different combinations and using previous conditions are designed and illustrated in Table XVIII.

TABLE XVIII

SUMMARY OF DATA FOR TEST PROBLEM NUMBER 3

Subproblem Number	Truck Capacity	Mean Unload Time (minutes)	Travel Time (minutes)	α_k	β_k	η_k
1	140	60	420	0.05	0.10	0.025
2	140	60	420	0.10	0.10	0.025
3	140	60	420	0.20	0.10	0.025
4	160	90	390	0.05	0.10	0.025
5	160	90	390	0.10	0.10	0.025
6	160	90	390	0.20	0.10	0.025
7	200	120	360	0.05	0.025	0.01
8	200	120	360	0.10	0.050	0.05
9	200	120	360	0.20	0.10	0.10

The purpose of this section is to solve these nine subproblems by treating them in the following categories:

Category 1. "E" type problem,

Category 2. "F" type problem using Algorithm (I) with $\gamma = 0.90$,

Category 3. "F" type problem using Algorithm (II) with $\delta = 0.50$.

Eighteen additional subproblems are solved by treating subproblem 9 as

Category 4. "F" type problem using Algorithm (I) with $\gamma = 0.70$, 0.80, 0.82, 0.85, 0.87, 0.92, 0.95, 0.97 and 0.99, and

Category 5. "F" type problem using Algorithm (II) with $\delta = 0.40$, 1.0, 1.5, 1.7, 2.0, 2.3, 2.5, 3.0, and 4.0.

As can be seen, a grand total of 45 subproblems will be solved using the data and information of this example problem.

Two procedures have been employed to solve this test problem. First, it is treated as an "E" type problem where the minimization of the total travelled distance is the criterion. Second, it is treated as a "F" type problem using the minimization of total elapsed time of the whole delivery system as a criterion. It is important to note that the upper bounds for travel and unload times for each route, as given in Table XVIII, are not used in the solution process of the problem when the "F" type procedure is employed. Therefore, the parameters to be considered at the time of analysis of results of the "F" type problem are truck capacity, probability levels α_k, β_k, and η_k, and other factors such as γ and δ.

<u>8.4.1 Results of Category 1</u>

Table XIX illustrates the minimization of the total travel distance of the delivery system under the existence of travel and unload time constraints and truck capacity (Category 1). The results indicate that by increasing the probability levels of α, β, and η the travel and unload times of each vehicle route and consequently the total elapsed time of the whole delivery system decreases. The number of vehicle routes and total travelled distance of the whole delivery system decreased from 17 to 8 routes and 1036 to 675 miles as the vehicle capacity increased from 140 to 200, respectively.

8.4.2 Results of Category 2

Table XX demonstrates the results of the nine subproblems for the "F" type problem using Algorithm (I) where $\gamma = 0.90$ (Category 2). The table shows that a better solution can be obtained in terms of minimum number of vehicles and minimum travel and unload times when this procedure is selected. It also shows that increasing truck capacity decreases the number of vehicle routes. The final observation is that when α_k (the route failure probability for travel time) increases, the total travel time of the whole delivery system decreases.

8.4.3 Results of Category 3

Table XXI, which is self explanatory, describes the improvement process in detail. The results show that the objective function which measures the total elapsed time of the whole delivery system has improved in all cases. It is important to note that the number of vehicle routes decreased as the truck capacity increased.

8.4.4 Results of Category 4

Table XXII shows the results of subproblem 9 where γ has increased from 0.70 to 0.99. The number of vehicle routes and total unload times is fixed for all cases. The total travel times decrease as the value of γ increases from 0.70 to 0.99. It is interesting to note

TABLE XIX

SUMMARY OF RESULTS OF "E" TYPE PROBLEM
(CATEGORY 1)

Problem Number	Distance (Mile)	Travel Time (Minutes)	Unload Time (Minutes)	Total Elapsed Time (Minutes)	Number of Routes
1	1036	1474	918	2392	17
2	1036	1456	918	2374	17
3	1036	1436	918	2354	17
4	783	1140	891	2031	11
5	783	1128	891	2019	11
6	783	1113	891	2004	11
7	675	994	876	1870	8
8	695	1010	903	1913	8
9	675	973	876	1849	8

TABLE XX

SUMMARY OF RESULTS FOR THE "F" TYPE PROBLEM
USING ALGORITHM (I) WHERE $\gamma = 0.90$
(CATEGORY 2)

Problem Number	Total Travel Time (minutes)	Total Unload Time (minutes)	Total Elapsed Time (minutes)	Number of Routes
1	964	867	1831	7
2	953	867	1820	7
3	942	867	1809	7
4	959	861	1820	6
5	949	861	1810	6
6	939	861	1800	6
7	902	896	1798	5
8	878	876	1754	5
9	861	853	1714	5

TABLE XXI

SUMMARY OF RESULTS FOR THE "F" TYPE PROBLEM
USING ALGORITHM (II) WHERE $\delta = 0.50$
(CATEGORY 3)

Problem Number	Total Travel Time (minutes)	Total Unload Time (minutes)	Total Elapsed Time (minutes)	Number of Routes
1	995	872	1867	8
2	986	872	1858	8
3	972	872	1844	8
4	965	861	1826	6
5	955	861	1816	6
6	946	861	1807	6
7	944	853	1797	5
8	911	876	1787	5
9	895	854	1749	5

TABLE XXII

SOLUTION OF SUBPROBLEM 9 USING ALGORITHM (I)
WITH VARIOUS VALUES OF γ
(CATEGORY 4)

γ	Travel Time (minutes)	Unload Time (minutes)	Total Elapsed Time (minutes)	Number of Routes
0.70	877	853	1730	5
0.80	886	854	1740	5
0.82	877	853	1730	5
0.85	877	853	1730	5
0.87	877	853	1730	5
0.90	861	853	1714	5
0.92	860	853	1713	5
0.95	860	853	1713	5
0.97	860	853	1713	5
0.99	860	853	1713	5

that the total travelled time remains constant when γ increases from 0.82 to 0.87 and from 0.92 to 0.99.

8.4.5 -Results of Category 5

Table XXIII illustrates the results of subproblem 9 where δ increases from 0.40 to 4.0. The amount of travel time remains constant as the value of δ increases from 1.0 to 2.3 and from 2.5 to 4.0. A total travel time of 887 minutes is obtained as δ increases from 1.0 to 2.3. No change in the number of vehicle routes occurred as δ increased from 0.40 to 4.0. The results from Tables XXII and XXIII indicate that the amount of total elapsed time provided by Algorithms (I) and (II) of the "F" type problem equalize as γ and δ both are assigned large values. It is therefore concluded that Algorithms (I) and (II) are closely related and that the results of these algorithms can be used for the purpose of comparison.

8.5 Summary

Several aspects of the SVRP have been analyzed in this chapter. The computational experience of the proposed procedure on three test problems has been presented, and the computational results of a SVRP having only probabilistic customer demands on two test problems has been compared with the available procedure from the literature. It has been shown that a SVRP can be treated as both "E" and "F" type problems. Usually the "E" type problem is expected to produce a larger number of vehicle routes because the objective function measures the total travelled distance with restrictions on travel and unload times and truck capacity. On the other hand, the "F" type problem measures the total

TABLE XXIII

SOLUTION OF SUBPROBLEM 9 USING ALGORITHM (I)
WITH VARIOUS VALUES OF δ
(CATEGORY 5)

δ	Travel Time (minutes)	Unload Time (minutes)	Total Elapsed Time (minutes)	Number of Routes
0.40	899	853	1752	5
0.50	895	854	1749	5
1.00	887	854	1741	5
1.50	887	854	1741	5
1.70	887	854	1741	5
2.00	887	854	1741	5
2.30	887	854	1741	5
2.50	860	853	1713	5
3.00	860	853	1713	5
4.00	860	853	1713	5

elapsed time of the whole delivery system with no restrictions on travel and unload times for each vehicle route. The computational results of the experiments on the 45 subproblems show that the algorithm is capable of solving different types of SVRP considering different conditions. Additionally, sensitivity analysis on the final result can be performed by changing the probability levels, upper bounds on travel and unload times, truck capacity, and by using different values of γ and δ for Algorithms (I) and (II) of the "F" type problem.

CHAPTER IX

USING INTERACTIVE COMPUTER PROGRAMS

9.1 Introduction

This chapter describes two interactive computer programs which primarily implement the route construction and route improvement stages of the SVRP. The computer program for the route construction stage of the problem, whether deterministic or probabilistic VRP, provides the user a tool for designing vehicle routes. The computer program for the PREGP and PARGP, whether the final solution for the decision variables is required to be continuous or integer, supply a good procedure for solving any types of LGP.

Both computer programs are interactive in such a way that the computer alerts the user for the necessary inputs. These two programs are coded in FORTRAN. The LIGP program is shown in Appendix A and the SVRP program is shown in Appendix B. For the user, the more important parameters are provided by the program in order to make the procedure faster and save system operating time. For instance, the values of γ and δ are provided by the SVRP program and presented to the user for selection. Additionally, these two interactive computer programs are designed to investigate the user's input data and then prompt the user to correct probable errors or inconsistencies. As a safeguard, the program will move into a new stage only when the input has been checked by the program and verified by the user. These interactive procedures are coded in such a way that any user, with or without previous knowledge about the computer and/or the mathematical structure of these models, can easily operate the system to determine the most desireable solution. Furthermore, the values to be entered are questioned and explained by the program. For instance, when several values must be entered, the program explains how to enter the data and instructs the operator to leave a blank space between the entries.

The remainder of this chapter explains these two interactive procedures in more detail. The next two sections deal with the description of the interactive computer program for the LIGP and for the SVRP, respectively.

9.2 Interactive Linear Integer Goal Programming

This section is concerned with the analysis of the linear integer goal programming procedures, LIPREGP and LIPARGP, as coded in FORTRAN and presented in Appendix A. The interactive procedure for the PREGP is based on the simplex method approach described by Zeleny [69], while the PARGP technique uses the concept of partitioning goal programming developed by Ravindaran [4]. These two techniques can be used to obtain both continuous and integer solutions, and additionally, the interactive integer PREGP technique can be used for the sensitivity analysis of the problem. The two integer programming methods discussed in Chapter VI are incorporated into these two goal programming procedures.

The interactive PREGP can be employed to

1. derive a solution to a general goal programming problem,
2. perform a sensitivity analysis,

3. derive an integer solution (pure or mixed integers) by the cutting plane method, and
4. derive an integer solution (pure or mixed integers) by the branch and bound procedure.

A need for the post optimality analysis comes after the solution of the problem has been obtained. This is because a number of changes may be necessary after the problem is solved. For instance, the goal attainment levels may increase or decrease, or the technological coefficients may need to be changed. This computer program can be used when it is desirable to evaluate a set of new solutions because of one or more of the following changes:

1. change one or more right-hand side values,
2. add one or more new decision variables,
3. add one or more new objective functions, or
4. change the coefficient of a nonbasic variable associated with the i^{th} row, j^{th} nonbasic column.

The PARGP performs exactly the same as the PREGP except that it does not consider the best feature of the PREGP, which is the sensitivity analysis. This is because when the optimal solution to the k^{th} subproblem is obtained, the PARGP procedure requires the deletion of all the nonbasic columns which have a negative value of (Zj - Cj) from the optimal tableau of the k^{th} problem for further consideration (before the addition of new goal constraints of such problem $K+1^{th}$). However, a continuous or integer solution to any problem with linear constraints and objective function terms can be obtained by each of these procedures. The algorithmic flowchart presented in Figure 8 gives the general idea of these computer programs.

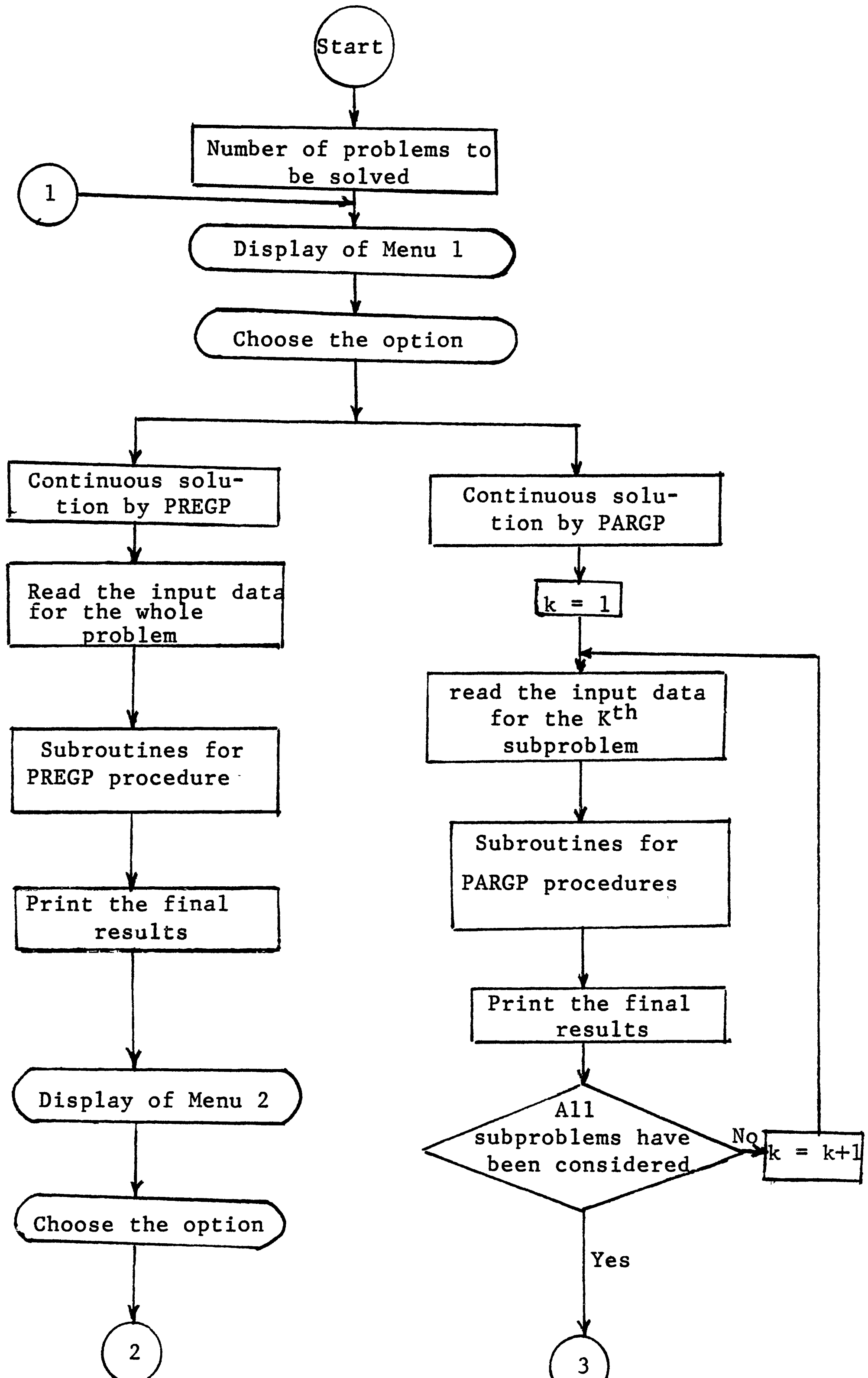

Figure 8. Algorithmic Flowchart of the Computer Program for LIGP (Continued)

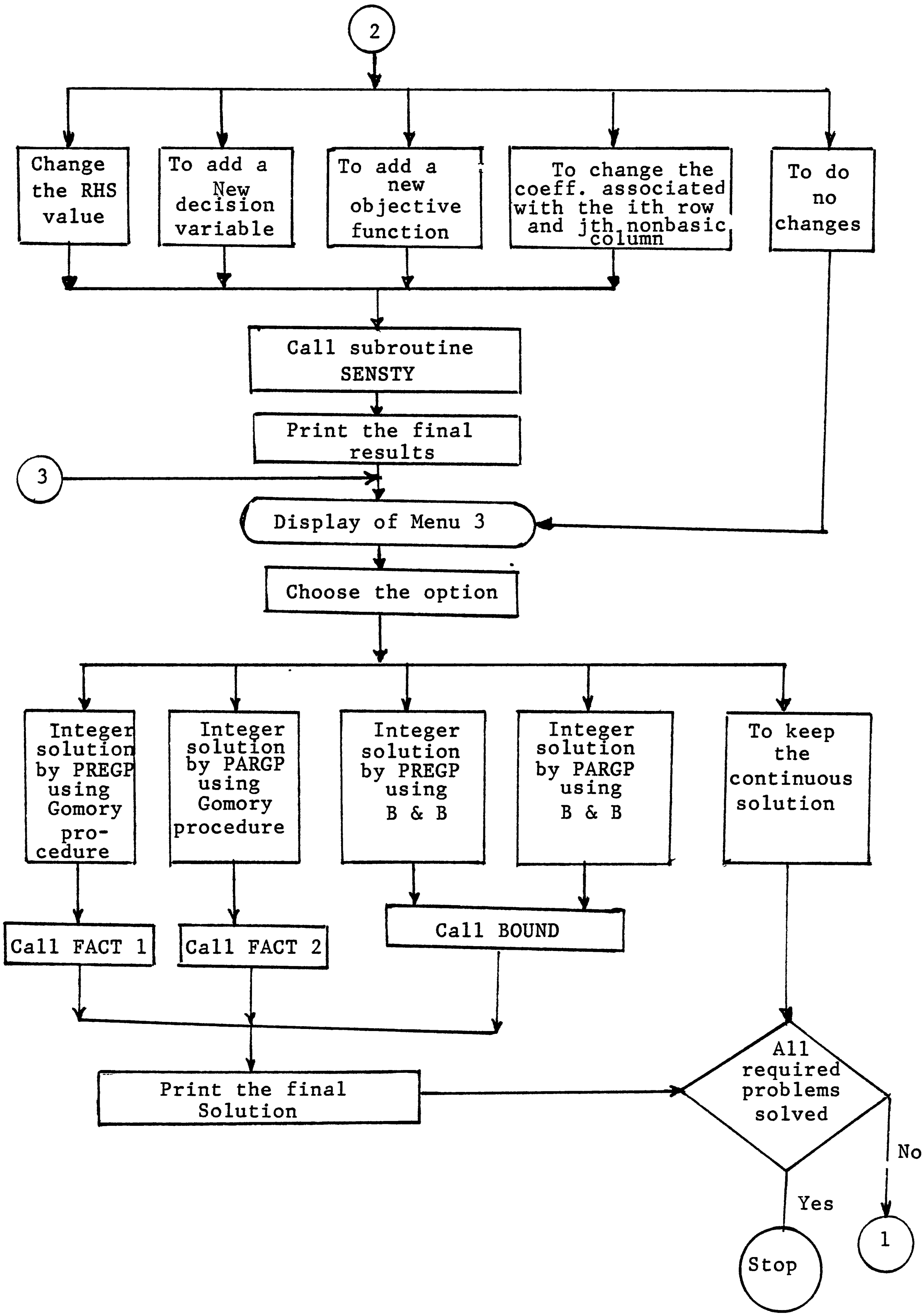

Figure 8. Continued

The first question asked by this program is the number of problems that are to be solved. After the user has input data and pressed the RETURN key, the program asks for verification of the operator's response to the question, then continues by displaying the options from Menu 1 as presented below. An input of "1" (PREGP) or "2" (PARGP) for this menu indicates that the LPREGP or LPARGP is to be used as a selected procedure for solving the desired problem.

DISPLAY OF MENU 1

CONTINUOUS SOLUTION BY PREGP PROCEDURE

*** ENTER 1 ***

CONTINUOUS SOLUTION BY PARGP PROCEDURE

*** ENTER 2 ***

*** CHOOSE THE OPTION ***

The user is allowed to choose a printing procedure for the intermediate computer calculation. The options are to

1. print all calculations in the tableau format, or
2. print only the basic variables and their values including the level of achievement of all priority goals.

After a continuous solution for the original problem is obtained, the program displays a menu of sensitivity analysis, Menu 2. One or more of the changes of the same type which are given in this menu can be performed at the same time. For instance, one can change one or more right hand side values or add one or more new decision variables of the problem.

*** DISPLAY OF MENU 2 ***

*** MENU FOR SENSITIVITY ANALYSIS ***

TO DO NO CHANGES ENTER 5

CHANGE THE RHS VALUES

** ENTER 1 **

TO ADD A NEW DECISION VARIABLE

** ENTER 2 **

TO ADD A NEW OBJECTIVE FUNCTION

** ENTER 3 **

TO CHANGE THE COEFFICIENT ASSOCIATED WITH THE

i^{th} ROW, j^{th} NONBASIC COLUMN

** ENTER 4 ***

*** CHOOSE THE OPTION ***

When no changes are required, the user enters 5 in order to move into the next stage. In this case, the program displays Menu 3, which allows the selection of choices for an integer method, or the option to terminate with a continuous solution only. In order to evaluate the integer solution of the required decision variables, the user can elect to use the final result of the original problem, or continue with the results associated with the last sensitivity analysis.

DISPLAY OF MENU 3

INTEGER SOLUTION BY PREGP USING GOMORY GP

*** ENTER 3 ***

INTEGER SOLUTION BY PREGP USING B & B

*** ENTER 4 ***

INTEGER SOLUTION BY PARGP USING GOMORY GP

*** ENTER 5 ***

INTEGER SOLUTION BY PARGP USING B & B

*** ENTER 6 ***

TO KEEP THE CONTINUOUS SOLUTION

*** ENTER 7 ***

*** CHOOSE THE OPTION ***

The method of data arrangement for this program is described in Section 9.2.1.

9.2.1 The Data Input Procedure

For the sake of time and quick data input, the operator is advised to arrange the data before the logon process begins. To use the PREGP technique, the following data arrangement is necessary:

1. number of constraints, number of variables, and total number of priority levels,
2. number of original decision variables, number of positive and negative deviational variables,
3. number of nonzero elements in the left hand side of the constraints,
4. right hand sides values,
5. basis which is the list of the negative deviation variables, and
6. number of nonzero elements in all priority levels.

Care should be taken in the arrangement of the input data for the PARGP procedure. The following steps should be followed before the arrangement of data for PARGP starts:

1. break down the original problem into as many subproblems as the total number of priorities,
2. the data arrangement for the first subproblem is exactly the same as the data arrangement for PREGP,
3. the number of constraints and variables for the K^{th} subproblem should be calculated as below:

Number of constraints for the K^{th} subproblem = (number of goal and rigid constraints used in all K subproblems) – (number of goal and rigid constraints used in all (k-1) subproblems), and,
Number of variables for the K^{th} subproblem = (number of variables (decisions and deviations) used in all k subproblems) – (number of variables (decisions and deviations) used in all (K-1) subproblems), and

4. enter "0" if no new constraint or no new variable has been used in the new subproblem.

9.3 Interactive Stochastic Vehicle Routing Problem

The main objective of this section is to describe the interactive computer program for the SVRP. Because deterministic VRP is a special case of the SVRP, the program is designed to solve any of these types of problems. A SVRP can be categorized as a VRP

1. with only probabilistic customer demand,
2. with probabilistic travel time, unload time, and customer demand when the total cost of whole system is expected to be minimized ("E" type problem), and

3. with probabilistic customer demand, travel and unload times when total elapsed time of whole delivery system is expected to be minimized ("F" type problem).

The algorithmic flowchart (Figure 9) gives the general idea about this interactive computer program.

The number of problems needed to be solved by the system is the first question the user responds to. Following the verification of this response by the operator, the program displays Menu 1 and expects a response of "1", "2", "3" or "4" as presented below:

DISPLAY OF MENU 1

SELECT ONE OF THE FOLLOWING

TO SOLVE THE DETERMINISTIC VRP

** ENTER 1 **

TO SOLVE A SVRP WITH PROBABILISTIC DEMAND

** ENTER 2 **

TO SOLVE SVRP OF "E" TYPE PROBLEM

** ENTER 3 **

TO SOLVE SVRP OF "F" TYPE PROBLEM

** ENTER 4 **

9.3.1 The Data Input Procedure

VRP: To prevent errors at the time of entering the data, the user needs to arrange the data before the logon process begins. The arrangement of data for the VRP is as shown below:

1. the distance type to be used (euclidean or linear),
2. number of demand points + depot,
3. truck capacity,

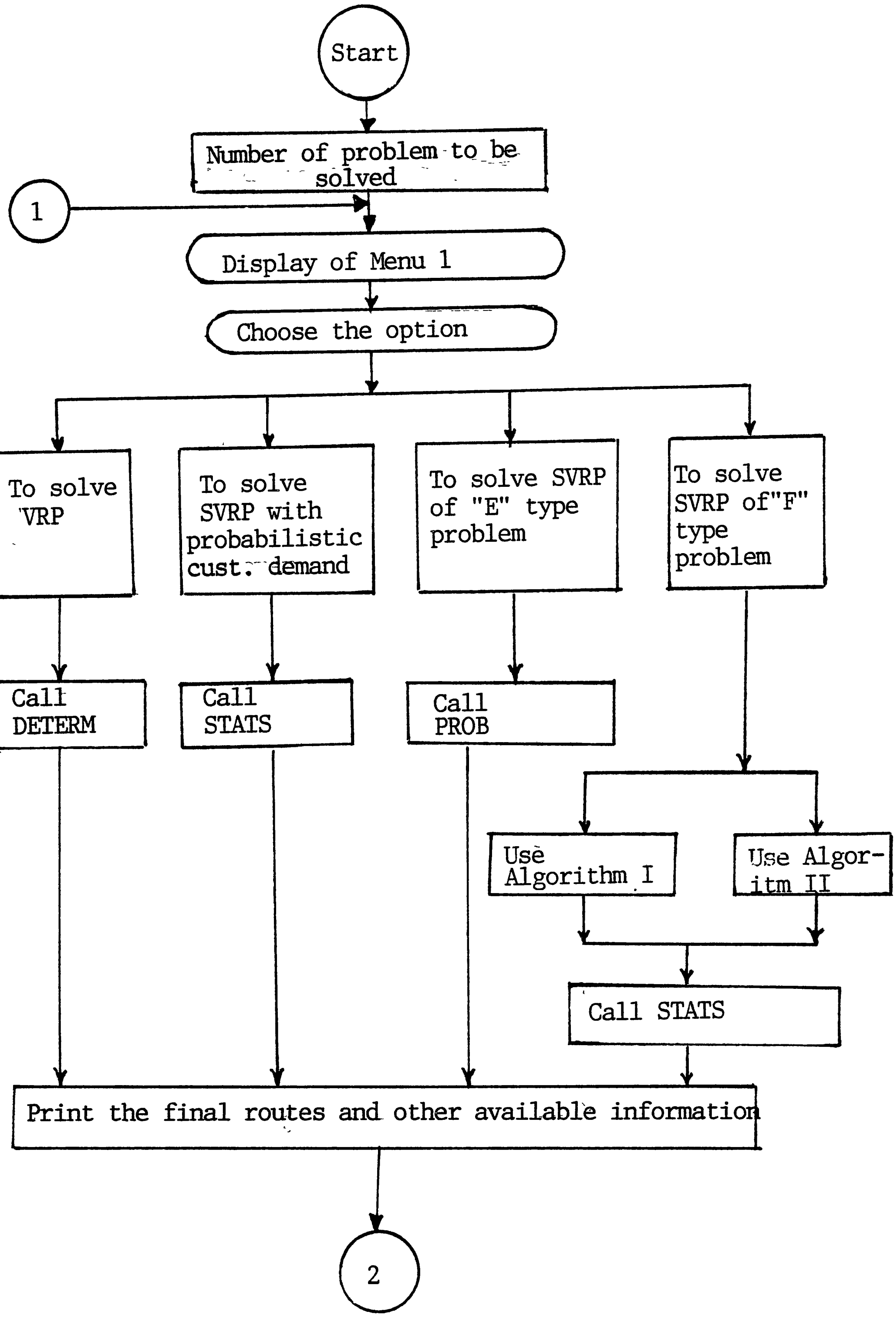

Figure 9. Algorithmic Flowchart of Computer Program for SVRP (Continued)

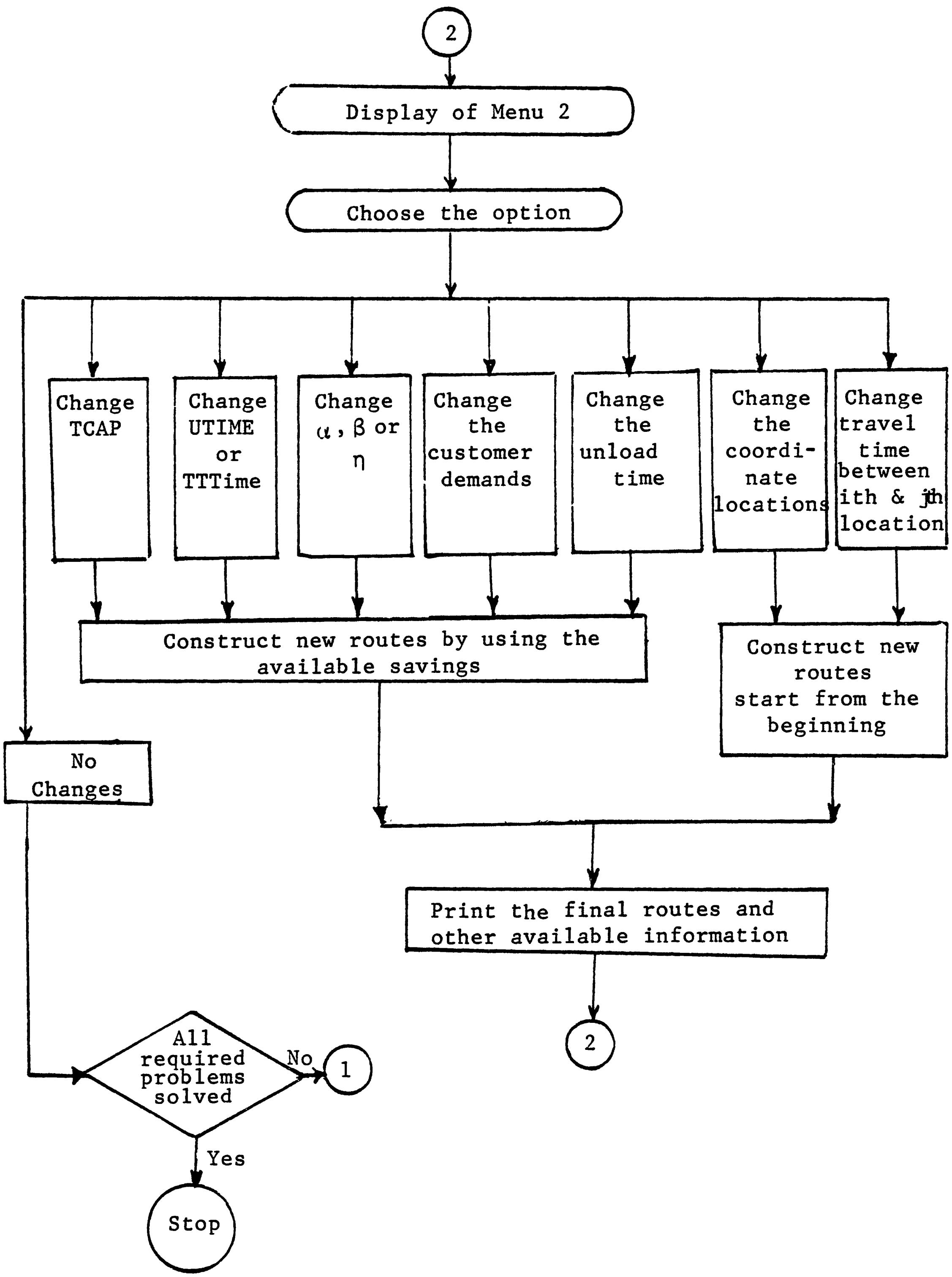

Figure 9. Continued

4. coordinate of depot first and then customer's locations, and
5. customer demands.

SVRP: Steps 1 through 4 of the data arrangement for the VRP can be used when one deals with the SVRP having only probabilistic customer demands. The remainder of steps for SVRP are:

5. the Z value (from the normal table) of the route failure probability when customer demand is probabilistic, and
6. mean and variance of customer demands.

The arrangement of data for the SVRP of the "E" type problem is shown below:

1. the type of distribution function for customer demand,
2. number of customer demands + depot,
3. capacity of truck,
4. total expected unload and travel times for each vehicle route,
5. the Z values for α, β, and η probability levels,
6. mean travel time,
7. variance of travel time,
8. mean and variance of unload time,
9. mean and variance of customer demand, and
10. type of distance (euclidean or linear).

The arrangement of data for the "F" type problem is very similar to the "E" type problem. However, in this case items 1, 4, and 10 are excluded from the data arrangement.

After all necessary data have been entered into the system, the computer will perform all necessary calculations and print the following outputs:

1. total cost, distance, or time depending on the nature of criteria,
2. set of all constructed vehicle routes,
3. total demand of each vehicle route,
4. total travel and unload times for each vehicle route (for "E" and "F" type problems), and
5. number of required vehicles.

In many cases the user will find it necessary to evaluate a set of new solutions by incorporating one or more of the following modifications:

1. change the truck capacity,
2. change the total unload and travel times,
3. change the value of α, β, and η,
4. change one or more customer demands,
5. change the unload time at one or more of stations,
6. change the coordinate of locations, and/or
7. change the travel time between the i^{th} and j^{th} customer.

One or more of these changes can be made when the algorithm displays Menu 2 as shown below:

DISPLAY OF MENU 2

SELECT ONE OF THE FOLLOWING

TO CHANGE THE CAPACITY OF TRUCK

** ENTER 1 **

TO CHANGE THE "UTIME" OR "TTTIME"

** ENTER 2 **

TO CHANGE "ALPHA", "BETA" AND "ETA"

** ENTER 3 **

TO CHANGE THE COORDINATE OF LOCATIONS

** ENTER 4 **

TO CHANGE THE CUSTOMER DEMAND

** ENTER 5 **

TO CHANGE THE UNLOAD TIME

** ENTER 6 **

TO CHANGE THE TRAVEL TIME

** ENTER 7 **

** TO DO NO CHANGES ENTER 8 **

When no changes are expected, the user should enter "8". In this case, the user is allowed to do one of the following:

1. enter "1" to change an "E" type problem to a "F" type problem,
2. enter "2" to change a "F" type problem to an "E" type problem, or
3. enter "3" to terminate.

Changes 1 through 5 of Menu 2 do not require recalculation and ranking of the savings which are associated with each of the two demand points. When the appropriate information for any specific changes has been entered, the program restarts the process of route construction, considering all new and old restrictions of the problem. Although these types of changes save operation as well as computer time, the 6th and 7th changes from list of Menu 2 save only operation time. For these two cases, the program restarts all necessary calculations for savings evaluations and ranks these savings from the largest to the smallest, as described in Chapter VII.

This program is well structured for considering more than one change at a time. For instance, it is possible to change truck capacity, values α, β and η, customer demands, and/or coordinates of locations before moving toward the calculation process. However, the program's structure demands that when making such changes, the coordinates of locations and travel time be considered last.

9.4 Summary

In this chapter, the important features of the interactive computer programs for SVRP and linear integer goal programming are described. Also, methods in which the decision maker can use the interactive LIPREGP for sensitivity analysis and determination of integer solutions using one of the LIGP techniques are demonstrated. A LIPARGP technique which can be used for deriving a continuous or an integer solution is also given. The methods of data arrangements for these two procedures are fully described.

The interactive SVRP is described in detail and it is shown that three different categories of these types of problems can be solved by this procedure. Without termination from the program, the user can change the type of problem which was being used previously. Also, the interactive SVRP can be used when the operator desires to evaluate a set of new solutions by changing the truck capacity, total travel and unload times, and/or probability levels.

CHAPTER X

CONCLUSIONS AND RECOMMENDATIONS

10.1 Conclusions

This dissertation has presented a study of a GP model of the general SVRP in which travel time, unload time, and customer demands are random variables. This research extends the state of the art in multiple objectives SVRPs by fulfilling the primary and secondary objectives and all the subobjectives of Chapter I. That is:

1. A GP model of the problem within the framework of the SVRP has been mathematically formulated,

2. A SVRP in which travel time, unload time, and customer demands may be represented as random variables has been developed,

3. An equivalent deterministic form of the SVRP for RCS and RIS of the problem has been formulated,

4. The existence of a new set of deterministic linear time constraints which are equivalent to the nonlinear set of time constraints of the problem for distributions such as poison, binomial, negative binomial, gamma, chi-square and exponential has been proven through Theory 5.2,

5. A linear GP formulation of the RIS of the problem where conflicting multiple objectives are treated explicitly has been developed,

6. The effects of the route failure probabilities of α_k and β_k on the total elapsed time of the system with $0 \leq \alpha_k \leq 1$ and

$0 \leq \beta_k \leq 1$ for all k have been proven through Theories 5.3, 5.4, and 5.5,

7. The existence of the optimum solution for the route construction stage of the problem has been proven through Theory 5.1,

8. A heuristic algorithm which is a modification of the Clarke and Wright heuristic procedure has been developed in order to solve the "E" type problem,

9. Two new heuristic approaches based on the concepts of the Clarke and Wright algorithm have been originated in order to solve the "F" type problem. The computational experiments show that these two procedures are closely related,

10. A comprehensive, interactive computer program for the SVRP has been developed and described. This program is capable of solving a VRP, a SVRP having only probabilistic customer demands, and a SVRP with "E" and "F" type problems. This program allows the decision maker's involvement in the solution process of the problem. The decision maker can evaluate the solution by changing the value of probability levels, truck capacity, customer demands, and other important parameters of the problem to fully analyze the sensitivity of the final solution, and

11. An interactive Computer Program for the LIGP using the concepts of preemptive and partitioning GP has been developed and described to determine the most favorable vehicle routes of the multiple objective SVRP's where the decision policies and customer requirements need to be fully considered. The interactive procedure allows a decision maker to provide an integer solution for the problem, and to understand the behavior of the system through the utilization of the sensitivity analysis of the optimal solution.

10.2 Recommendations

There are several directions in which additional research should be conducted in the area of the SVRP and the LIGP technique. Some possible considerations for future research are presented below:

1. To develop an interactive computer program as a link between the interactive SVRP and LIGP programs in order to eliminate the operator's time for mathematical formulation of the GP problem which is based on the constructed vehicle routes from the RCS of the problem.

2. To develop new heuristic approaches for solving the "E" and "F" type problems.

3. To develop an iterative procedure that solves the "F" type problem optimally.

4. To apply a computer graphic system to the interactive procedure developed for the SVRP to help the decision maker visualize the constructed vehicle routes.

5. To consider other stochastic elements such as vehicle breakdowns together with the SVRP which is developed in this research.

6. To develop a heuristic approach for solving a GP problem where decision variables are required to be 0-1.

7. To develop an interactive nonlinear integer goal program that can solve the nonlinear constraints of the type generated by the CCP with 0-1 decision variables.

BIBLIOGRAPHY

1. Abouelmagd, A. A. "Aggregate Production and Manpower Planning Models," Ph.D. dissertation, unpublished, Oklahoma State University, May, 1985.

2. Allison, J. D. "Workload Balancing in Vehicle Routing Problems." Ph.D. dissertation, unpublished, Oklahoma State University, 1986.

3. Aneja, Y. P. and K. P. K. Nair. "Bicriteria Transportation Problem." Management Science, Vol. 25 (1979), pp. 73-78.

4. Arthur, J. L. and Ravindran, A. "An Efficient Goal Programming Algorithm Using Constraint Partitioning and Variable Elimination," Management Science, Vol. 24, No. 8, April, 1978.

5. Balinski, M. L. and R. E. Quandt. "On an Integer Operations Program for a Delivery Program." Operations Research, Vol. 12 (1964), pp. 300-304.

6. Ball, M. O., B. L. Golden, A. A. Asad, and L. O. Bodin. "Planning for Truck Fleet Size in the Presence of Common-Carrier Option." Decision Science, Vol. 14(1) (1983), pp. 103-120.

7. Bodin, L., B. Golden, A. Asad, and M. Ball. "Vehicle Routing and Scheduling." Computers and Operations Research, Vol. 10 (1983), pp. 67-211.

8. Buxery, G. "The Vehicle Scheduling Problem and Monte Carlo Simulation." Journal of Operations Research Society, Vol. 30 (1979), pp. 563-573.

9. Carbone, R. "Public Facilities Location Under Stochastic Demand." Journal of Information, Vol. 12 (1974), pp. 261-270.

10. Charnes, A. and W. Cooper. "Deterministic Equivalents for Optimizing and Satisfying Under Chance-Constraints." Operations Research, Vol. 11 (1963), pp. 18-39.

11. Cheshire, I. M., A. M. Malleson, and P. F. Nacache. "A Dual Heuristic for Vehicle Scheduling." Journal of Operational Research Society, Vol. 33 (1982), pp. 51-61.

12. Christofides, N., A. Mingozzi, and P. Toth. "State-Space Relaxation Procedure for the Computation of Bounds to Routing Problems." Networks, Vol. 11 (1981), pp. 145-164.

13. Christofides, N. and S. Eilon. "The Vehicle Routing Problem." Combinatorial Optimization. New York: Wiley, 1979, Chapters VI and XI.

14. Christofides, N., Eilon, S., and C. D. T. Watson-Gandy. Distribution Management, Hafner Publication Company, New York, 1971.

15. Christofides, N. and S. Eilon. "Algorithms for Large Scale TSP." Operational Research Quarterly, Vol. 23 (1972), pp. 511-518.

16. Christofides, N., A. Mingozzi, and P. Toth. "Exact Algorithms for the Vehicle Routing Problem, Based on Spanning Tree and Shortest Path Relaxation." Mathematical Programming, Vol. 20 (1981), pp. 255-282.

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

18. Cook, M. Thomas and R. A. Russell. "A Simulation and Statistical Analysis of Stochastic Vehicle Routing with Timing Constraints." Decision Science, Vol. 9 (1978), pp. 673-687.

19. Dantzig, G. P. and J. H. Ramser. "The Truck Dispatching Problem." Management Science, Vol. 6 (1959), pp. 80-91.

20. Eisner, M. J., R. S. Kaplan, and J. V. Soden. "Admissable Decision Rules for the E-Model of Chance-Constrained Programming." Management Science, Vol. 21 (1971), pp. 337-353.

21. Evans, S. R. and J. P. Norback. "An Heuristic Method for Solving Time-Sensitive Routing Problems." Journal of Operational Research Society, Vol. 35 (1984), pp. 407-414.

22. Federgruen, A. and P. Zipkin. "A Combined Vehicle Routing and Inventory Allocation Problem." Operations Research, Vol. 32 (1984), pp. 1019-1037.

23. Fisher, M. L. and R. Jaikumer. "A Generalized Assignment Heuristic for Vehicle Routing." Networks, Vol. 11 (1981), pp. 109-124.

24. Foster, B. A. and D. M. Ryan. "An Integer Programming Approach to the Vehicle Scheduling Problem." Operational Research Quarterly, Vol. 27 (1976), pp. 3367-3384.

25. Gaskell, T. J. "Bases for Vehicle Fleet Scheduling." Operational Research Quarterly, Vol. 27 (1976), pp. 367-384.

26. Gillett, B. E. and L. R. Miller. "A Heuristic Algorithm for the Vehicle Dispatching Problem." Operations Research, Vol. 22 (1974), pp. 340-349.

27. Gillett, B. and J. Johnson. "Multiterminal Vehicle Dispatch Algorithm." Omega, Vol. 4 (1976), pp. 711-718.

28. Goicoechea A., D. R. Hansen, and L. Ducksteln. Multiobjective Decision Analysis with Engineering and Business Applications. New York: John Wiley & Sons, 1982.

29. Golden, B. L. and J. R. Yee. "A Framework for Probabilistic Vehicle Routing." AIIE, Vol. 11 (1979), pp. 109-112.

30. Golden, B. L. and R. A. Stewart. "Vehicle Routing with Probabilistic Demands." Proceedings of the Tenth Annual Symp. on the Interface of Computer Science and Statistics, Gaithersburg, Maryland, April 1977.

31. Golden, B. L., T. L. Magnantic, and H. Q. Nguyen. "Implementing Vehicle Routing Algorithms." Networks, Vol. 7 (1977), pp. 113-148.

32. Griffith, R. E. and R. A. Stewart. "A Nonlinear Programming Technique for the Optimization of Continuous Processing Systems." Management Science, Vol. 7 (1960), pp. 379-392.

33. Hansotia, B. J. "Stochastic Linear Programming with Recourse: A Tutorial." Decision Science, Vol. 11 (1980), pp. 151-166.

34. Hines W. W. and Montgomery, D. C. Probability and Statistics in Engineering and Management Science. Second edt., John Wiley and Sons, New York, 1980.

35. Holmes, R. A. and R. G. Parker. "A Vehicle Scheduling Procedure Based Upon Savings and a Solution Perturbation Scheme." Operational Research Quarterly, Vol. 27 (1976), pp. 83-92.

36. Hwang, C. L., S. R. Paidy, K. Yoon, and A. S. M. Masud. "Mathematical Programming with Multiple Objectives: A Tutorial." Computers and Operations Research, Vol. 7 (1980), pp. 5-31.

37. Hwang, C. L. and A. S. M. Masud. Multiple Objective Decision Making-Methods and Applications. New York: Springer-Verlag, Berlin Heidelbert, 1979.

38. Ignizio, J. P. Goal Programming and Extensions. Massachusetts: Lexington Books, 1976.

39. Kataoka, S. "A Stochastic Programming Model." Econometrica, Vol. 31 (1963), pp. 181-196.

40. Kolman, B. and Beck, R. E. Elementary Linear Programming with Applications. Academic Press, New York, 1980.

41. Krolak, P., W. Flets, and J. Nelson. "A Man-Machine Approach Towards Solving the Generalized Truck Dispatching Problem." Transportation Science, Vol. 6 (1972), pp. 149-169.

42. Lee, S. M. Goal Programming For Decision Analysis. Auerbach Publishers Inc., Philadelphia, 1972.

43. Lee, S. M. and A. J. Wynne. "Separable Goal Programming." Multiple Criteria Analysis. Eds. Peter Nijkamp and J. Spronk. England: Gower, 1981.

44. Lewis M. R. "Integar Goal Programming: Methods, Computations, Applications," Ph.D. dissertation, unpublished, Virginia Polytechnic Institute and State University, July, 1976.

45. Lin, S. and B. W. Kernighan. "An Effective Heuristic Algorithm for the Traveling Salesman Problem." Operations Research, Vol. 21 (1973), pp. 498-512.

46. Masud, A. S. M. and C. L. Hwang. "Interactive Sequential Goal Programming." Journal of Operational Research Society, Vol. 33 (1981), pp. 391-400.

47. Magnanti,T. L. "Combinatorial Optimization and Vehicle Fleet Planning: Prospectives and Prospects." Networks, Vol. 7 (1981), pp. 179-213.

48. Mole, R. H. and S. R. Jameson. "A Sequential Route-Building Algorithm Employing a Generalized Savings Criteria." Operational Research Quarterly, Vol. 27 (1976), pp. 503-511.

49. Park, Y. B. "The Solution of Vehicle Routing Problems in a Multiple Objective Environment." (Unpub. Ph.D. dissertation, Oklahoma State University, 1984).

50. Russell, R. A. "An Effective Heuristic for the MTOUR Traveling Salesman Problem with some Side Conditions." Operations Research, Vol. 25 (1977), pp. 517-524.

51. Shannon, R. E. Systems Simulation the Art and Science. Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1975.

52. Skitt, R. A. and Levary, R. R. "Vehicle Routing via Column Generation," European Journal of Operational Research, Vol. 21 (1985), pp. 65-75.

53. Stacey, P. J. "Practical Vehicle Routing Using Computer Programs." Journal of Operational Research Society, Vol. 34 (1983), pp. 975-981.

54. Stewart, W. R. J. "New Algorithms for Deterministic and Stochastic Vehicle Routing Problems," Ph.D. dissertation, University of Maryland, 1981.

55. Stewart, W. R. and B. L. Golden. "A Lagrangian Relaxation Heuristic for Vehicle Routing." European Journal of Operational Research, Vol. 15 (1984), pp. 84-88.

56. Taha, H. A. Integer Programming, Theory, Applications, and Computations. New York: Academic Press, 1975.

57. Tillman, F. A. "The Multiple Terminal Delivery Problem with Probabilistic Demands." Transportation Science, Vol. 3 (1969), pp. 192-204.

58. Tillman, F. A. and T. M. Cain. "An Upper Bound Algorithm for the Single and Multiple Terminal Delivery Problem." Management Science, Vol. 18 (1972), pp. 664-682.

59. Tillman, F. A. and H. Cohran. "A Heuristic Approach for Solving the Delivery Problem." Journal of Industrial Engineering, Vol. 19 (July 1968), pp. 354-358.

60. Turner, W. C., J. C. Ghare, and L. R. Fourds. "Transportation Routing Problem--A Survey." AIIE, Vol. 6 (1974), pp. 288-301.

61. Turner, W. C. and V. T. Vu. "Systems Design for Rural Refuse Collection." AIIE, Vol. 11 (1979), pp. 84-85.

62. Vajda, S. Probabilistic Programming. New York: Academic Press, 1970.

63. Waters, C. D. J. "Interactive Vehicle Routing." Operational Research Society, Vol. 35 (1984), pp. 821-826.

64. Williams, B. W. "Vehicle Scheduling: Proximity Priority Searching." Operational Research Society, Vol. 33 (1982), pp. 961-966.

65. Wren, P. C. and A. Holliday. "Computer Scheduling of Vehicles from One or More Depots to a Number of Delivery Points." Operational Research Quarterly, Vol. 23 (1972), pp. 333-334.

66. Yee, J. R. and B. L. Golden. "A Note on Determining Operating Strategies for Probabilistic Vehicle Routing." Naval Research Logistics Quarterly, Vol. 27 (1980), pp. 159-163.

67. Yellow, P. C. "A Computational Modification ot the Savings Method of Vehicle Scheduling." Operational Research Quarterly, Vol. 21 (1970), pp. 281-283.

68. Zare-Mehrjerdi, Yahia. "Chance-Constrained Programming of the Manpower Grading Problem in Production Area." Master thesis, unpublished, St. Mary's University, 1984.

69. Zeleny, M. Multiple Criteria Decision Making. New York: McGraw Hill, 1982.

APPENDIX A

INTERACTIVE COMPUTER PROGRAM FOR LINEAR

INTEGER GOAL PROGRAMMING

```
C*******************************************
C*     FORTRAN PROGRAM TO SOLVE THE LINEAR    *
C*     INTEGER PREEMPTIVE GOAL PROGRAMMING    *
C*     (LIPREGL).AND LINEAR INTEGER PARTITION- *
C*     NING GOAL PROGRAMMING(LIPARGP) PROBLEMS *
C*                                            *
C*******************************************
C*                                            *
C*******************************************
C*                                            *
C*         AUTHOR: YAHYA ZARE-MEHRJERDI       *
C*        ADVISOR:DR.M.P.TERRELL              *
C*       COMPUTER: IBM 3081D                  *
C*           DATE: NOVEMBER 1986              *
C*                                            *
C*         SCHOOL OF INDUSTRIAL               *
C*      ENGINEERING AND MANAGEMENT            *
C*                                            *
C*      OKLAHOMA STATE UNIVERSITY             *
C*        STILLWATER,OK. 74078                *
C*                                            *
C*******************************************
C*
C**********************************************************
C*   THIS PROGRAM ALLOWS  THE USER TO FIND A CONTINOUS OR
C*   INTEGER SOLUTION OF A LINEAR GOAL PROGRAMMING PROBLEM
C*   USING PREEMPTIVE GOAL PROGRAMMING(PREGP) OR PARTITIONNIG
C*   GOAL PROGRAMMING(PARGP)METHODS.
C*   AN INTEGER SOLUTION OF PROBLEM CAN BE OBTAINED USING
C*   EITHER CUTTING PLANE OR BRANCH AND BOUND TECHNIQUES.
C*   ADDITIONNALLY,USER CAN OBTAIN A MIXED INTEGER SOLUTION
C*   OF THE PROBLEM BY EMPLOYING EITHER OF THESE GP METHODS.
C*   FINALLY,PREGP CAN BE USED FOR THE PURPOSE OF SENSITIVITY
C*   ANALYSIS OF THE PROBLEM.
C**********************************************************
C
C*   THE FOLLOWING SUBROUTING ARE USED IN THIS
C*   PROGRAM.
C*
C*      INTERS=  TO READ THE INPUT DATA
C*      PIVCOL=  TO FIND THE PIVOT COLUMN
C*      PIVROW=  TO FIND THE PIVOT ROW
C*     CALC  =   TO UPDATE THE NEW TABLEAU
C*      PARTG =  IS USED FOR PARTITIONNING GP.
C*      ACHECK=  TO DETERMINE THE # OF ALTERNATIVE SOLUTIONS FOR PARGP
C*      INILS =  TO PREPARE THE INITIAL TABLEAU FOR SUBPROBLEMS
```

```
C*              OTHER THAN THE FIRST ONE FOR PARGP
C*       TSORT =  IT SORTS THE BASIC AND NONBASIC VARIABLES WHOSE VALUES
C*                ARE REQUIRED TO BE INTEGER
C
C*       BOUND =  TO CONTROL THE PROGRAM FOR THE BRANCH AND BOUND PROCEDU
C*       BRNCH =  TO DETERMINE THE SOURCE ROW AND DEVELOPE THE NEW
C                 CONSTRAINTS FOR THE BRANCH AND BOUND PROCEDURE
C
C*     SUMMARY=  IT STORE ALL INTEGER SOLUTIONS IN ORDER TO PROVIDE
C*                THE USER WITH THE LIST OF ALL INTEGER VARIABLES
C
C      MEMOH =  CONSISTS THE CONTENTS OF MENU 1,2 AND 3.
C*       ADDUP =  TO ENTER THE DATA FOR NEW PRIORITY LEVEL
C*       DUALS =  STANDS FOR DUAL SIMPLEX METHOD
C*       PIVROW=  TO FIND THE PIVOT ROW
C*       FACT1 =  TO CONTROL THE PROGRAM FOR PREEMPTIVE LIGP TECHNIQUE
C*       FACT2 =  TO CONTROL THE PROGRAM FOR PARTITIONNING LIGP
C*                TECHNIQUE
C*
C*       INTGR =  TO DEVELOP THE GOMQRY CUTTING PLANE FOR PURE AND MIXED
C*                INTEGER VARIABLES FOR BOTH PREGP AND PARGP TECHNIQUES
C*
C*       SADD=    TO PROVIDE THE OBJECTIVE FUNCTION FOR THE FINAL TABLEAU
C*                AND TO DOWNGRADE THE PREVIOUS OBJECTIVES BY ONE LEVEL
C*       GOMORY=  TO FIND THE SOURCE ROW AND COUNT THE # OF AVAILABLE
C*                INTEGER VARIABLES
C*
C*       BINVRS=  TO CALCULATE THE INVERSE OF MATRIX B
C*
C*       SENSTY=  TO PERFORM THE SENSITIVITY ANALYSIS FOR PREGP.
C*
C*****************************************************
C*
C*          DEFINITION OF VARIABLES
C*
C*       IPEMPT =   STANDS FOR PREEMPTIVE GOAL PROGRAMMING PROCEDURE
C*       IPART  =  STANDS FOR PARTITIONNIG GOAL PROGRAMMING PROCEDURE
C*
C*       NPRO   =  TOTAL NUMBER OF PRIORITIES
C*       NOPRO  =  NUMBER OF PRIORITIES.NOTE THAT NOPRO=NPRO
C*                 FOR PREEMPTIVE PROCEDURE.NOPRO=1 FOR SGL.
C*       NOV    =  NUMBER OF VARIABLES
C*       NOC    =  NUMBER OF CONSTRAINTS
C*        IP    =  IS THE NUMBER OF CONSTRAINTS PLUS ONE
C*        IC    =  IS THE NUMBER OF CONSTRAINTS PLUS PRIORITIES
C*        IW    =  IS THE NUMBER OF ALL VARIABLES PLUS ONE
```

```
C*       NDS     =  NUMBER OF ORIGINAL DECISION VARIABLES
C*      LND1     =  NUMBER OF ORIGINAL NEGATIVE DEVIATIONS
C*      LPD1     =  NUMBER OF ORIGINAL POSITIVE DEVIATIONS
C*     NPRNT     =  1 MEANS PRINT ALL INTERMEDIATE TABLEAUES
C*               =  2 MEANS TO PRINT THE LIST OF ALL VARIABLES AND
C*                    PRIORITY LEVELS AND THEIR VALUES
C*    IBOUND     =  O USE LIPREGP TOGETHER WITH THE CUTTING PLANE METHOD
C*               =  1 USE LIPARGP TOGETHER WITH THE CUTTING PLANE METHOD
C*               =  3 USE THE BRANCH AND BOUND TECHNIQUE
C*     INTGP     =  O KEEP THE FINAL SOLUTION CONTINOUS
C*               =  1 FIND THE INTEGER SOLUTION OF THE PROBLEM
C*    NXREAL     =  NUMBER OF INTEGER VARIABLES
C*    INTRANT    =  NUMBER OF ITERATIONS
C*       KING    =  COUNTS THE NUMBER OF VARIABLES FOR THE PARGP PROCEDURE
C*      NNZRO    =  NUMBER OF NONZERO ELEMENST IN THE NEW CONSTRAINTS
C*      IPZRO    =  NUMBER OF NONZERO ELEMENTS IN THE NEW PRIORITY
C*     KINPRO    =  IS THE NUMBER OF PRIORITIES INCLUDING THE ABSOLUTE ONE
C*     IPROBL    =  NUMBER OF PROBLEMS TO BE SOLVED
C*     NONZRO    =  NUMBER OF NONZERO ELEMENTS IN THE MATRIX OF
C*                  TECHNOLOGICAL COEFFICIENTS
C*      FRACT    =  IS THE FRACTION PART OF THE VALUE OF A VARIABLE
C*       KNOC    =  NUMBER OF NEW CONSTRAINTS TO BE ADDED IN PARGP
C*       KNOV    =  NUMBER OF NEW VARIABLES TO BE ADDED IN PARGP
C*       IPVC    =  STANDS FOR PIVOT COLUMN
C*       IPROW   =  INDICATES PIVOT ROW
C*       INO     =  IS A COUNTER
C*       XMAX    =  STANDS FOR THE MAXIMUM
C*      IB(I)    =  ARRAY OF BASIC VARIABLES
C*      ID(I)    =  ARRAY OF ALL VARIABLES(DECISION PLUS DEVIATION)
C*      IN(I)    =  ARRAY OF GOAL ACHIVEMENT LEVELS
C*      IV(I)    =  ARRAY OF VALUE OF RIGHT HAND SIDE VALUES
C*    IBOR(I)    =  ARRAY OF BASIC VARIABLES (USED FOR SENSITIVITY ANAL.)
C*   IVZAR(I)    =  ARRAY OF VALUE OF RHS(USED FOR THE SENSITIVITY ANAL.)
C*    ISDD(I)    =  ARRAY OF ORIGINAL LIST OF VARIABLES(USED FOR
C*                  SENSITIVITY ANAL.)
C*    SSIN(I)    =  ARRAY OF GOAL ACHIVEMENT LEVELS(USED FOR SENSITITY
C*                  ANAL.)
C*   LDECS(I)    =  ARRAY OF DECISION VARIABLES
C*   LPDEV(I)    =  ARRAY OF POSITIVE DEVIATIONS
C*   LNDEV(I)    =  ARRAY OF NEGATIVE DEVIATIONS
C*   IREAL(I)    =  LIST OF REQUIRED INTEGER VARIABLES
C*     IPM(I)    =  O INDICATES THAT VARIABLE I IS NONBASIC
C*               =  OTHERWISE THE NONBASIC VARIABLE HAS ALREADY BEEN
C*                  DELETED FROM THE TABLEAU
C*     DUM(I)    =  ARRAY OF CUTTING PLANE
C*  NONBAS(I)    =  THE LIST OF NONBASIC VARIABLES FROM THE REQUIRED
```

```
C*                   INTEGER VALUED VARIABLES
C*   KBASE(I)    =  THE LIST OF BASIC VARIABLES FROM THE LIST OF THE
C*                   REQUIRED INTEGER VALUED VARIABLES
C*    XDOM(I)    =  ARRAY OF NEW CONSTRAINTS USED IN THE BRANCH AND
C*                   BOUND TECHNIQUE
C*    XMOD(I)    =  IT IS EQUAL TO THE -XDOM(I)
C*    SIV(I)     =  TO SAVE THE VALUE OF THE RHS VALUES
C*    ISIN(I)    =  TO SAVE THE LIST OF THE BASIC VARIABLES
C*     SIN(I)    =  TO SAVE THE VALUE OF THE PRIORITY LEVELS
C*    ISID(I)    =  TO SAVE THE LIST OF THE DECISION VARIABLES
C*     AIV(I)    =  TO SAVE THE VALUE OF PRIORITY LEVELS OF THE OPTIMAL
C*                   TABLEAU FOR SOLVING SUBPROBLEM TWO
C*     AIN(I)    =  TO SAVE THE RHS VALUE OF THE OPTIMAL TABLEAU FOR
C*                   SOLVING SUBPROBLEM TWO
C*   IDVAR(I)    =  TO SAVE THE LIST OF THE VARIABLES OF THE OPTIMAL
C*                   TABLEAU FOR SOLVING SUBPROBLEM TWO
C*  IABASE(I)    =  TO SAVE THE LIST OF THE BASIC VARIABLES OF THE
C*                   OPTIMAL TABLEAU FOR SOLVING SUBPROBLEM TWO
C*     KBB(I)    =  TO SAVE THE LIST OF THEBASIC VARIABLES OF OPTIMAL
C*                   TABLEAU OF SUBPROBLEM 1
C*     RHV(I)    =  TO SAVE THE VALUE OF THE BASIC VARIABLES OF THE
C*                   OPTIMAL TABLEAU OF SUBPROBLEM 1
C*    ARHN(I)    =  TO SAVE THE VALUE OF THE PRIORITY LEVELS OF THE
C*                   OPTIMAL TABLEAU OF SUBPROBLEM 1
C*     KDD(I)    =  TO SAVE THE LIST OF THE DECISION VARIABLES OF THE
C*                   OPTIMAL TABLEAU OF SUBPROBLEM 1
C*       F(I)    =  TO SAVE THE VALUE OF THE PRIORITY LEVEL OF SUB-
C*                   PROBLEM 1
C*      FF(I)    =  TO SAVE THE VALUE OF THE PRIORITY LEVELS OF SUB
C*                   PROBLEM 2
C*   SARRY(I)    =  TO SAVE THE INTEGER SOLUTION OF VARIABLES
C*   XARRY(I)    =  TO SAVE THE LIST OF THE INTEGER VARIABLES
C*     IBU(I)    =  TO SAVE THE LIST OF BASIC VARIABLES OF THE OPTIMAL
C*                   TABLEAU OF SUBPROBLEM 2
C*     IUN(I)    =  TO SAVE THE VALUE OF THE PRIORITY LEVELS OF THE
C*                   OPTIMAL TABLEAU OF SUBPROBLEM 2
C*     IUD(I)    =  TO SAVE THE LIST OF THE DECISION VARIABLES OF THE
C*                   OPTIMAL TABLEAU OF SUBPROBLEM 2
C(*     IUV(I)    =  TO SAVE THE VALUE OF THE BASIC VARIABLES OF THE
C*                   OPTIMAL TABLEAU OF SUBPROBLEM 2
C*   TAB(I,J)    = ARRAY OF TABLEAU OF THE ORIGINAL PROBLEM
C*    ZZ(1,J)    =  ARRAY OF TABLEAU OF THE JTH VARIABLE OF PRIORITY 1
C*     C(I,J)    =  ARRAY OF THE PRIORITY WEIGHTS FOR THE JTH VARIABLE
C*                   AND ITH PRIORITY LEVEL
C*     Z(I,J)    =  ARRAY OF TABLEAU OF PRIORITY VALUES
C*   SZV(I,J)    =  AN ARRAY USED FOR SENSITIVITY ANALYSIS
```

```
C* STOF(I,J)    =  AN ARRAY USED FOR THE SENSITIVITY ANALYSIS
C* ATAB(I,J)    =  ARRAY OF THE OPTIMAL TABLEAU USED FOR THE BRANCH AND
C*                 AND BOUND TECHNIQUE
C* AZE(I,J)     =  THIS ARRAY IS USED IN THE BRANCH AND BOUND TECHNIQUE
C* ATT(I,J)     =  ARRAY OF OPTIMAL TABLEAU OF SUBPROBLEM 1
C* AZZ(I,J)     =  ARRAY OF PRIORITY COEFFICIENTS OF OPTIMAL TABLEAU
C*                 OF SUBPROBLEM 1
C* TUT(I,J)     =  ARRAY OF OPTIMAL SOLUTION OF SUBPROBLEM 2
C* ZUZ(I,J)     =  ARRAY FROM THE OPTIMAL TABLEAU OF SUBPROBLEM 2
C*SENS(I,J)     =  GIVES THE MATRIX OF B INVERS
C* SZZ(I,J)     =  THIS ARRAY IS USED FOR THE SENSITIVITY ANALYSIS
C* STABB(I,J)   =  THIS TABLEAU IS USED IN THE SENSITIVITY ANALYSIS
C
C*****************************************************
C*          MAIN PROGRAM                             *
C*****************************************************
C*
          DIMENSION TAB(100,100),Z(100,100),ID(100),IB(100)
          DIMENSION IBOR(100),IVZAR(100)
          DIMENSION STOF(100,100),ISDD(100),SZV(100,100),SSIN(100)
          DIMENSION TABB(100,100),ZZ(1,100),IPM(1000),IREAL(50)
          DIMENSION IV(100),IN(100),LEVATT(50),C(100,100)
          DIMENSION LDECS(100),LPDEV(100),LNDEV(100)
          REAL IV,IN
          COMMON/B1/TAB,IV,ID,IB,LIT
          COMMON/B2/Z,IN,IP,IC,IW
          COMMON/B3/IPVC,XMAX,NOV,INO
          COMMON/B4/NPRNT,NOC,NOPRO
          COMMON/B5/NPRO,IPM,TABB,ZZ
          COMMON/B6/IPEMPT,IPART,IBOUND
          COMMON/B7/ICHECK,INTGP,IREAL,NXREAL
          COMMON/B11/KAT,KIT
          COMMON/B12/ITRATN,KINPRO,LEVATT
          COMMON/B13/ICOUNT,IFLAG
          COMMON/B15/IPAT,KRAZY
          COMMON/B19/IBUND
          COMMON/B20/LDECS,LPDEV,LNDEV,LTOT1,LTOT2,LTOT3
          COMMON/SS1/IBOR,IVZAR,STOF,ISDD,SZV,SSIN
C
C  NUMBER OF PROBLEMS YOU WISH TO SOLVE
C
          WRITE(6,5)
          WRITE(10,5)
5         FORMAT(//2X,'ENTER THE NUMBER OF PROBLEMS YOU WISH TO SOLVE')
7         READ (5,*) IPROBL
C
```

```
          WRITE(6,8) IPROBL
          WRITE(10,8) IPROBL
8         FORMAT(//2X,'NUMBER OF PROBLEMS=',2X,I2)
          IF(IPROBL.LE.O) THEN
          WRITE(6,6)
          WRITE(10,6)
6         FORMAT(//5X,'REENTER AGAIN')
          GO TO 7
          ENDIF
          DO 3001 I=1,IPROBL
C***
C**  TO COUNT THE NUMBER OF ITERATIONS
          ITRATN=O
          IBUND=O
          ICOUNT=O
C*  TO KEEP THE NUMBER OF VARIABLES.
          KAT=O
C* TO DISPLAY MENU 1
C*
          JOYL=1
          CALL MEMOH(JOYL,ISEN)
          IF (IPART.EQ.1) CALL PARTG
          IF(IPEMPT.EQ.1) GO TO 1000
          GO TO 3003
C***** CALL FOR INPUT DATA
1000      ICHECK=O
          WRITE (6,151)
          WRITE(10,151)
151       FORMAT(//10X,'ALGORITHM IS USING GP PREEMTIVE PROCEDURE')
          INTUR=1
          CALL INTERS (INTUR)
C***      CALL FOR PIVOT COLUMN AND PIVOT ROW
C***
          INO=1
           DO 20 INO=1,NOPRO
 10       CALL PIVCOL
          IF(XMAX.EQ.O) GO TO 20
          IF(IPVC.EQ.O) GO TO 20
          CALL PIVROW(IPROW)
          CALL CALC(IPROW)
          CALL PTRG(NOV)
C**
          IF(INO.EQ.1) GO TO 10
          IF(IN(INO).EQ.O) GO TO 20
          GO TO 10
 20         CONTINUE
```

```
          ISSS=0
C TO SPECIFY THE TYPE OF SENSITIVITY ANALYSIS
502       WRITE(6,500)
          WRITE(10,500)
500       FORMAT(5X,'DO YOU WISH TO DO ANY SENSITIVITY ANALYSIS')
          WRITE(6,501)
          WRITE(10,501)
501       FORMAT(5X,'** ENTER 1 FOR YES **'//5X,'** ENTER 2 FOR NO *')
          READ(5,*) NOYES
          IF(NOYES.EQ.2) GO TO 3003
C*
C* TO DISPLAY MENU 3
C*
          JOYL=2
          CALL MEMOH(JOYL,ISEN)
          IF(ISEN.EQ.5) GO TO 3003
          ISSS=ISSS+1
          IF(ISSS.EQ.1) THEN
          IAM=1
          ELSE
          WRITE(6,503)
          WRITE(10,503)
503       FORMAT(/5X,'DO YOU LIKE TO WORK WITH THE FINAL TABLEAU'/
     +5X,'OF THE ORIGINAL PROBLEM')
          WRITE(6,504)
          WRITE(10,504)
504       FORMAT(/5X,'ENTER',2X,'1::YES',2X,'2:NO')
          READ(5,*) NOYYSS
          IF(NOYYSS.EQ.1) THEN
          IAM=2
          CALL SENSTY(ISEN,IAM)
          IAM=3
          ELSE
          IAM=3
          ENDIF
          ENDIF
          CALL SENSTY(ISEN,IAM)
          ISUMMY=3
          CALL SUMMRY(IB,IN,NOPRO,IV,JZJJ,NOC,ISUMMY)
          GO TO 502
C TO DETERMINE AN INTEGER PROCEDURE,CUTTING PLANE METHOD OR BRANCH
C AND BOUND TECHNIQUE
3003     JOYL=3
         CALL MEMOH(JOYL,ISEN)
         IF(IPART.EQ.1) GO TO 3000
         IF(INTGP.EQ.1) THEN
```

```
          IAM=2
          CALL SENSTY(ISEN,IAM)
          IF(IBOUND.EQ.1.AND.IPEMPT.EQ.1) THEN
          CALL FACT1
          GO TO 3001
          ENDIF
C* TO DISPLAY MENU 2
3000      IF(IBOUND.EQ.1.AND.IPART.EQ.1) THEN
          CALL FACT2
          GO TO 3001
          ENDIF
          IF(IBOUND.EQ.2) THEN
          CALL PTRG(NOV)
          IBUND=1
          CALL BOUND
          GO TO 3001
          ENDIF
          ENDIF
 3001     CONTINUE
          STOP
          END
C*************************************************
C*              SUBROUTINE INTERS               *
C*************************************************
          SUBROUTINE INTERS(INTUR)
          DIMENSION IBOR(100),IVZAR(100)
          DIMENSION STOF(100,100),SZV(100,100),ISDD(100),SSIN(100)
          DIMENSION TAB(100,100),Z(100,100),ID(100),IV(100)
          DIMENSION IB(100),IREAL(50),IN(100),C(100,100)
          DIMENSION TABB(100,100),LEVATT(50),ZZ(1,100),IPM(1000)
          DIMENSION LDECS(100),LPDEV(100),LNDEV(100)
          REAL IV,IN,ISM,LEVATT
          COMMON/B1/TAB,IV,ID,IB,LIT
          COMMON/B2/Z,IN,IP,IC,IW
          COMMON/B3/IPVC,XMAX,NOV,INO
          COMMON/B4/NPRNT,NOC,NOPRO
          COMMON/B5/NPRO,IPM,TABB,ZZ
          COMMON/B6/IPEMPT,IPART,IBOUND
          COMMON/B7/ICHECK,INTGP,IREAL,NXREAL
          COMMON/B8/C
          COMMON/B12/ITRATN,KINPRO,LEVATT
          COMMON/B20/LDECS,LPDEV,LNDEV,LTOT1,LTOT2,LTOT3
          COMMON/SS1/IBOR,IVZAR,STOF,ISDD,SZV,SSIN
          GO TO (2,4),INTUR
2         WRITE(6,9)
          WRITE(10,9)
```

```
 9      FORMAT(10X,'-->','ENTER 1 FOR PRINTING ALL TABLEAU'/
     +13X,'ENTER 2 FOR PRINTING VARIABLES AND THEIR VALUES'/)
 100    CONTINUE
        READ(5,*,END=100) NPRNT
 101    CONTINUE
        WRITE(6,10)
        WRITE(10,10)
 10     FORMAT(5X,'-->',2X,'ENTER NO. OF CONTS.'//5X,'-->',2X,
     +  'ENTER NO.OF VARIABLES'//5X,'-->',2X,'ENTER NO. OF PRIORITIES')
        READ(5,*,END=101) NOC,NOV,NPRO
        IF(IPEMPT.EQ.1) NOPRO=NPRO
        IF(IPART.EQ.1)  NOPRO=1
        WRITE(6,20) NOC,NOV,NOPRO
        WRITE(10,20) NOC,NOV,NOPRO
 20     FORMAT(5X,'NOC=',I2//5X,'NOV=',I2//5X,'NOPRO=',I2)
        WRITE(6,5)
        WRITE(10,5)
        READ(5,*) ICORR
        IF(ICORR.EQ.2) THEN
        WRITE(6,7)
        WRITE(10,7)
        GO TO 101
        ENDIF
        WRITE(6,21)
        WRITE(10,21)
 21     FORMAT(5X,'-->',2X,'ENTER THE NUMBER OF ALL VARIABLES'
     +/10X,'ENTER THE # OF POSITIVE DEVIATIONS'/10X,'ENTER THE # OF
     +ALL NEGATIVE DEVIATIONS')
        READ(5,*) NDS,LPD1,LND1
        WRITE(6,600)
        WRITE(10,600)
        READ(5,*) (LDECS(I),I=1,NDS)
        IF(LPD1.EQ.0) GO TO 151
        WRITE(6,601)
        WRITE(10,601)
        READ(5,*) (LPDEV(J),J=1,LPD1)
151     IF(LND1.EQ.0) GO TO 152
        WRITE(6,602)
        WRITE(10,602)
600     FORMAT(5X,'-->',2X,'ENTER THE LIST OF DECISION VARIABLES')
601     FORMAT(5X,'-->',2X,'ENTER THE LIST OF POS- DEV VARIABLES')
602     FORMAT(5X,'-->',2X,'ENTER THE LIST OF NEG-DEV VARIABLES')
        READ(5,*) (LNDEV(K),K=1,LND1)
152     LTOT1=NDS
        LTOT2=LPD1
        LTOT3=LND1
```

```
C
      DO 22 I=1,NOV
      ID(I)=I
      ISDD(I)=I
22    CONTINUE
4     IF(INTGP.EQ.1) THEN
30    WRITE(6,170)
      WRITE(10,170)
170   FORMAT(//10X,'--->',2X,'ENTER THE NUMBER OF INTEGER VARIABLS')
      READ(5,*) NXREAL
      WRITE(6,171) NXREAL
      WRITE(10,171) NXREAL
171   FORMAT(//10X,'--->',2X,'ENTER',2X,I3,2X,'VARIABLES NAMES')
      READ(5,*) (IREAL(I),I=1,NXREAL)
      WRITE(6,605)
      WRITE(10,605)
605   FORMAT(//20X,'LIST OF THE REQUIRED INTEGER VARIALES')
      DO 604 I=1,NXREAL
      WRITE(6,603) IREAL(I)
      WRITE(10,603) IREAL(I)
603   FORMAT(//20X,I5)
604   CONTINUE
      WRITE(6,5)
      WRITE(10,5)
      READ(5,*) ICORR
      IF(ICORR.EQ.2) THEN
      WRITE(6,7)
      WRITE(10,7)
      GO TO 30
      ENDIF
      RETURN
      ENDIF
      DO 41 I=1,NOC
      DO 41 J=1,NOV
      STOF(I,J)=0.0
 41   TAB(I,J)=0.0
      WRITE(6,42)
      WRITE(10,42)
42    FORMAT(5X,'-->',2X,'ENTER NUMBER OF NONZERO ELEMENTS IN'
     +/10X,'THE TECHNOLOGICAL MATRIX')
      READ(5,*) NONZRO
      WRITE(6,23)
      WRITE(10,23)
 23   FORMAT(5X,'-->',2X,'ENTER THE TECHNOLOGICAL COEFFICIENTS')
      WRITE(6,31)
      WRITE(10,31)
```

```
31         FORMAT(5X,'-->',2X,'ENTER ROW I ,COLUMN J AND THEN ITS'/
     +10X,'VALUE.',2X,'LEAVE ONE SPACE BETWEEN ENTRIES')
 102       CONTINUE
           DO 43 I=1,NONZRO
33         READ(5,*,END=102) L,M,VALUE
           WRITE(6,32) L,M,VALUE
           WRITE(10,32) L,M,VALUE
32         FORMAT(10X,'ROW=',2X,I3,2X,'COLUMN=',2X,I3,2X,'VALUE=',F8.4)
           WRITE(6,5)
           WRITE(10,5)
           READ(5,*) ICORR
           IF(ICORR.EQ.2) THEN
           WRITE(6,7)
           WRITE(10,7)
           GO TO 33
           ENDIF
           TAB(L,M)=VALUE
           STOF(L,M)=VALUE
 43        CONTINUE
           DO 40 I=1,NOPRO
           DO 40 J=1,NOV
           SZV(I,J)=0.0
 40        Z(I,J)=0
           WRITE(6,50)
           WRITE(10,50)
 50        FORMAT(5X,'-->',2X,'ENTER THE RIGHT HAND SIDE VALUES')
 103       CONTINUE
           DO 60 I=1,NOC
35         READ(5,*,END=103) IV(I)
           WRITE(6,34) IV(I)
           WRITE(10,34) IV(I)
34         FORMAT(5X,'RHS=',2X,F9.4)
           WRITE(6,5)
           WRITE(10,5)
           READ(5,*) ICORR
           IF(ICORR.EQ.2) THEN
           WRITE(6,7)
           WRITE(10,7)
           GO TO 35
           ENDIF
 60        IVZAR(I)=IV(I)
           WRITE(6,70)
           WRITE(10,70)
  70       FORMAT(5X,'-->','ENTER THE INITIAL BASIC VARIABLES')
 104       CONTINUE
           DO 80 I=1,NOC
```

```
37        READ(5,*,END=104) IB(I)
          WRITE(6,36) IB(I)
          WRITE(10,36) IB(I)
36        FORMAT(5X,'BAISC VARIABLE=',2X,I4)
          WRITE(6,5)
          WRITE(10,5)
          READ(5,*) ICORR
          IF(ICORR.EQ.2) THEN
          WRITE(6,7)
          WRITE(10,7)
          GO TO 37
          ENDIF
80        IBOR(I)=IB(I)
 106      CONTINUE
          DO 1201 I=1,NOPRO
          DO 1201 J=1,NOV
1201      C(I,J)=0.
          WRITE(6,1202)
          WRITE(10,1202)
1202      FORMAT(5X,'-->',2X,'ENTER THE NUMBER OF NONZERO ELEMENTS'
     +/10X,'IN THE PRIORITY WEIGHT MATRIX')
          READ(5,*,END=106) NZROP
          WRITE(6,38)
          WRITE(10,38)
38        FORMAT(5X,'-->',2X,'ENTER PRIORITY NUMBER I ,VARIABLE #'/10X,
     +'THEN ITS PRIORITY WEIGHT.',2X,'LEAVE A SPACE BETWEEN ENTRIES')
          DO 1203  I=1,NZROP
45        READ(5,*) J,KK,PL
          WRITE(6,39) J,KK,PL
          WRITE(10,39) J,KK,PL
39        FORMAT(10X,'PRIORITY # =',2X,I3,2X,'VARIABLE # =',2X,I3,2X,
     +'PRIORITY WEIGHT =',2X,F8.4)
          WRITE(6,5)
          WRITE(10,5)
          READ(5,*) ICORR
          IF(ICORR.EQ.2) THEN
          WRITE(6,7)
          WRITE(10,7)
          GO TO 45
          ENDIF
1203      C(J,KK)=PL
          IF(ICHECK.EQ.1) RETURN
          IROUTY=1
          IF(IROUTY.EQ.1) GO TO 186
          DO 1200 J=1,NOPRO
          DO 1200 I=1,NOC
```

```
      K=IB(I)
      Z(J,K)=0
 1200 CONTINUE
      JJ=IB(1)
      JJ1=JJ-1
      DO 150 I=1,NOPRO
      DO 150 J=1,JJ1
      SUM=0
      ISM=0
      DO 160 K=JJ,NOV
      KK=K-JJ1
      ISM=ISM+C(I,K)*IV(KK)
 160  SUM=SUM+C(I,K)*TAB(KK,J)
      Z(I,J)=SUM-C(I,J)
      SZV(I,J)=Z(I,J)
 150  IN(I)=ISM
      GO TO 1888
186   CONTINUE
      DO 180 I=1,NOPRO
      DO 180 K=1,NOV
      DO 182 KK=1,NOC
      IF(IB(KK).EQ.ID(K)) THEN
      Z(I,K)=0.
      GO TO 180
      ENDIF
 182  CONTINUE
      SUM=0
      ISM=0
      DO 183 KK=1,NOC
      LOK=IB(KK)
      SUM=SUM+C(I,LOK)*TAB(KK,K)
 183  ISM=ISM+C(I,LOK)*IV(KK)
      Z(I,K)=SUM-C(I,K)
 180  IN(I)=ISM
5     FORMAT(2X,'CORRECT',5X,'ENTER',5X,'1:YES',5X,'2:NO')
7     FORMAT(2X,'REENTER AGAIN')
       RETURN
       END
C***********************************************
C*            SUBROUTINE PIVCOL                *
C***********************************************
      SUBROUTINE PIVCOL
      DIMENSION IN(100)
      DIMENSION Z(100,100),IPINK(100)
      REAL IN,IV,LEVATT
      COMMON/B2/Z,IN,IP,IC,IW
```

```
         COMMON/B3/IPVC,XMAX,NOV,INO
         COMMON/B13/ICOUNT,IFLAG
         COMMON/B19/IBUND
C  SAVE THE NAME OF ENTERING VARIABLE INTO THE BASIS
C
         DO 5 I=1,NOV
 5       IPINK(I)=0
         IPVC=0
         XMAX=0
         IF(IBUND.EQ.0) THEN
         DO 10 J=1,NOV
         IF(XMAX.GE.Z(INO,J)) GO TO 10
         XMAX=Z(INO,J)
         IPVC=J
 10      CONTINUE
         ELSE
         DO 20 J=1,NOV
         IF(Z(INO,J).LE.0.) GO TO 20
         XMAX=Z(INO,J)
         IPVC=J
         GO TO 21
20       CONTINUE
         ENDIF
21       IF(IPVC.NE.0) THEN
         IPINK(IPVC)=IPVC
         ENDIF
         IF(ABS(XMAX).LE.1E-4) XMAX=0.
 60      INO1=INO-1
         IF(INO1.EQ.0) RETURN
         IF(IPVC.EQ.0) RETURN
         DO 40 I=1,INO1
         IF(Z(I,IPVC).LE.-.0001) THEN
         IG=1
         GO TO 80
         ELSE
         IG=0
         ENDIF
 40      CONTINUE
         IF(IG.EQ.0) RETURN
C
C PURPOSE TO FIND A PIVOT COLUMN OR AN ENTERING VARIABLE
C WHICH DOES NOT DESTROIED THE PREVIOUS PRIORITY LEVELS.
C
 80     IPV=0
        XMAX=0
        IF(IBUND.EQ.0) THEN
```

```
        DO 50 J=1,NOV
        IF(XMAX.GE.Z(INO,J)) GO TO 50
        IF(J.EQ.IPINK(J)) GO TO 50
        XMAX=Z(INO,J)
        IPV=J
 50     CONTINUE
        ELSE
        DO 85 J=1,NOV
        IF(Z(INO,J).LE.O.) GO TO 85
        IF(J.EQ.IPINK(J)) GO TO 85
        XMAX=Z(INO,J)
        IPV=J
        GO TO 86
85      CONTINUE
        ENDIF
86      IPVC=IPV
        IF(IPVC.NE.O) THEN
        IPINK(IPVC)=IPVC
        ENDIF
        IF(XMAX.EQ.O) GO TO 70
        GO TO 60
 70     RETURN
        END
C*************************************************
C*          SUBROUTINE PIVROW                    *
C*************************************************
         SUBROUTINE PIVROW(IPROW)
         DIMENSION TAB(100,100),IV(100)
         DIMENSION IREAL(50),ID(100),IB(100)
         REAL IN,IV,LEVATT
         COMMON/B1/TAB,IV,ID,IB,LIT
         COMMON/B3/IPVC,XMAX,NOV,INO
         COMMON/B4/NPRNT,NOC,NOPRO
         COMMON/B7/ICHECK,INTGP,IREAL,NXREAL
         COMMON/B15/IPAT,KRAZY
C
         IF(INTGP.EQ.1) THEN
C     CHECK FOR FEASIBILITY
         KRAZY=O
         DO 100 I=1,NOC
         IF(IV(I).LT.O.) THEN
         KRAZY=KRAZY+1
         RETURN
         ENDIF
100      CONTINUE
         ENDIF
```

```
C
C TEST FOR THE PIVOT ROW
        IPAT=0
        IFSBL=0
        IDGN=0
        RAT=200000000.
 310    DO 300 I=1,NOC
        IF(TAB(I,IPVC)) 300,300,312
 312    RATIO=IV(I)/TAB(I,IPVC)
        IF(RAT.EQ.RATIO) IDGN=1
        IF(RAT.LT.RATIO) GO TO 300
        RAT=RATIO
        IPROW=I
  300   CONTINUE
        DO 500 I=1,NOC
        IF(TAB(I,IPVC).LE.0.) THEN
        IFSBL=IFSBL+1
        ENDIF
500     CONTINUE
        IF(IFSBL.EQ.NOC) THEN
        IPAT=1
        RETURN
        ENDIF
        RETURN
        END
C************************************************
C*            SUBROUTINE CALC                  *
C************************************************
        SUBROUTINE CALC(IPROW)
        DIMENSION TAB(100,100),Z(100,100),IV(100),LEVATT(50),IN(100)
        DIMENSION TT(200,101),IB(100),ID(100),B(200,101)
        REAL IV,IN,LEVATT
        COMMON/B1/TAB,IV,ID,IB,LIT
        COMMON/B2/Z,IN,IP,IC,IW
        COMMON/B3/IPVC,XMAX,NOV,INO
        COMMON/B4/NPRNT,NOC,NOPRO
        COMMON/B6/IPEMPT,IPART,IBOUND
        COMMON/B12/ITRATN,KINPRO,LEVATT
        ITRATN=ITRATN+1
C**
        IB(IPROW)=ID(IPVC)
        IP=NOC+1
        IC=NOC+NOPRO
        IW=NOV+1
        LIT=0
        LOOP=0
```

```
      DO 5 K=1,NOC
5     IF(IV(K).GE.O) LOOP=LOOP+1
      DO 10 I=1,IC
      DO 10 J=1,IW
10    TT(I,J)=0
      DO 20 I=1,NOC
      DO 20 J=1,NOV
20    TT(I,J)=TAB(I,J)
      DO 30 I=1,NOC
30    TT(I,IW)=IV(I)
      K=0
      DO 40 I=IP,IC
      K=K+1
      DO 40 J=1,NOV
40    TT(I,J)=Z(K,J)
      K=0
      DO 50 I=IP,IC
      K=K+1
50    TT(I,IW)=IN(K)
      DO 313 I=1,IC
      DO 314 J=1,IW
      IF(I.EQ.IPROW) GO TO 315
      IF(J.EQ.IPVC) GO TO 1111
      B(I,J)=TT(I,J)-TT(IPROW,J)*TT(I,IPVC)/TT(IPROW,IPVC)
      GO TO 314
315   B(I,J)=TT(I,J)/TT(IPROW,IPVC)
      GO TO 314
1111  DO 1112  K=1,IW
      IF(K.EQ.IPROW) GO TO 1113
      B(K,J)=0
      GO TO 1112
1113  B(IPROW,IPVC)=1.
1112  CONTINUE
314   CONTINUE
313   CONTINUE
      DO 100 I=1,IC
      DO 100 J=1,IW
100   IF(ABS(B(I,J)).LE.1E-4) B(I,J)=0.
      DO 70 I=1,NOC
      DO 70  J=1,NOV
70    TAB(I,J)=B(I,J)
      DO 80 I=1,NOC
80    IV(I)=B(I,IW)
      K=0
      DO 90 I=IP,IC
      K=K+1
```

```
      DO 90 J=1,NOV
      Z(K,J)=B(I,J)
      IN(K)=B(I,IW)
      IF(ABS(IN(K)).LE.0.01) IN(K)=0.
 90   CONTINUE
      RETURN
      END
C*****************************************************
C*          SUBROUTINE PTRG                          *
C*****************************************************
      SUBROUTINE PTRG(NOV)
      DIMENSION TAB(100,100),IB(100),IV(100),ID(100),IN(100)
      DIMENSION Z(100,100),LEVATT(50),C(100,100)
      DIMENSION IPM(1000),TABB(100,100),ZZ(1,100)
      REAL IN,IV,LEVATT
      COMMON/B1/TAB,IV,ID,IB,LIT
      COMMON/B2/Z,IN,IP,IC,IW
      COMMON/B4/NPRNT,NOC,NOPRO
      COMMON/B5/NPRO,IPM,TABB,ZZ
      COMMON/B6/IPEMPT,IPART,IBOUND
      COMMON/B12/ITRATN,KINPRO,LEVATT
C
      WRITE(6,10) ITRATN
      WRITE(10,10) ITRATN
  10  FORMAT(//10X,'TABLE'//10X,'ITERATION',2X,I5)
      WRITE(6,5)
      WRITE(10,5)
 5    FORMAT(//)
      GO TO (1,2),NPRNT
1     WRITE(6,20)
      WRITE(10,20)
 20   FORMAT(10X,'BASIS',35X,'VARIABLES',40X ,'VALUES')
      WRITE(6,100)
      WRITE(10,100)
      WRITE(6,30) (ID(I),I=1,NOV)
      WRITE(10,30) (ID(I),I=1,NOV)
 30   FORMAT(4X,17(5X,I2))
      WRITE(6,100)
      WRITE(10,100)
 100  FORMAT(10X,'-----------------------------------------------
     +-------------------------------------------------')
      DO 50 I=1,NOC
      WRITE(6,60)  IB(I),(TAB(I,J),J=1,NOV),IV(I)
      WRITE(10,60) IB(I),(TAB(I,J),J=1,NOV),IV(I)
 60   FORMAT(/1X,I2,1X,5(F16.6,1X),F16.6)
 50   CONTINUE
```

```
      DO 70 I=1,NOPRO
      WRITE(6,80) (Z(I,J),J=1,NOV),IN(I)
      WRITE(10,80) (Z(I,J),J=1,NOV),IN(I)
80    FORMAT(/3X,5(F16.6,1X),F16.6)
70    CONTINUE
2     WRITE(6,101)
      WRITE(10,101)
101   FORMAT(////10X,'VARIABLES',18X,'VALUES')
      WRITE(6,5)
      WRITE(10,5)
      DO 102 I=1,NOC
      WRITE(6,103) IB(I),IV(I)
      WRITE(10,103) IB(I),IV(I)
103   FORMAT(15X,I3,10X,F16.6)
102   CONTINUE
      IF(ITRATN.EQ.0) RETURN
      WRITE(6,106)
      WRITE(10,106)
106   FORMAT(//15X,'PRIORITY',10X,'GOAL ACHIEVEMENT')
      DO 104 I=1,NOPRO
      WRITE(6,105) I,IN(I)
      WRITE(10,105) I,IN(I)
105   FORMAT(//15X,I8,10X,F16.6)
104   CONTINUE
      RETURN
      END
C**************************************************
C*          SUBROUTINE PARTG                       *
C**************************************************
      SUBROUTINE PARTG
      DIMENSION TAB(100,100),IV(100),IB(100),ID(100),Z(100,100)
      DIMENSION IN(100),IREAL(50),TAA(100,100)
      DIMENSION IPM(1000),TABB(100,100),C(100,100),ZZ(1,100)
      DIMENSION LEVATT(50),SAVE(50,100),CC(100,100)
      DIMENSION LDECS(100),LPDEV(100),LNDEV(100)
      REAL IN,IV,LEVATT
      COMMON/B1/TAB,IV,ID,IB,LIT
      COMMON/B2/Z,IN,IP,IC,IW
      COMMON/B3/IPVC,XMAX,NOV,INO
      COMMON/B4/NPRNT,NOC,NOPRO
      COMMON/B5/NPRO,IPM,TABB,ZZ
      COMMON/B6/IPEMPT,IPART,IBOUND
      COMMON/B7/ICHECK,INTGP,IREAL,NXREAL
      COMMON/B8/C
      COMMON/B10/KING
      COMMON/B11/KAT,KIT
```

```
      COMMON/B12/ITRATN,KINPRO,LEVATT
      COMMON/B15/IPAT,KRAZY
      COMMON/B17/ICALL,CC,SAVE,ISIGN
      COMMON/B20/LDECS,LPDEV,LNDEV,LTOT1,LTOT2,LTOT3
C
      WRITE(6,10)
      WRITE(10,10)
10    FORMAT(//10X,'ALGORITHM IS USING THE PARGP PROCEDURE')
      ICHECK=0
      ISIGN=0
      KINPRO=1
      INTUR=1
      CALL INTERS (INTUR)
C
C TO SAVE THE VECTOR OF PRIORITY WEIGHTS
C
      DO 18 I=1,100
      DO 18 J=1,50
18    SAVE(J,I)=0.
      DO 19 I=1,NOV
19    SAVE(1,I)=C(1,I)
      ICALL=1
C**   CALL FOR PIVOT COLUMN AND PIVOT ROW
 40   INO=1
C TO COUNT THE NUMBER PRIORITIES
C
 50   CALL PIVCOL
      IF(XMAX.EQ.0) GO TO 20
      IF(IPVC.EQ.0) GO TO 20
      CALL PIVROW(IPROW)
      CALL CALC(IPROW)
C     IF(LIT.NE.0.OR.LIT.NE.NOC) GO TO 20
      CALL PTRG(NOV)
      GO TO 50
 20   KINO=INO+1
 100  IF(KINO.GT.NPRO) GO TO 30
      LEVATT(KINPRO)=IN(1)
      KINPRO=KINPRO+1
      IOMID=1
      CALL ACHECK(ICHECK,IOMID)
      IF(ISIGN.EQ.1) RETURN
      IF(ICHECK.EQ.1) THEN
      CALL INILS(ICHECK,LX)
      ICALL=ICALL+1
      DO 17 I=1,NOV
      SAVE(ICALL,I)=CC(1,I)
```

```
17        CONTINUE
          ENDIF
60        CALL PIVCOL
          IF(XMAX.EQ.O.OR.IPVC.EQ.O) GO TO 80
          CALL PIVROW(IPROW)
          CALL CALC(IPROW)
          CALL PTRG(NOV)
          GO TO 60
 80       KINO=KINO+1
          GO TO 100
 30       RETURN
          END
C**********************************************
C*             SUBROUTINE ACHECK              *
C**********************************************
          SUBROUTINE ACHECK(ICHECK,IOMID)
          DIMENSION TAB(100,100),IV(100),IB(100),ID(100),IN(100)
          DIMENSION Z(100,100),ZZ(1,100),TABB(100,100),TAA(100,100)
          DIMENSION IPM(1000),LEVATT(50),SAVE(50,100),IDD(100)
          DIMENSION CC(100,100)
          REAL IN,IV,LEVATT
          COMMON/B1/TAB,IV,ID,IB,LIT
          COMMON/B2/Z,IN,IP,IC,IW
          COMMON/B3/IPVC,XMAX,NOV,INO
          COMMON/B4/NPRNT,NOC,NOPRO
          COMMON/B5/NPRO,IPM,TABB,ZZ
          COMMON/B6/IPEMPT,IPART,IBOUND
          COMMON/B10/KING
          COMMON/B11/KAT,KIT
          COMMON/B12/ITRATN,KINPRO,LEVATT
          COMMON/B17/ICALL,CC,SAVE,ISIGN
          GO TO (1,2),IOMID
C**  PURPOSE TO DETERMINE THE NUMBER OF ALTERNATIVE SOLUTIONS
1         K=0
          DO 10 I=1,NOV
 10       IF(Z(1,I).EQ.O.) K=K+1
C**   TO CHECK FOR ALTERNATIVE SOLUTIONS
C ICHEK=1 INDICATES THAT THERE EXIST AT LEAST ONE ALTER  SOLUTION.
          NZRO=K-NOC
          IF(NZRO.GE.1) THEN
          ICHECK=1
          ISIGN=O
          ELSE
C THE PRESENT SOLUTION IS OPTIMAL FOR THE ORIGINAL PROBLEM WITH
C RESPECT TO ALL PRIORITIES
C
```

```
          ICHECK=0
          ISIGN=1
          ENDIF
C** TO DETERMINE A NONBASIC COLUMN WITH NEGATIVE CRITERION COEFF.
2         K=0
          DO 20 I=1,NOV
C**       IPM(I) INDICATES THE SPECIFIC NONBASIC COLUMN
          IPM(I)=0
          DO 30 J=1,NOC
30        IF(ID(I).EQ.IB(J)) GO TO 20
          IF(Z(1,I).LT.0.) THEN
C** TO COUNT THE NUMBER OF NONBASIC COLUMNS
          K=K+1
          IPM(I)=I
          ENDIF
 20       CONTINUE
          NOVV=NOV-K
          DO 40 I=1,NOVV
          DO 40 J=1,NOC
          ZZ(1,I)=0.
 40       TABB(I,J)=0.
C***
C***       KING COUNTS NUMBER OF VARIABLES
C***
          IF(KAT.EQ.0) THEN
          KING=NOV
          ELSE
          KING=KING+KAT
          ENDIF
          IDID=0
          IDO=0
          DO 60 I=1,NOV
          IF(I.EQ.IPM(I)) GO TO 150
          IDID=IDID+1
          IDO=IDO+1
          ZZ(1,IDO)=Z(1,IDID)
          DO 70 J=1,NOC
 70       TABB(J,IDO)=TAB(J,IDID)
          IDD(IDO)=ID(IDID)
          GO TO 60
 150      IDID=IDID+1
 60       CONTINUE
C***
C*** NOV INDICATES THE NUMBER OF VARIABLES CONSIDERING
C*** THE FACT THAT SOME OF THEM CAN BE ELEMINATED
C*** DURING PREVIOUS ITERATIONS.
```

```
C***
         NOV=IDO
         DO 80 I=1,NOV
         DO 90 J=1,NOC
 90      TAB(J,I)=TABB(J,I)
         ID(I)=IDD(I)
 80      Z(1,I)=ZZ(1,I)
         CALL PTRG(NOV)
         IF(ICHECK.EQ.1.AND.ISIGN.EQ.0) RETURN
         IF(ICHECK.EQ.0.AND.ISIGN.EQ.1) THEN
         DO 50 LOLY=KINPRO,NPRO
         K=LOLY+1
         IN(K)=0.
         CALL INILS(ICHECK,LX)
         ICALL=ICALL+1
         DO 9 I=1,NOC
         SAVE(ICALL,I)=CC(1,I)
9        CONTINUE
         DO 12 I=1,NOC
         DO 13 J=1,KIT
         IF(IB(I).EQ.ID(J)) THEN
         LOCT=IB(I)
         IN(K)=IN(K)+SAVE(K,LOCT)*IV(I)
         LEVATT(K)=IN(K)
         GO TO 12
         ENDIF
 13      CONTINUE
 12      CONTINUE
 50      CONTINUE
         CALL PTRG(NOV)
         ENDIF
         RETURN
         END
C*************************************************
C*            SUBROUTINE INILS                   *
C*************************************************
         SUBROUTINE INILS(ICHECK,LX)
         DIMENSION TAB(100,100),IV(100),IB(100),ID(100),IN(100)
         DIMENSION TAA(100,100),TABB(100,100),C(100,100)
         DIMENSION LEVATT(50),Z(100,100),ZZ(1,100),IPM(1000)
         DIMENSION TBC(100,100),SAVE(50,100),CC(100,100)
         COMMON/B1/TAB,IV,ID,IB,LIT
         COMMON/B2/Z,IN,IP,IC,IW
         COMMON/B3/IPVC,XMAX,NOV,INO
         COMMON/B4/NPRNT,NOC,NOPRO
         COMMON/B5/NPRO,IPM,TABB,ZZ
```

```
      COMMON/B6/IPEMPT,IPART,IBOUND
      COMMON/B8/C
      COMMON/B9/NOC1,NOC2
      COMMON/B10/KING
      COMMON/B12/ITRATN,KINPRO,LEVATT
      COMMON/B17/ICALL,CC,SAVE,ISIGN
      REAL IN,IV,LEVATT
      NOC1=NOC
      CALL ADDUP(CC)
C
      NOC2=NOC
      NOC3=NOC2-NOC1
      NNOC=NOC1+1
      DO 5 NBC=1,NOC3
      DO 30 J=1,NOV
      DO 20 I=1,NOC1
      IF(IB(I).EQ.ID(J)) THEN
      L=I
      IF(TAB(NNOC,J).NE.O.) THEN
      IF(TAB(L,J).EQ.O.) GO TO 30
      DD=TAB(NNOC,J)/TAB(L,J)
      DO 10 K=1,NOV
      TAA(L,K)=-DD*TAB(L,K)
      MARY=NOV+1
      TAA(L,MARY)=-DD*IV(L)
      TBC(NNOC,K)=TAB(NNOC,K)+TAA(L,K)
      TAB(NNOC,K)=TBC(NNOC,K)
10    CONTINUE
      IV(NNOC)=IV(NNOC)+TAA(L,MARY)
      GO TO 30
      ENDIF
      ENDIF
 20   CONTINUE
 30   CONTINUE
      NNOC=NNOC+1
 5    CONTINUE
C
      NOC3=NOC2-NOC1
      NNOC=NOC1+1
      DO 6 NBC=1,NOC3
      IF(IV(NNOC).GE.O) GO TO 40
      KNNN=NNOC-1
      DO 31 J=1,NOV
      IF(TAB(NNOC,J).EQ.-1.) THEN
      K=O
      DO 32 I=1,KNNN
```

```
      IF(TAB(I,J).EQ.O.) K=K+1
32    CONTINUE
      IF(K.EQ.KNNN) THEN
      IET=ID(J)
      IYES=J
      GO TO 34
      ENDIF
      ENDIF
 31   CONTINUE
      GO TO 40
 34   DO 33 J=1,NOV
      TAB(NNOC,J)=-TAB(NNOC,J)
      IF(ABS(TAB(NNOC,J)).LE.0.0002) TAB(NNOC,J)=0.
 33   CONTINUE
      IPD=IB(NNOC)
      IV(NNOC)=-IV(NNOC)
      IF(ABS(IV(NNOC)).LE.0.0002) IV(NNOC)=0.
      IB(NNOC)=IET
      PJET=CC(1,IET)
      CC(1,IET)=CC(1,IPD)
      CC(1,IPD)=PJET
      DO 42 KJ=1,NOV
 42   IF(ID(KJ).EQ.IPD) INNO=KJ
      POTT=C(1,IYES)
      C(1,IYES)=C(1,INNO)
      C(1,INNO)=POTT
40    NNOC=NNOC+1
 6    CONTINUE
      KING1=KING+1
      DO 70 I=1,NOV
      DO 80 J=1,NOC
      IF(IB(J).EQ.ID(I)) THEN
      Z(1,I)=0.
      GO TO 70
      ENDIF
 80   CONTINUE
      BSUM=0.
      ASUM=0.
      DO 90 J=1,NOC
      K=IB(J)
      ASUM=ASUM+CC(1,K)*TAB(J,I)
      BSUM=BSUM+CC(1,K)*IV(J)
 90   CONTINUE
      Z(1,I)=ASUM-C(1,I)
      IN(1)=BSUM
 70   CONTINUE
```

```
      CALL PTRG(NOV)
      RETURN
      END
C*************************************************
C*         SUBROUTINE ADDUP                      *
C*************************************************
      SUBROUTINE ADDUP(CC)
      DIMENSION TAB(100,100),Z(100,100),ID(100),IB(100)
      DIMENSION IV(100),IN(100),C(100,100),TABB(100,100)
      DIMENSION LEVATT(50),ZZ(1,100),IPM(1000)
      DIMENSION CC(100,100)
      DIMENSION ZXY(80,100)
      REAL IN,IV,LEVATT
      COMMON/B1/TAB,IV,ID,IB,LIT
      COMMON/B2/Z,IN,IP,IC,IW
      COMMON/B3/IPVC,XMAX,NOV,INO
      COMMON/B4/NPRNT,NOC,NOPRO
      COMMON/B5/NPRO,IPM,TABB,ZZ
      COMMON/B6/IPEMPT,IPART,IBOUND
      COMMON/B8/C
      COMMON/B10/KING
      COMMON/B11/KAT,KIT
      COMMON/B12/ITRATN,KINPRO,LEVATT
      NOCP=NOC
      NOVP=NOV+1
2     WRITE(6,1)
      WRITE(10,1)
 1    FORMAT(10X,'-->',2X,'ENTER THE NUMBER OF NEW CONST. AND'
     +/15X,'NUMBER OF NEW VARIABLES')
      READ(5,*) KNOC,KNOV
      KAT=KNOV
      WRITE(6,133) KNOC,KNOV
      WRITE(10,133) KNOC,KNOV
133   FORMAT(10X,'KNOC=',I2,5X,'KNOV=',I2)
      WRITE(6,114)
      WRITE(10,114)
      READ(5,*) ICORR
      IF(ICORR.EQ.2) THEN
      WRITE(6,115)
      WRITE(10,115)
      GO TO 2
      ENDIF
      IF(KNOV.EQ.0.AND.KNOC.EQ.0) GO TO 1079
C TO CONSIDER A SPECIAL SITUATION
C THIS IS THE CASE WHEN KNOV=0 AND KNOC IS GREATER ZERO
      IF(KNOC.NE.0.AND.KNOV.EQ.0) THEN
```

```
      NOC=NOC+KNOC
      GO TO 2101
      ENDIF
      NOC=NOC+KNOC
      INOV1=NOV+1
      NOV=NOV+KNOV
      K=KING
      DO 10 I=1,KNOV
      K=K+1
      ID(INOV1)=K
      IPM(K)=0
      INOV1=INOV1+1
 10   CONTINUE
C KIT COUNTES THE TOTAL NUMBER OF VARIABLES
       KIT=KING+KNOV
       DO 20 I=NOVP,NOV
       DO 20 J=1,NOCP
 20    TAB(J,I)=0.
C**   PURPOSE TO READ THE VALUE OF EACH ELEMENT OF EACH CONST.
       KING1=KING+1
2101   DO 2000 K=1,KNOC
       DO 2000 I=1,KIT
 2000  ZXY(K,I)=0.
C
C
       WRITE(6,2007)
       WRITE(10,2007)
2007   FORMAT(5X,'-->',2X,'ENTER # OF NONZERO ELEMENTS IN THE'
     +/10X,'NEW CONSTRAINTS')
C
       READ(5,*) NNZRO
       WRITE(6,3) NNZRO
       WRITE(10,3) NNZRO
3      FORMAT('# OF NONZERO ELEMENTS = ',5X,I5)
       WRITE(6,4)
       WRITE(10,4)
4      FORMAT(5X,'-->',2X,'ENTER ROW I , COLUMN J AND ITS VALUE'/10X
     +,'LEAVE ONE SPACE BETWEEN THE ENTRIES')
       DO 2008 I=1,NNZRO
5      READ(5,*) L,M,VALUE
       WRITE(6,116) L,M,VALUE
       WRITE(10,116) L,M,VALUE
       WRITE(6,114)
       WRITE(10,114)
       READ(5,*) ICORR
       IF(ICORR.EQ.2) THEN
```

```
          WRITE(6,115)
          WRITE(10,115)
          GO TO 5
          ENDIF
          ISOR=L-NOCP
2008      ZXY(ISOR,M)=VALUE
C
          DO 2003 J=1,NOV
          IDONE=ID(J)
          DO 2004 N=1,KIT
          IF(IDONE.EQ.N) THEN
          DO 2001 K=1,KNOC
          L=K+NOCP
          TAB(L,J)=ZXY(K,N)
2001      CONTINUE
          GO TO 2003
          ENDIF
2004      CONTINUE
2003      CONTINUE
C**  PURPOSE TO READ THE BASIS AND RIGHT HAND SIDE VALUES OF
C**  NEW CONSTRAINTES
C*
        WRITE(6,6)
        WRITE(10,6)
6       FORMAT(5X,'-->',2X,'ENTER ROW I, COLUMN J AND THE RHS VALUES')
        DO 33339 I=1,KNOC
7       READ(5,*) L,IBB,VIV
        WRITE(6,117)
        WRITE(10,117)
117     FORMAT('ROW I =',2X,I3,2X,'BASIS=',2X,I3,2X,'RHS=',2X,F8.4)
        WRITE(6,114)
        WRITE(10,114)
        READ(5,*) ICORR
        IF(ICORR.EQ.2) THEN
        WRITE(6,115)
        WRITE(10,115)
        GO TO 7
        ENDIF
        IV(L)=VIV
        IB(L)=IBB
33339   CONTINUE
C***
        KING1=KING+1
1079    WRITE(6,2009)
        WRITE(10,2009)
2009    FORMAT(5X,'-->',2X,'ENTER # OF NONZERO ELEMENTS IN THE'
```

```
     +/10X,'NEW PRIORITY MATRIX')
        READ(5,*) IPZRO
        DO 112 I=1,KIT
112     CC(1,I)=0.
        WRITE(6,118)
        WRITE(10,118)
118     FORMAT(5X,'-->',2X,'ENTER VAR #,2X,ITS PRIORITY WEIGHT')
        DO 113 I=1,IPZRO
        WRITE(6,9)
        WRITE(10,9)
9       FORMAT('NUMBER OF NONZERO ELEMENTS=',2X,I5)
8       READ(5,*) L,VALUE
        WRITE(6,119) L,VALUE
        WRITE(10,119) L,VALUE
119     FORMAT('VARIABLE=',2X,I3,2X,'PRIORITY WEIGHT=',2X,F8.4)
        WRITE(6,114)
        WRITE(10,114)
        READ(5,*) ICORR
        IF(ICORR.EQ.2) THEN
        WRITE(6,115)
        WRITE(10,115)
        GO TO 8
        ENDIF
113     CC(1,L)=VALUE
        IBET=0
        DO 110 I=1,KIT
        IF(I.GE.KING1) GO TO 1999
        IF(IPM(I).EQ.I) GO TO 110
 1999   IBET=IBET+1
        C(1,IBET)=CC(1,I)
 110    CONTINUE
114     FORMAT(2X,'CORRECT',2X,'ENTER',2X,'1:YES',2X,'2:NO')
115     FORMAT('REENTER AGAIN')
116     FORMAT(2X,'ROW I=',2X,I3,'COLUMN J=',2X,I3,2X,'VALUE=',2X,
     +F8.4)
        RETURN
        END
C*************************************************
C*            SUBROUTINE DUALS                  *
C*************************************************
C**
           SUBROUTINE DUALSX(IPROW,LAB,IBALL)
           DIMENSION TAB(100,100),IV(100),IB(100),ID(100)
           DIMENSION LEVATT(50),Z(100,100),ZZ(1,100),B(200,101)
           DIMENSION TT(100,100),IPM(1000),TABB(100,100),IN(100)
           DIMENSION ISVZ(100),PRLV(50,60)
```

```
            REAL IN,IV,LEVATT
            COMMON/B1/TAB,IV,ID,IB,LIT
            COMMON/B2/Z,IN,IP,IC,IW
            COMMON/B3/IPVC,XMAX,NOV,INO
            COMMON/B4/NPRNT,NOC,NOPRC
            COMMON/B5/NPRO,IPM,TABB,ZZ
            COMMON/B6/IPEMPT,IPART,IBOUND
            COMMON/B9/NOC1,NOC2
            COMMON/B12/ITRATN,KINPRO,LEVATT
C
C** TO FIND THE PIVOT ROW
            LAB=O
            IBALL=O
 40         AMOST=O.
            DO 10 I=1,NOC2
            IF(IV(I).GE.AMOST) GO TO 10
            I1=I
            AMOST=IV(I1)
10          CONTINUE
            IF(AMOST.EQ.O.) RETURN
            R=1.E+10
            IPROW=I1
C**   TO FIND THE PIVOT COLUMN
            IF(IPART.EQ.1) THEN
            J1=O
            DO 20 J=1,NOV
            IF(TAB(I1,J).GE.O.) GO TO 20
            IF(Z(1,J).GE.O.) GO TO 20
            RR=Z(1,J)/TAB(I1,J)
            IF(RR.LT.R) J1=J
            IF(RR.LT.R) R=RR
 20         CONTINUE
            IPVC=J1
            IF(IPVC.EQ.O) THEN
            IBALL=1
            RETURN
            ENDIF
            ENDIF
C CONSIDER A SITUATION WHEN PREEMPTIVE GOAL PROGRAMMING IS CONCERNED
C
         8F(IPEMPT.EQ.1) THEN
         IKOT=O
         DO 61 KL=1,NOV
         IF(TAB(I1,KL).GE.O) GO TO 61
         IKOT=IKOT+1
         ISVZ(IKOT)=KL
```

```
      DO 60 K=1,NOPRO
      IF(Z(K,KL).GE.0) THEN
      PRLV(K,IKOT)=0.
      GO TO 60
      ENDIF
      PRLV(K,IKOT)=ABS(Z(K,KL)/TAB(I1,KL))
60    CONTINUE
61    CONTINUE
      I=1
      L=2
      IPVC=0
90    IF(L.GT.IKOT) GO TO 92
      DO 91 J=1,NOPRO
      IF(PRLV(J,I).EQ.PRLV(J,L)) GO TO 91
      IF(PRLV(J,I).LT.PRLV(J,L)) THEN
      ILL=I
      I=ILL
      L=L+1
      IPVC=ISVZ(I)
      GO TO 90
      ELSE
      ILL=L
      I=ILL
      IPVC=ISVZ(L)
      L=L+1
      GO TO 90
      ENDIF
91    CONTINUE
      ENDIF
C
92    CALL CALC(IPROW)
      CALL PTRG(NOV)
C** CHECK FOR OTHER NEGATIVE RIGHTE HAND SIDE VALUES
      J=0
      DO 30 I=1,NOC
30    IF(IV(I).GE.0) J=J+1
      IF(J.EQ.NOC) THEN
      DO 51 INO=1,NOPRO
65    CALL PIVCOL
      IF(XMAX.EQ.0.OR.IPVC.EQ.0) GO TO 52
      CALL PIVROW(IPROW)
      CALL CALC(IPROW)
      CALL PTRG(NOV)
      IF(INO.EQ.1) GO TO 65
51    CONTINUE
 52   LAB=1
```

```
      GO TO 50
      ENDIF
      GO TO 40
 50   RETURN
      END
C**************************************************
C*          SUBROUTINE FACT1                       *
C**************************************************
C**
        SUBROUTINE FACT1
        DIMENSION IB(100),IREAL(50),IV(100),IN(100)
        DIMENSION DUM(100),IPM(1000),TABB(100,100),Z(100,100)
        DIMENSION ID(100),ZZ(1,100),LEVATT(50),TAB(100,100)
        DIMENSION ZPRO(100,100),INPRO(100),ZPNEW(100,100)
        DIMENSION LDECS(100),LPDEV(100),LNDEV(100)
        REAL IN,IV,INPRO,LEVATT
        INTEGER ZROCUT
        COMMON/B1/TAB,IV,ID,IB,LIT
        COMMON/B2/Z,IN,IP,IC,IW
        COMMON/B3/IPVC,XMAX,NOV,INO
        COMMON/B4/NPRNT,NOC,NOPRO
        COMMON/B5/NPRO,IPM,TABB,ZZ
        COMMON/B6/IPEMPT,IPART,IBOUND
        COMMON/B7/ICHECK,INTGP,IREAL,NXREAL
        COMMON/B10/KING
        COMMON/B11/KAT,KIT
        COMMON/B12/ITRATN,KINPRO,LEVATT
        COMMON/B13/ICOUNT,IFLAG
        COMMON/B14/ZPRO,INPRO,KPRIOR,ZPNEW,MVARR,MROWSS
        COMMON/B15/IPAT,KRAZY
        COMMON/B20/LDECS,LPDEV,LNDEV,LTOT1,LTOT2,LTOT3
C
C PURPOSE TO CONTROL THE PROGRAM FOR LIPREGP PROBLEM
C
        IFLAG=1
        MROWSS=NOC
        MVARR=NOV
        KPRIOR=NPRO+1
C MOVE THE FIRST PRIORITY LEVEL INTO THE SECOND LEVEL IN
C ORDER TO SAVE A POSITION FOR THE NEW ABSOLUTE PRIORITY
C LEVEL
C
        K=1
        DO 101 I=1,NPRO
        K=K+1
        INPRO(K)=IN(I)
```

```
          DO 102 J=1,NOV
          ZPRO(K,J)=0.0
          ZPRO(K,J)=Z(I,J)
 102      CONTINUE
 101      CONTINUE
          IDEN=NOV-NXREAL
          IDECID=0
          IF(IDEN.GE.1) IDECID=1
C TO KEEP THE NUMBER OF EXISTING PRIORITY LEVELS
21        L=K
          CALL INTGR(L,IDECID,ZROCUT)
          IF(ZROCUT.EQ.1) GO TO 2
          GO TO (1,2),ICOUNT
C TO START FROM THE FIRST PRIORITY LEVEL
 1        INO=1
          DO 5 INO=1,L
 10       CALL PIVCOL
          IF(XMAX.EQ.0) GO TO 5
          IF(IPVC.EQ.0) GO TO 5
          CALL PIVROW(IPROW)
          CALL CALC(IPROW)
          IF(ITRATN.EQ.200) RETURN
          CALL PTRG(NOV)
          IF(INO.EQ.1) GO TO 10
          IF(IN(INO).EQ.0) GO TO 5
 5        CONTINUE
          GO TO 21
2         ISUMMY=3
          CALL SUMMRY(IB,IN,NOPRO,IV,JZJJ,NOC,ISUMMY)
          RETURN
          END
C**************************************************
C*         SUBROUTINE FACT2                       *
C**************************************************
C
          SUBROUTINE FACT2
          DIMENSION IB(100),IREAL(50),IV(100),LEVATT(50),IN(100)
          DIMENSION DUM(100),IPM(1000),TABB(100,100) ,Z(100,100)
          DIMENSION TAB(100,100),ZPRO(100,100),ID(100),ZZ(1,100)
          DIMENSION LDECS(100),LPDEV(100),LNDEV(100)
          DIMENSION INPRO(100),ZPNEW(100,100)
          REAL IN,IV,INPRO,LEVATT
          INTEGER ZROCUT
          COMMON/B1/TAB,IV,ID,IB,LIT
          COMMON/B2/Z,IN,IP,IC,IW
          COMMON/B3/IPVC,XMAX,NOV,INO
```

```
      COMMON/B4/NPRNT,NOC,NOPRO
      COMMON/B5/NPRO,IPM,TABB,ZZ
      COMMON/B6/IPEMPT,IPART,IBOUND
      COMMON/B7/ICHECK,INTGP,IREAL,NXREAL
      COMMON/B10/KING
      COMMON/B11/KAT,KIT
      COMMON/B12/ITRATN,KINPRO,LEVATT
      COMMON/B13/ICOUNT,IFLAG
      COMMON/B14/ZPRO,INPRO,KPRIOR,ZPNEW,MVARR,MROWSS
      COMMON/B15/IPAT,KRAZY
      COMMON/B20/LDECS,LPDEV,LNDEV,LTOT1,LTOT2,LTOT3
C
C PURPOSE TO CONTROL THE PROGRAM FOR LIPARGP PROBLEM
C
      IFLAG=1
      MROWSS=NOC
      KPRIOR=NPRO+1
      IOMID=2
      CALL ACHECK(ICHECK,IOMID)
      MVARR=NOV
      K=2
      INPRO(K)=IN(1)
      DO 19 J=1,100
19    ZPRO(K,J)=0.0
      DO 22 J=1,NOV
      ZPRO(K,J)=Z(1,J)
22    CONTINUE
C TO DETERMINE THE TYPE OF THE CUTTING PLANE THAT SHOULD BE USED
      IDEN=NOV-NXREAL
      IDECID=0
      IF(IDEN.GE.1) IDECID=1
C  TO KEEP THE NUMBER OF EXISTING PRIORITY LEVELS
C
21    L=K
      CALL INTGR(L,IDECID,ZROCUT)
      IF(ZROCUT.EQ.1) GO TO 2
      GO TO (1,2),ICOUNT
1     INO=1
      DO 5 INO=1,L
10    CALL PIVCOL
      IF(XMAX.EQ.0.OR.IPVC.EQ.0) GO TO 5
      CALL PIVROW(IPROW)
      NOPRO=2
      CALL CALC(IPROW)
      IF(ITRATN.EQ.20) RETURN
      CALL PTRG(NOV)
```

```
      IF(INO.EQ.1) GO TO 10
 5    CONTINUE
      GO TO 21
 2    ISUMMY=3
      CALL SUMMRY(IB,IN,NOPRO,IV,JZJJ,NOC,ISUMMY)
      RETURN
      END
C**************************************************
C*          SUBROUTINE INTGR                       *
C**************************************************
        SUBROUTINE INTGR(L,IDECID,ZROCUT)
        DIMENSION IB(100),IV(100),IN(100)
        DIMENSION ZPRO(100,100),INPRO(100)
        DIMENSION ZPNEW(100,100),IREAL(50)
        DIMENSION DUM(100),IPM(1000),LEVATT(50),ZZ(1,100)
        DIMENSION TABB(100,100),Z(100,100),ID(100),TAB(100,100)
        REAL IN,IV,INPRO,LEVATT
        INTEGER ZROCUT
        COMMON/B1/TAB,IV,ID,IB,LIT
        COMMON/B2/Z,IN,IP,IC,IW
        COMMON/B3/IPVC,XMAX,NOV,INO
        COMMON/B4/NPRNT,NOC,NOPRO
        COMMON/B5/NPRO,IPM,TABB,ZZ
        COMMON/B6/IPEMPT,IPART,IBOUND
        COMMON/B7/ICHECK,INTGP,IREAL,NXREAL
        COMMON/B12/ITRATN,KINPRO,LEVATT
        COMMON/B13/ICOUNT,IFLAG
        COMMON/B14/ZPRO,INPRO,KPRIOR,ZPNEW,MVARR,MROWSS
C**  PURPOSE TO DETERMINE THE MAXIMUM FRACTION OF THE
C**  RIGHT HAND SIDE.
C**
        IF(NXREAL.NE.0) THEN
        CALL GOMORY(JB,KROWS,FMAX)
        IF(JB.EQ.1) THEN
        WRITE(6,106)
        WRITE(10,106)
106     FORMAT(//10X,'THE REQUIRED VARIABLES ARE INTEGERE VALUED')
        ICOUNT=2
        RETURN
        ELSE
        IROW=KROWS
        GO TO 107
        ENDIF
        ENDIF
        FMAX=0.0
        DO 11 I=1,NOC
```

```
          DO 10 J=1,MVARR
          IF(IPEMPT.EQ.1) THEN
          IF(IB(I).NE.J) GO TO 10
          GO TO 3
          ENDIF
C USING PARTITIONNING GOAL PROGRAMMING PROCEDURE
          IF(IPART.EQ.1) THEN
          LOCD=ID(J)
          IF(IB(I).NE.LOCD) GO TO 10
          GO TO 3
          ENDIF
3         IRHS=IV(I)
          FRACT=IV(I)-IRHS
          IF(FRACT.LT.0.98) GO TO 1
          IV(I)=IV(I)-FRACT+1.0
          FRACT=0.0
 1        IF(FMAX-FRACT) 5,11,11
 5        FMAX=FRACT
          IROW=I
          GO TO 11
 10       CONTINUE
 11       CONTINUE
107       IF(FMAX.LE.0.1) THEN
          ICOUNT=2
          IFLAG=2
          RETURN
          ENDIF
C**
C** TO DEVELOP THE GOMORY CUTTING PLANE CONSTRAINT
C**
          IF(IDECID.EQ.0) THEN
          DO 25 J=1,NOV
          ITT =TAB(IROW,J)
          FRACT=TAB(IROW,J)-ITT
          IF(ABS(FRACT).LE.0.00001) GO TO 20
          IF(FRACT.GE.0.0) GO TO 15
          DUM(J)=1.0+FRACT
          GO TO 25
 15       DUM(J)=FRACT
          GO TO 25
 20       DUM(J)=0.0
25        CONTINUE
          ELSE
C PURPOSE TO DETERMINE A GOMORY CUTTING PLANE CONSTRAINTE
C FOR THE MIXED INTEGER VALUES.
C
```

```
          WRITE(6,1004)
          WRITE(10,1004)
1004      FORMAT(//20X,'*** PROGRAM IS USING MIXED INTEGER PROC')
          DO 1000 K=1,NOV
          KABA=0
          DO 1001 J=1,NXREAL
1001      IF(IREAL(J).EQ.K) KABA=KABA+1
C
C
          IF(KABA.EQ.1) GO TO 1002
C
C
          IF(TAB(IROW,K).GE.0) THEN
          DUM(K)=TAB(IROW,K)
          ELSE
          DUM(K)=(FMAX/(FMAX-1.))*TAB(IROW,K)
          ENDIF
          GO TO 1000
C FOR INTEGER VALUES
1002      ITT=TAB(IROW,K)
          FRACT=TAB(IROW,K)-ITT
          IF(ABS(FRACT).LE.FMAX) THEN
          DUM(K)=FRACT
          ELSE
          DUM(K)=(FMAX/(1.-FMAX))*(1-FRACT)
          ENDIF
1000      CONTINUE
          ENDIF
C CHECK FOR  ZERO-CUTTING PLANE. SUCH CUT MAY EXIST WHEN PARTITION
C ING GP PROCEDURE IS USED TO SOLVE A SMALL SIZED PROBLEM.
C
          ZROCUT=0
          ICOTT=0
          DO 260 ILL=1,NOV
          IF(DUM(ILL)) 260,27,260
27        ICOTT=ICOTT+1
260       CONTINUE
          IF(ICOTT.EQ.NOV) THEN
          ZROCUT=1
          WRITE(6,28)
          WRITE(10,28)
28        FORMAT(//10X,'A ZERO-CUT IS DETECTED')
          IF(IPART.EQ.1) THEN
          WRITE(6,29)
          WRITE(10,29)
29        FORMAT(//10X,'TRY ANOTHER PROCEDURE'//10X,'USE PREEMPTIVE
```

```
     +    GP PROCEDURE')
          RETURN
          ENDIF
          IF(IPEMPT.EQ.1) THEN
          WRITE(6,30)
          WRITE(10,30)
30        FORMAT(//10X,'TRY ANOTHER PROCEDURE'//10X,'USE PARTIT
     +    IONNIG GP PROCEDURE')
          ENDIF
          RETURN
          ENDIF
C**  ADD CUTTING PLANE FRACTION TO THE FINAL TABLEUA
C USING PREEMPTIVE GOAL PROGRAMMING PROCEDURE
C
          IF(IPEMPT.EQ.1) THEN
          NVAR1=NOV+1
          NVAR=NOV+2
          NCONS=NOC+1
          ID(NVAR1)=NVAR1
          ID(NVAR)=NVAR
          IB(NCONS)=NVAR
          DO 70 I=1,NOC
          DO 70 J=NVAR1,NVAR
 70       TAB(I,J)=0.0
          DO 71 I=2,L
          DO 71 J=NVAR1,NVAR
71        Z(I,J)=0.0
          TAB(NCONS,NVAR1)=-1.0
          TAB(NCONS,NVAR)=1.0
          DO 90 J=1,NOV
 90       TAB(NCONS,J)=DUM(J)
          IV(NCONS)=FMAX
          GO TO 26
          ENDIF
C USING PARTITIONNING GOAL PROGRAMMING PROCEDURE
C
          IF(IPART.EQ.1) THEN
          LOCT=ID(NOV)
          NVAR1=LOCT+1
          NVAR2=LOCT+2
          NCONS=NOC+1
          NVV1=NOV+1
          NVV2=NOV+2
          ID(NVV1)=NVAR1
          ID(NVV2)=NVAR2
          IB(NCONS)=NVAR2
```

```
          DO 81 I=1,NOC
          DO 81 J=NVV1,NVV2
81        TAB(I,J)=0.0
          DO 83 I=2,L
          DO 83 J=NVV1,NVV2
83        Z(I,J)=0.0
          TAB(NCONS,NVV1)=-1.
          TAB(NCONS,NVV2)=1.
C TO ORGANIZE THE NEW ROW CONSTRAINT FOR THE SIMPLEX TABLEAU
         DO 82 J=1,NOV
  82     TAB(NCONS,J)=DUM(J)
         IV(NCONS)=FMAX
         NVAR=NVV2
         GO TO 26
         ENDIF
C**ADD A NEW PRIORITY TO THE PRIORITY MATRIX
 26      CONTINUE
C USING PREEMPTIVE GOAL PROGRAMMING PROCEDURE
          IF(IPEMPT.EQ.1) THEN
          DO 91 I=1,NVAR
          DO 95 J=1,NCONS
          IF(IB(J).EQ.I) THEN
          ZPRO(1,I)=0.0
          GO TO 91
          ELSE
          ZPRO(1,I)=TAB(NCONS,I)
          ENDIF
95        CONTINUE
91        CONTINUE
          DO 92 I=2,L
          DO 92 J=NVAR1,NVAR
 92       ZPRO(I,J)=0.0
          ZPRO(1,NVAR1)=-1.
          ZPRO(1,NVAR)=0.0
          GO TO 94
          ENDIF
          IF(IPART.EQ.1) THEN
          DO 191 I=1,NVV2
          DO 192 J=1,NCONS
          IF(IB(J).EQ.ID(I)) THEN
          ZPRO(1,I)=0.0
          GO TO 191
          ELSE
          ZPRO(1,I)=TAB(NCONS,I)
          ENDIF
192       CONTINUE
```

```
191       CONTINUE
          ZPRO(1,NVV1)=-1.
          ZPRO(1,NVV2)=0.0
          ZPRO(2,NVV1)=0.0
          ZPRO(2,NVV2)=0.0
          ENDIF
C**    TO DETERMINE THE VALUE OF THE NEW PRIORITY LEVEL
C**
 94       CONTINUE
          ICARE=2
          CALL SADD(IFLAG,NCONS,ICARE,LINE)
C**
C**  TO UPDATE NUMBER OF CONSTRAINTS,VARIABLES AND PRIORITIES.
          NOV=NVAR
          NOC=NCONS
          NPRO=KPRIOR
          NOPRO=NPRO
          IF(IFLAG.EQ.2) THEN
          DO 1103 J=1,NOV
          Z(1,J)=ZPRO(1,J)
          IN(1)=INPRO(1)
1103      CONTINUE
C SET THE Z VALUES OF THE NEW VARIABLES TO ZERO
         IF(IPART.EQ.1) THEN
         Z(2,NVV1)=0.0
         Z(2,NVV2)=0.0
         ENDIF
C USING PREEMPTIVE GP PROCEDURE
C
         IF(IPEMPT.EQ.1) THEN
         Z(2,NVAR1)=0.0
         Z(2,NVAR)=0.0
         ENDIF
         ENDIF
         IF(IFLAG.EQ.1) THEN
         DO 103 I=1,L
         DO 103 J=1,NOV
         Z(I,J)=ZPRO(I,J)
         IN(I)=INPRO(I)
 103     CONTINUE
         ENDIF
         ICOUNT=1
         IFLAG=2
C USING PARTITIONNING GP PROCEDURE
         IF(IPART.EQ.1) THEN
         NOPRO=2
```

```
      CALL PTRG(NOV)
      RETURN
      ENDIF
C USING PREEMPTIVE GP PROCEDURE
C
      IF(IPEMPT.EQ.1) THEN
      CALL PTRG(NOV)
      ENDIF
      RETURN
      END
C***********************************************************
C*          SUBROUTINE SADD                               *
C***********************************************************
      SUBROUTINE SADD(IFLAG,NCONS,ICARE,LINE)
      DIMENSION TAB(100,100),IV(100),IB(100),ID(100),IN(100)
      DIMENSION Z(100,100),ZPNEW(100,100),ZPRO(100,100),INPRO(100)
      DIMENSION TABB(100,100),IPM(1000),LEVATT(50),ZZ(1,100)
      REAL IN,IV,INPRO,LEVATT
      COMMON/B1/TAB,IV,ID,IB,LIT
      COMMON/B2/Z,IN,IP,IC,IW
      COMMON/B3/IPVC,XMAX,NOV,INO
      COMMON/B4/NPRNT,NOC,NOPRO
      COMMON/B5/NPRO,IPM,TABB,ZZ
      COMMON/B6/IPEMPT,IPART,IBOUND
      COMMON/B12/ITRATN,KINPRO,LEVATT
      COMMON/B14/ZPRO,INPRO,KPRIOR,ZPNEW,MVARR,MROWSS
      GO TO (1,2),ICARE
1     IFLAG=1
      IF(IPART.EQ.1) THEN
      K=2
      INPRO(K)=IN(1)
      DO 10 J=1,100
10    ZPRO(K,J)=0.0
      DO 20 J=1,NOV
      ZPRO(K,J)=Z(1,J)
 20   CONTINUE
      GO TO 50
      ENDIF
      IF(IPEMPT.EQ.1) THEN
      K=1
      KAKE=NPRO-1
      DO 30 I=1,KAKE
      K=K+1
      INPRO(K)=IN(I)
      DO 40 J=1,NOV
      ZPRO(K,J)=0.
```

```
        ZPRO(K,J)=Z(I,J)
40      CONTINUE
30      CONTINUE
        ENDIF
50      LINE=K
        RETURN
2       BSUM=IV(NCONS)
        DO 70 I=1,NOV
        DO 80 J=1,NCONS
        IF(IB(J).EQ.ID(I)) THEN
        ZPNEW(1,I)=0.0
        GO TO 70
        ENDIF
80      CONTINUE
        ZPNEW(1,I)=TAB(NCONS,I)
70      CONTINUE
        IF(IFLAG.EQ.1) THEN
        DO 100 I=1,NOV
        ZPRO(1,I)=ZPNEW(1,I)
100     CONTINUE
        INPRO(1)=BSUM
        RETURN
        ENDIF
        DO 101 I=1,NOV
        ZPRO(1,I)=ZPNEW(1,I)+Z(1,I)
101     CONTINUE
        INPRO(1)=IN(1)+BSUM
        RETURN
        END
C***************************************************
C*          SUBROUTINE GOMORY                      *
C***************************************************
C
      SUBROUTINE GOMORY(JB,KROWS,FMAX)
      DIMENSION ID(100),INPRO(100),TAB(100,100),Z(100,100)
      DIMENSION IREAL(50),IV(100),IB(100),IN(100),ZPNEW(100,100)
      DIMENSION IPM(1000),TABB(100,100),ZZ(1,100),ZPRO(100,100)
      DIMENSION LEVATT(50)
      REAL IN,IV,INPRO,LEVATT
      COMMON/B1/TAB,IV,ID,IB,LIT
      COMMON/B2/Z,IN,IP,IC,IW
      COMMON/B3/IPVC,XMAX,NOV,INO
      COMMON/B4/NPRNT,NOC,NOPRO
      COMMON/B5/NPRO,IPM,TABB,ZZ
      COMMON/B6/IPEMPT,IPART,IBOUND
      COMMON/B7/ICHECK,INTGP,IREAL,NXREAL
```

```
      COMMON/B12/ITRATN,KINPRO,LEVATT
      COMMON/B13/ICOUNT,IFLAG
      COMMON/B14/ZPRO,INPRO,KPRIOR,ZPNEW,MVARR,MROWSS
      JB=0
      KD=0
C COUNT THE NUMBER OF INTEGER VARIABLES WHICH ARE AVAILABLE.
      DO 17 I=1,NXREAL
      DO 18 J=1,NOC
      IF(IREAL(I).EQ.IB(J)) THEN
      IRHS=IV(J)
      FRACT=IV(J)-IRHS
      IF(FRACT.LE.0.02.OR.FRACT.GE.0.98) THEN
      KD=KD+1
      GO TO 17
      ENDIF
      ENDIF
18    CONTINUE
17    CONTINUE
      IF(KD.LT.NXREAL) GO TO 12
      JB=1
      RETURN
C TO FIND THE MAXIMUM FRACTION FOR THE REQUIRED INTEGER VARIABLES
 12   FMAX=0.0
      DO 10 I=1,NOC
      DO 11 J=1,NXREAL
      IF(IB(I).NE.IREAL(J)) GO TO 11
      IRHS=IV(I)
      FRACT=IV(I)-IRHS
      IF(FRACT.LE..99) GO TO 1
      IV(I)=IV(I)-FRACT+1.
      FRACT=0.0
1     IF(FMAX-FRACT) 5,10,10
5     FMAX=FRACT
      KROWS=I
      GO TO 10
11    CONTINUE
10    CONTINUE
      RETURN
      END
C***********************************************
C*          SUBROUTINE TSORT                   *
C***********************************************
C
        SUBROUTINE TSORT(KBASE,NONBAS,LL,LOW)
        DIMENSION TAB(100,100),Z(100,100),ID(100),IB(100),IV(100)
        DIMENSION IN(100),C(100,100),TABB(100,100),ZZ(1,100)
```

```
         DIMENSION LEVATT(50),IPM(1000),KBASE(50),NONBAS(50),IREAL(50)
         REAL IN,IV,LEVATT
         COMMON/B1/TAB,IV,ID,IB,LIT
         COMMON/B2/Z,IN,IP,IC,IW
         COMMON/B3/IPVC,XMAX,NOV,INO
         COMMON/B4/NPRNT,NOC,NOPRO
         COMMON/B5/NPRO,IPM,TABB,ZZ
         COMMON/B6/IPEMPT,IPART,IBOUND
         COMMON/B7/ICHECK,INTGP,IREAL,NXREAL
         COMMON/B8/C
         COMMON/B10/KING
         COMMON/B11/KAT,KIT
         COMMON/B12/ITRATN,KINPRO,LEVATT
         COMMON/B13/ICOUNT,IFLAG
         COMMON/B15/IPAT,KRAZY
C SAVE THE FOLLOWING VECTORS FOR THE REQUIRED INTEGER VARIABLES
         DO 10 I=1,NXREAL
         KBASE(I)=0
         NONBAS(I)=0
 10      CONTINUE
C THE FOLLOWING SECTION SORT THE BASIC AND NONBASIC VARIABLES
C WHOSE VALUE REQUIRED TO INTEGERED
C TO RECORD THE NONINTEGER BASIC VARIABLES
         LOW=0
         DO 30 I=1,NXREAL
         DO 20 J=1,NOC
         IF(IREAL(I).EQ.IB(J)) THEN
         LOW=LOW+1
         KBASE(LOW)=IB(J)
         GO TO 30
         ENDIF
20       CONTINUE
30       CONTINUE
C
C RECORD THE LIST OF NONBASIC VARIABLES TO BE INTEGERED.
C
         LL=0
         DO 40 I=1,NXREAL
         DO 50 J=1,LOW
         IF(IREAL(I).EQ.KBASE(J)) THEN
         IZZ=0
         GO TO 40
         ELSE
         IZZ=1
         ENDIF
 50      CONTINUE
```

```
      IF(IZZ.EQ.1) THEN
      LL=LL+1
      NONBAS(LL)=IREAL(I)
      ENDIF
40    CONTINUE
      RETURN
      END
C***********************************************
C*            SUBROUTINE BOUND                 *
C***********************************************
C
      SUBROUTINE BOUND
      DIMENSION ATAB(100,100),AZE(100,100),AIV(100),AIN(100)
      DIMENSION IABASE(100),IDVAR(100),ATT(100,100),AZZ(100,100)
      DIMENSION KBB(100),RHV(100),ARHN(100),KDD(100),IN(100)
      DIMENSION TAB(100,100),Z(100,100),IB(100),ID(100),IV(100)
      DIMENSION TABB(100,100),ZZ(1,100),INPRO(100),XDOM(100)
      DIMENSION ZPRO(100,100),XMOD(100),IPM(1000),IREAL(50)
      DIMENSION F(50),FF(50),ZPNEW(100,100),LEVATT(50)
      DIMENSION TUT(100,100),ZUZ(100,100),IUN(100),IUB(100)
      DIMENSION IUD(100),IUV(100),XARRY(200),SARRY(200),ISETT(100)
      DIMENSION LDECS(100),LPDEV(100),LNDEV(100)
      REAL IN,IV,IUV,IUN,INPRO,LEVATT
      INTEGER ZROCUT,SETNOV,SETNOC,SETLL,SUTLL
      COMMON/B1/TAB,IV,ID,IB,LIT
      COMMON/B2/Z,IN,IP,IC,IW
      COMMON/B3/IPVC,XMAX,NOV,INO
      COMMON/B4/NPRNT,NOC,NOPRO
      COMMON/B5/NPRO,IPM,TABB,ZZ
      COMMON/B6/IPEMPT,IPART,IBOUND
      COMMON/B7/ICHECK,INTGP,IREAL,NXREAL
      COMMON/B10/KING
      COMMON/B12/ITRATN,KINPRO,LEVATT
      COMMON/B13/ICOUNT,IFLAG
      COMMON/B14/ZPRO,INPRO,KPRIOR,ZPNEW,MVARR,MROWSS
      COMMON/B15/IPAT,KRAZY
      COMMON/B16/ATAB,AZE,AIV,AIN,IABASE,IDVAR,XDOM,XMOD
      COMMON/B18/IQUE,SETNOV,SETNOC,SETLL
      COMMON/B20/LDECS,LPDEV,LNDEV,LTOT1,LTOT2,LTOT3
      NUMBER=0
      IQUE=0
      JZJJ=0
      NIBB=1
      IF(IPART.EQ.1) THEN
      IFLAG=1
      MVARR=NOV
```

```
      MROWSS=NOC
      KPRIOR=NPRO+1
      MVARR=NOV
      K=2
      INPRO(K)=IN(1)
      DO 19 J=1,100
19    ZPRO(K,J)=0.0
      DO 22 J=1,NOV
      ZPRO(K,J)=Z(1,J)
 22   CONTINUE
      GO TO 21
      ENDIF
      IF(IPEMPT.EQ.1) THEN
      IFLAG=1
      MROWSS=NOC
      MVARR=NOV
      KPRIOR=NPRO+1
      K=1
      DO 101 I=1,NPRO
      K=K+1
      INPRO(K)=IN(I)
      DO 102 J=1,NOV
      ZPRO(K,J)=0.
      ZPRO(K,J)=Z(I,J)
102   CONTINUE
101   CONTINUE
      GO TO 21
      ENDIF
 21   L=K
C TO CONSTRUCT TWO NEW CONSTRAINTES
C
      IBR=1
      IQUE=IQUE+1
      WRITE(6,2222)
      WRITE(10,2222)
2222  FORMAT(//10X,'USING IBR=1')
      SETNOV=NOV
      SETNOC=NOC
      SETLL=L
      CALL BRNCH(IBR,L,FMAX,JB.ZROCUT,IROW)
      IF(ZROCUT.EQ.1) RETURN
      IF(JB.EQ.1) THEN
      ISUMMY=1
      CALL SUMMRY(IB,IN,NOPRO,IV,JZJJ,NOC,ISUMMY)
      IF(NUMBER.EQ.1) GO TO 1
      GO TO 92
```

```
      ENDIF
      GO TO (1,2),ICOUNT
2     INO=1
      NIBR=IBR
      DO 13 INO=1,L
      DO 600 LOT=1,NOC
      IF(IV(LOT).LT.O.) GO TO 601
600   CONTINUE
10    CALL PIVCOL
      IF(XMAX.EQ.O.OR.IPVC.EQ.O) GO TO 5
      CALL PIVROW(IPROW)
      IF(IPART.EQ.1) NPRO=2
      CALL CALC(IPROW)
      IBR=3
      CALL BRNCH(IBR,L,FMAX,JB,ZROCUT,IROW)
      IF(JB.EQ.1) THEN
      ISUMMY=1
      CALL SUMMRY(IB,IN,NOPRO,IV,JZJJ,NOC,ISUMMY)
      CALL PTRG(NOV)
      GO TO 602
      ENDIF
      CALL PTRG(NOV)
      DO 603 I=1,NOC
      IF(I.EQ.IROW) THEN
      IRIGHT=IV(I)
      HTT=IV(I)-IRIGHT
      ENDIF
603   CONTINUE
      IF(INO.EQ.1) GO TO 10
      GO TO 5
601   CALL DUALSX(IPROW,LAB,IBALL)
      IBR=3
      CALL BRNCH(IBR,L,FMAX,JB,ZROCUT,IROW)
      IF(JB.EQ.1) THEN
      ISUMMY=1
      CALL SUMMRY(IB,IN,NOPRO,IV,JZJJ,NOC,ISUMMY)
      CALL PTRG(NOV)
      GO TO 602
      ENDIF
5     IF(IN(2).EQ.O) GO TO 602
13    CONTINUE
602   IBR=NIBR
      IF(IBR.EQ.2) GO TO 1000
      DO 2111 I=1,NOC
      DO 222 J=1,NOV
      ATT(I,J)=TAB(I,J)
```

```
222      CONTINUE
        RHV(I)=IV(I)
        KBB(I)=IB(I)
2111    CONTINUE
        DO 23 I=1,NOPRO
        DO 24 J=1,NOV
        AZZ(I,J)=Z(I,J)
        KDD(J)=ID(J)
24      CONTINUE
        ARHN(I)=IN(I)
23      CONTINUE
        NVARL=NOV
        KNNOC=NOC
        SUTLL=L
C
C SAVE THE VALUE OF PRIORITY OF ONE AND TWO
C
        DO 9 I=1,SUTLL
9       F(I)=IN(I)
C
C PREPARE TO SOLVE THE SECOND PROBLEM
C
        IBR=2
        IQUE=IQUE+1
        WRITE(6,22222)
        WRITE(10,22222)
22222   FORMAT(//10X,'USING IBR=2')
        NOV=SETNOV
        NOC=SETNOC
        LNEW=SETLL
        IF(IQUE.EQ.2) LNEW=SETLL-1
        DO 11 I=1,NOC
        DO 20 J=1,SETNOV
        TAB(I,J)=ATAB(I,J)
20      CONTINUE
        IV(I)=AIV(I)
        IB(I)=IABASE(I)
11      CONTINUE
C SET NOPRO EQUAL TO L
        DO 30 I=1,LNEW
        DO 40 J=1,SETNOV
        Z(I,J)=AZE(I,J)
        ID(J)=IDVAR(J)
 40     CONTINUE
        IN(I)=AIN(I)
 30     CONTINUE
```

```
        IF(IQUE.EQ.2) THEN
        NOPRO=L-1
        CALL PTRG(NOV)
        ICARE=1
        CALL SADD(IFLAG,NCONS,ICARE,LINE)
        K=LINE
        ELSE
        CALL PTRG(NOV)
        ENDIF
C
C STORE XDOM IN XMOD
      DO 50 J=1,SETNOV
      XDOM(J)=0.
      XDOM(J)=XMOD(J)
50    CONTINUE
C
      FMAX=1.-FMAX
      CALL BRNCH(IBR,L,FMAX,JB,ZROCUT,IROW)
      IF(JB.EQ.1.OR.ZROCUT.EQ.1) GO TO 1
      GO TO 2
1000  CONTINUE
      IF(IPART.EQ.1) NPRO=2
      DO 8 I=1,NPRO
8     FF(I)=IN(I)
C
C PREPARE TO SOLVE THE PARTITIONNING PROBLEM
         IF(IPART.EQ.1) THEN
         IF(F(1).EQ.0.AND.F(2).GT.0) THEN
         IF(FF(1).EQ.0.AND.FF(2).EQ.0) GO TO 21
         ENDIF
         IF(FF(1).EQ.0.AND.FF(2).GT.0) THEN
         IF(F(1).EQ.0.AND.F(2).EQ.0) GO TO 57
         ENDIF
         IF(F(1).GT.0.AND.FF(1).EQ.0) GO TO 21
         IF(F(1).EQ.0.AND.FF(1).GT.0) GO TO 57
         NPRO=2
         GO TO 58
         ENDIF
C
COMPARE THE SOLUTION OF THESE TOW PROBLEMS(FOR PREEMPTIVE)
         IF(F(2).GT.0.AND.FF(2).GT.0) GO TO 77
C TERMINATION RULE FOR SUB #1
         IF(F(2).GT.0.AND.FF(2).LE.0) GO TO 56
C TERMINATION RULE FOR SUB #2
         IF(F(2).LE.0.AND.FF(2).GT.0) GO TO 57
         GO TO 58
```

```
C CONTINUE BRANCHING FROM SUB #2
 56       CALL PTRG(NOV)
          GO TO 21
C CONTINUE BRANCHING FROM SUB #1
57        NOV=NVARL
          NOC=KNNOC
          DO 51 I=1,NOC
          DO 52 J=1,NOV
          TAB(I,J)=ATT(I,J)
52        CONTINUE
          IV(I)=RHV(I)
          IB(I)=KBB(I)
51        CONTINUE
C SET NOPRO TO L
          DO 53 I=1,SUTLL
          DO 54 J=1,NOV
          Z(I,J)=AZZ(I,J)
          ID(J)=KDD(J)
54        CONTINUE
          IN(I)=ARHN(I)
53        CONTINUE
          CALL PTRG(NOV)
          IF(ITRATN.GT.100) RETURN
          GO TO 21
58        IAIA=0
          DO 55 KL=1,NPRO
          IF(F(KL).EQ.O.AND.FF(KL).EQ.O) THEN
          IAIA=IAIA+1
          IF(IAIA.EQ.NPRO) GO TO 71
          GO TO 55
          ENDIF
          IF(F(KL).LT.FF(KL)) THEN
          IF(JB.EQ.1.AND.IQUE.EQ.2)  GO TO 68
          IF(IQUE.EQ.2) GO TO 65
          GO TO 68
C SAVE THE TABLEAUE OF SUB #2
65        NIBB=2
          KVV1=NOV
          KCC1=NOC
          DO 66 I=1,KCC1
          DO 67 J=1,KVV1
67        TUT(I,J)=TAB(I,J)
          IUV(I)=IV(I)
          IUB(I)=IB(I)
66        CONTINUE
          DO 69 I=1,L
```

```
          DO 70 J=1,NOV
          ZUZ(I,J)=Z(I,J)
          IUD(J)=ID(J)
70        CONTINUE
69        IUN(I)=IN(I)
C TO USE TABLE OF SUB#1 FOR FURTHER COMPUTATIONS
68        NOV=NVARL
          NOC=KNNOC
          DO 61 I=1,NOC
          DO 62 J=1,NOV
          TAB(I,J)=ATT(I,J)
62        IV(I)=RHV(I)
          IB(I)=KBB(I)
61        CONTINUE
          DO 63 I=1,SUTLL
          DO 64 J=1,NOV
          Z(I,J)=AZZ(I,J)
          ID(J)=KDD(J)
64        CONTINUE
          IN(I)=ARHN(I)
63        CONTINUE
          GO TO 76
          ELSE
          IF(IQUE.EQ.2) GO TO 71
          GO TO 76
71        NIBB=2
          KVV1=NVARL
          KCC1=KNNOC
          DO 72 I=1,KCC1
          DO 73 J=1,KVV1
73        TUT(I,J)=ATT(I,J)
          IUV(I)=RHV(I)
          IUB(I)=KBB(I)
72        CONTINUE
          DO 74 I=1,L
          DO 75 J=1,KVV1
          ZUZ(I,J)=AZZ(I,J)
          IUD(J)=KDD(J)
75        CONTINUE
          IUN(I)=ARHN(I)
74        CONTINUE
          GO TO 76
          ENDIF
55        CONTINUE
76        CALL PTRG(NOV)
          GO TO 21
```

```
77      IF(IQUE.EQ.2) THEN
        WRITE(6,91)
        WRITE(10,91)
91      FORMAT(//10X,'THIS PROBLEM HAS NO INTEGER SOLUTION')
        GO TO 1001
        ENDIF
92      NUMBER=1
        IF(NIBB.EQ.2) THEN
        NOV=KVV1
        NOC=KCC1
        DO 78 I=1,NOC
        DO 79 J=1,NOV
 79     TAB(I,J)=TUT(I,J)
        IV(I)=IUV(I)
 78     IB(I)=IUB(I)
        DO 80 I=1,NOPRO
        DO 81 J=1,NOV
        Z(I,J)=ZUZ(I,J)
        ID(J)=IUD(J)
 81     CONTINUE
        IN(I)=IUN(I)
80      CONTINUE
        CALL PTRG(NOV)
        GO TO 21
        ENDIF
1       IF(NUMBER.EQ.0) GO TO 92
1001    ISUMMY=2
        CALL SUMMRY(IB,IN,NOPRO,IV,JZJJ,NOC,ISUMMY)
        ISUMMY=3
        CALL SUMMRY(IB,IN,NOPRO,IV,JZJJ,NOC,ISUMMY)
        RETURN
        END
C***********************************************
C*          SUBROUTINE BRNCH                    *
C***********************************************
       SUBROUTINE BRNCH(IBR,L,FMAX,JB,ZROCUT,IROW)
       DIMENSION TAB(100,100),Z(100,100),IB(100),ID(100),IN(100)
       DIMENSION IV(100),INPRO(100),ZPRO(100,100),IPM(1000)
       DIMENSION ZPNEW(100,100),XDOM(100),XMOD(100),ZZ(1,100)
       DIMENSION ATAB(100,100),AIV(100),AIN(100),AZE(100,100)
       DIMENSION IABASE(100),IDVAR(100),ATT(100,100),AZZ(100,100)
       DIMENSION KBB(100),RHV(100),ARHN(100),TABB(100,100)
       DIMENSION LEVATT(50),IREAL(50),KBASE(50),NONBAS(50)
       REAL IN,IV,INPRO,LEVATT
       INTEGER ZROCUT,SETNOV,SETNOC,SETLL,SUTLL
       COMMON/B1/TAB,IV,ID,IB,LIT
```

```
      COMMON/B2/Z,IN,IP,IC,IW
      COMMON/B3/IPVC,XMAX,NOV,INO
      COMMON/B4/NPRNT,NOC,NOPRO
      COMMON/B5/NPRO,IPM,TABB,ZZ
      COMMON/B6/IPEMPT,IPART,IBOUND
      COMMON/B7/ICHECK,INTGP,IREAL,NXREAL
      COMMON/B10/KING
      COMMON/B12/ITRATN,KINPRO,LEVATT
      COMMON/B13/ICOUNT,IFLAG
      COMMON/B14/ZPRO,INPRO,KPRIOR,ZPNEW,MVARR,MROWSS
      COMMON/B15/IPAT,KRAZY
      COMMON/B16/ATAB,AZE,AIV,AIN,IABASE,IDVAR,XDOM,XMOD
      COMMON/B18/IQUE,SETNOV,SETNOC,SETLL
      CALL TSORT(KBASE,NONBAS,LL,LOW)
      IFFECT=0
      ZROCUT=0
      JB=0
      KD=0
      DO 10 I=1,LOW
      DO 9 J=1,NOC
      IF(KBASE(I).EQ.IB(J)) THEN
      IRHS=IV(J)
      FRACT=IV(J)-IRHS
      IF(FRACT.LE.0.02.OR.FRACT.GE.0.98)  THEN
      KD=KD+1
      GO TO 10
      ENDIF
      ENDIF
 9    CONTINUE
10    CONTINUE
      KD=KD+LL
      IF(KD.LT.NXREAL) THEN
      GO TO (20,2,400),IBR
400   RETURN
      ELSE
      JB=1
      WRITE(6,40)
40    FORMAT(//15X,'ALL REQUIRED VARIABLES ARE INTEGER')
      RETURN
      ENDIF
20    FMAX=0.0
C
      IF(IFFECT.EQ.1) THEN
      DO 300 I=1,NOC
      DO 301 J=1,LOW
      IF(IB(I).NE.KBASE(J)) GO TO 301
```

```
        IF(IV(I).EQ.O.) GO TO 301
        IRHS=IV(I)
        FRACT=IV(I)-IRHS
        IF(FRACT.LT.O.5) THEN
        FRAKSN=FRACT/IV(I)
        ELSE
        FRAKSN=(1.-FRACT)/IV(I)
        ENDIF
        IF(FMAX-FRAKSN) 302,300,300
302     FMAX=FRAKSN
        IROW=I
        GO TO 300
301     CONTINUE
300     CONTINUE
        ELSE
        DO 30 I=1,NOC
        DO 11  J=1,LOW
        IF(IB(I).NE.KBASE(J)) GO TO 11
        IRHS=IV(I)
        FRACT=IV(I)-IRHS
        IF(FRACT.LE.O.99) GO TO 111
C
        IV(I)=IV(I)-FRACT+1.
        FRACT=O.O
111     IF(FMAX-FRACT) 5,30,30
5       FMAX=FRACT
        IROW=I
        GO TO 30
11      CONTINUE
30      CONTINUE
        ENDIF
C
        IF(FMAX.LE.O.01) THEN
        ICOUNT=1
        RETURN
        ENDIF
C
C TO DEVELOPE TWO NEW CONSTRAINTS
C
        IPROW=IROW
        DO 25 J=1,NOV
        IF(ID(J).EQ.IB(IPROW)) GO TO 21
        XDOM(J)=TAB(IPROW,J)
        IF(XDOM(J).EQ.O) THEN
        XMOD(J)=XDOM(J)
        ELSE
```

```
      XMOD(J)=-XDOM(J)
      ENDIF
      GO TO 25
21    XDOM(J)=0.0
      XMOD(J)=0.0
 25   CONTINUE
      WRITE(6,110) (XDOM(J),J=1,NOV)
      WRITE(10,110) (XDOM(J),J=1,NOV)
      WRITE(6,110) (XMOD(J),J=1,NOV)
      WRITE(10,110) (XMOD(J),J=1,NOV)
110   FORMAT(//2X,10(F8.5,2X))
C CHECK FOR ZERO-CUTTING PLANES
      ZROCUT=0
      ICOTT=0
      DO 26 ILL=1,NOV
      IF(XDOM(ILL)) 26,27,26
27    ICOTT=ICOTT+1
26    CONTINUE
      IF(ICOTT.EQ.NOV) THEN
      ZROCUT=1
      WRITE(6,28)
      WRITE(10,28)
28    FORMAT(//10X,'A ZERO-CUT IS DETECTED')
      IF(IPART.EQ.1) THEN
      WRITE(6,29)
      WRITE(10,29)
29    FORMAT(//10X,'TRY ANOTHER PROCEDURE'//10X,'USE PREEMPTIVE
     + GP PROCEDURE')
      ENDIF
C PURPOSE TO CONSIDER THE PREEPTIVE GOAL PROGRAMMING PROCEDURE
      IF(IPEMPT.EQ.1) THEN
      WRITE(6,24)
      WRITE(10,24)
24    FORMAT(//10X,'TRY ANOTHER PROCEDURE'//10X,'USE PARTITIONING
     + GP PROCEDURE')
      ENDIF
      RETURN
      ENDIF
C SAVE THE TABLEAU,BASIS,RHS, DECISION VARIABLES,AND MATRIX OF PRIORITY
C WEIGHTES
      DO 70 I=1,NOC
      DO 80 J=1,NOV
      ATAB(I,J)=TAB(I,J)
 80   CONTINUE
      AIV(I)=IV(I)
      IABASE(I)=IB(I)
```

```
70      CONTINUE
        DO 81 I=1,NOPRO
        DO 82 J=1,NOV
        AZE(I,J)=Z(I,J)
        IDVAR(J)=ID(J)
82      CONTINUE
        AIN(I)=IN(I)
81      CONTINUE
C
C ADD THE CUTTING PLANES INTO THE TABLE
C
2       IF(IPEMPT.EQ.1) THEN
        NVAR1=NOV+1
        NVAR=NOV+2
        NCONS=NOC+1
        ID(NVAR1)=NVAR1
        ID(NVAR)=NVAR
        IB(NCONS)=NVAR
        DO 90 I=1,NOC
        DO 90 J=NVAR1,NVAR
90      TAB(I,J)=0.
        DO 92 I=2,L
        DO 92 J=NVAR1,NVAR
92      Z(I,J)=0.
        TAB(NCONS,NVAR1)=-1.
        TAB(NCONS,NVAR)=1.
        DO 93 J=1,NOV
 93     TAB(NCONS,J)=XDOM(J)
        IV(NCONS)=FMAX
        ENDIF
C
C  ADD A NEW PRIORITY TO THE PRIORITY MATRIX
C
        IF(IPEMPT.EQ.1) THEN
        DO 94 I=1,NVAR
        DO 95 J=1,NCONS
        IF(IB(J).EQ.I) THEN
        ZPRO(1,I)=0.0
        GO TO 94
        ELSE
        ZPRO(1,I)=TAB(NCONS,I)
        ENDIF
95      CONTINUE
94      CONTINUE
        DO 96 I=2,L
        DO 96 J=NVAR1,NVAR
```

```
 96     ZPRO(I,J)=0.
        ZPRO(1,NVAR1)=-1.
        ZPRO(1,NVAR)=0.
        GO TO 194
        ENDIF
C
C FOR PARTITIONNING PROCEDURE
C
        IF(IPART.EQ.1) THEN
        LOCT=ID(NOV)
        NVAR1=LOCT+1
        NVAR2=LOCT+2
        NCONS=NOC+1
        NVV1=NOV+1
        NVV2=NOV+2
        ID(NVV1)=NVAR1
        ID(NVV2)=NVAR2
        IB(NCONS)=NVAR2
        DO 101 I=1,NOC
        DO 101 J=NVV1,NVV2
101     TAB(I,J)=0.
        DO 102 J=NVV1,NVV2
102     Z(2,J)=0.
        TAB(NCONS,NVV1)=-1.
        TAB(NCONS,NVV2)=1.
        DO 103 J=1,NOV
103     TAB(NCONS,J)=XDOM(J)
        IV(NCONS)=FMAX
        NVAR=NVV2
        ENDIF
        IF(IPART.EQ.1) THEN
        DO 104 I=1,NVV2
        DO 105 J=1,NCONS
        IF(IB(J).EQ.ID(I)) THEN
        ZPRO(1,I)=0.
        GO TO 104
        ELSE
        ZPRO(1,I)=TAB(NCONS,I)
        ENDIF
105      CONTINUE
104     CONTINUE
        ZPRO(1,NVV1)=-1.
        ZPRO(1,NVV2)=0.
        ZPRO(2,NVV1)=0.
        ZPRO(2,NVV2)=0.
        ENDIF
```

```
C
194     CONTINUE
C UPDATE NUMBER OF CONSTRAINTES ,VARIABLES,AND PRIORITIES
C
        NOV=NVAR
        NOC=NCONS
        NPRO=KPRIOR
        NOPRO=NPRO
        BSUM=IV(NCONS)
        INPRO(1)=BSUM
        DO 107 J=1,NOV
        Z(1,J)=ZPRO(1,J)
        IN(1)=INPRO(1)
107     CONTINUE
C
C       IF(IFLAG.EQ.1) THEN
        IF(IQUE.LE.2) THEN
        DO 108 I=2,L
        DO 109 J=1,NOV
        Z(I,J)=ZPRO(I,J)
109     CONTINUE
        IN(I)=INPRO(I)
108     CONTINUE
        IN(1)=IV(NCONS)
        ENDIF
        ICOUNT=2
        IF(IPART.EQ.1) THEN
        NOPRO=2
        CALL PTRG(NOV)
        RETURN
        ENDIF
        IF(IPEMPT.EQ.1) THEN
        CALL PTRG(NOV)
        ENDIF
        RETURN
        END
C***********************************************
C*         SUBROUTINE SUMMRY                   *
C***********************************************
        SUBROUTINE SUMMRY(IB,IN,NOPRO,IV,JZJJ,NOC,ISUMMY)
C
        DIMENSION IB(100),IV(100),SARRY(200),XARRY(200),ISETT(100)
        DIMENSION LDECS(100),LPDEV(100),LNDEV(100)
        DIMENSION IN(100),BINE(100),KBUT(50)
        REAL IV,IN
        INTEGER XARRY
```

```
      COMMON/B20/LDECS,LPDEV,LNDEV,LTOT1,LTOT2,LTOT3
C
      GO TO (83,82,999),ISUMMY
C
C TO STORE THE INTEGER SOLUTIONS
C
 83   JZJJ=JZJJ+1
      IF(JZJJ.EQ.1) THEN
      DO 84 J=1,NOC
      SARRY(J)=IV(J)
 84   XARRY(J)=IB(J)
      ISETT(JZJJ)=NOC
      DO 60 L=1,NOPRO
60    BINE(L)=IN(L)
      KBUT(JZJJ)=NOPRO
      ENDIF
      IF(JZJJ.GE.2) THEN
      MAN1=1
      KBBC=JZJJ-1
      DO 70 I=1,KBBC
70    MAN1=MAN1+ISETT(I)
      MAN2=MAN1+NOC-1
      K=0
      DO 91 J=MAN1,MAN2
      K=K+1
      SARRY(J)=IV(K)
91    XARRY(J)=IB(K)
      ISETT(JZJJ)=NOC
      MAN3=1
      LAM=JZJJ-1
      DO 65 L=1,LAM
65    MAN3=MAN3+KBUT(L)
      MAN4=MAN3+NOPRO-1
      LK=0
      DO 66 J=MAN3,MAN4
      LK=LK+1
66    BINE(J)=IN(LK)
      KBUT(JZJJ)=NOPRO
      ENDIF
      RETURN
C
 82   CONTINUE
C TO SUMMARIZE THE INTEGER SOLUTIONS
C
      IF(JZJJ.EQ.0) THEN
      WRITE(6,90)
```

```
      WRITE(10,90)
      WRITE(6,92)
      WRITE(10,92)
92    FORMAT(//20X,'THIS PROBLEM HAS NO INTEGER SOLUTION')
      RETURN
      ENDIF
      WRITE(6,90)
      WRITE(10,90)
90    FORMAT(//20X,'SUMMARY OF INTEGER SOLUTION')
      JET1=1
      JET3=1
      DO 85 I=1,JZJJ
      JET2=ISETT(I)
      JET=JET1+JET2-1
      WRITE(6,87)
      WRITE(10,87)
87    FORMAT(//20X,'VARIABLES',11X,'VALUE')
      DO 86 II=JET1,JET
      WRITE(6,89) XARRY(II),SARRY(II)
      WRITE(10,89) XARRY(II),SARRY(II)
89    FORMAT(//20X,I8,10X,F16.6)
86    CONTINUE
      JET1=JET+1
      JET4=KBUT(I)
      JET5=JET3+JET4-1
      WRITE(6,67)
      WRITE(10,67)
67    FORMAT(//20X,'PRIORITY',11X,'VALUE')
      IOT=0
      DO 68 JJ=JET3,JET5
      IOT=IOT+1
      WRITE(6,69) IOT,BINE(JJ)
      WRITE(10,69) IOT,BINE(JJ)
69    FORMAT(//20X,I8,10X,F16.6)
68    CONTINUE
      JET3=JET5+1
85    CONTINUE
      RETURN
C
999   WRITE(6,109)
      WRITE(10,109)
      WRITE(6,101)
      WRITE(10,101)
101   FORMAT(//10X,'OPTIMAL SOLUTION FOR ORIGINAL DECIS. VARIABLES')
      DO 102 J=1,LTOT1
      IBIB=0
```

```
      DO 103 I=1,NOC
      IF(IB(I).EQ.LDECS(J)) THEN
      WRITE(6,104) J,IV(I)
      WRITE(10,104) J,IV(I)
104   FORMAT(//10X,'X(',I2,1X,')=',6X,F16.6)
      IBIB=1
      GO TO 102
      ENDIF
103   CONTINUE
      IF(IBIB.EQ.0) THEN
      WRITE(6,105) J
      WRITE(10,105) J
105   FORMAT(//10X,'X(',I2,1X,')=',16X,'0.0000')
      ENDIF
102   CONTINUE
      WRITE(6,109)
      WRITE(10,109)
109   FORMAT(//5X,'************************************************')
      WRITE(6,15550)
      WRITE(10,15550)
15550 FORMAT(//20X,'OVERACHIVEMENTS')
      DO 106 J=1,LTOT2
      IAIA=0
      DO 107 I=1,NOC
      IF(IB(I).EQ.LPDEV(J)) THEN
      WRITE(6,108) J,IV(I)
      WRITE(10,108) J,IV(I)
108   FORMAT(//10X,'D+(',I2,1X,')=',6X,F16.6)
      IAIA=1
      GO TO 106
      ENDIF
107   CONTINUE
      IF(IAIA.EQ.0) THEN
      WRITE(6,110) J
      WRITE(10,110) J
110   FORMAT(//10X,'D+(',I2,1X,')=',16X,'0.0000')
      ENDIF
106   CONTINUE
C
      WRITE(6,109)
      WRITE(10,109)
      WRITE(6,111)
      WRITE(10,111)
111   FORMAT(//20X,'UNDERACHIVEMENT')
      DO 112 J=1,LTOT3
      ICIC=0
```

```
        DO 113 I=1,NOC
        IF(IB(I).EQ.LNDEV(J)) THEN
       WRITE(6,115) J,IV(I)
       WRITE(10,115) J,IV(I)
115     FORMAT(//10X,'D-(',I2,1X,')=',6X,F16.6)
       ICIC=1
       GO TO 112
       ENDIF
113    CONTINUE
       IF(ICIC.EQ.0) THEN
       WRITE(6,114) J
       WRITE(10,114) J
114    FORMAT(//10X,'D-(',I2,1X,')=',16X,'0.0000')
       ENDIF
112    CONTINUE
       WRITE(6,109)
       WRITE(10,109)
       RETURN
       END
C*************************************************
C*      SUBROUTINE MEMOH                        *
C*************************************************
C
        SUBROUTINE MEMOH(JOYL,ISEN)
C
        DIMENSION IREAL(50)
        COMMON/B6/IPEMPT,IPART,IBOUND
        COMMON/B7/ICHECK,INTGP,IREAL,NXREAL
C
        GO TO (1,2,4),JOYL
C DISPLAY OF MENU 1
C
1       WRITE(6,7)
        WRITE(10,7)
7       FORMAT(15X,'DISPLAY OF MENU 1 ')
        WRITE(6,10)
        WRITE(10,10)
10      FORMAT(/5X,'CONTINOUSE SOLUTION BY PREGP PROCEDURE')
        WRITE(6,11)
        WRITE(10,11)
11      FORMAT(/5X,'***ENTER 1 ***')
        WRITE(6,20)
        WRITE(10,20)
20      FORMAT(/5X,'CONTINOUSE SOLUTION BY PARGP PROCEDURE')
        WRITE(6,21)
        WRITE(10,21)
```

```
21      FORMAT(/5X,'*** ENTER 2 ***')
        IF(JOYL.EQ.1) GO TO 3
C       TO DISPLAY OF MENU 3
4       WRITE(6,8)
        WRITE(10,8)
8       FORMAT(/15X,'DISPLAY OF MENU 3 ')
        WRITE(6,40)
        WRITE(10,40)
40      FORMAT(/5X,'INTEGER SOLUTION BY PREGP USING CUTTING PLANE')
        WRITE(6,41)
        WRITE(10,41)
41      FORMAT(/5X,'*** ENTER 3 ***')
        WRITE(6,50)
        WRITE(10,50)
50      FORMAT(/5X,'INTEGER SOLUTION BY PREGP USING B & B ')
        WRITE(6,51)
        WRITE(10,51)
51      FORMAT(/5X,'*** ENTER 4 ***')
        WRITE(6,60)
        WRITE(10,60)
60      FORMAT(/5X,'INTEGER SOLUTION BY PARGP USING CUTTING PLANE')
        WRITE(6, 61)
        WRITE(10,61)
61      FORMAT(/5X,'*** ENTER 5 ***')
        WRITE(6,68)
        WRITE(10,68)
68      FORMAT(/5X,'FIND INTEGER SOLUTION BY PARGP USING B & B')
        WRITE(6,69)
        WRITE(10,69)
69      FORMAT(/5X,'*** ENTER 6 ***')
C
        WRITE(6,65)
        WRITE(10,65)
65      FORMAT(/5X,'TO KEEP THE CONTINOUSE SOLUTION')
        WRITE(6,66)
        WRITE(10,66)
66      FORMAT(/5X,'*** ENTER 7 ***')
3       WRITE(6,9)
        WRITE(10,9)
9       FORMAT(/15X,'*** CHOOSE THE OPTION ***')
        READ (5,*) MOO
        IF(MOO.EQ.7) RETURN
         IF(MOO.EQ.1) THEN
         WRITE(6,70)
         WRITE(10,70)
         IPART=0
```

```
IPEMPT=1
INTGP=C
IBOUND=O
GO TO 999
ENDIF
IF(MOO.EQ.2) THEN
WRITE(6.71)
WRITE(10,71)
IPART=1
IPEMPT=O
INTGP=O
IBOUND=O
GO TO 999
ENDIF
IF(JOYL.EQ.1) RETURN
IF(MOO.EQ.3) THEN
WRITE(6,9O)
WRITE(10,9O)
IPEMPT=1
IPART=O
INTGP=1
IBOUND=1
GO TO 999
ENDIF
IF(MOO.EQ.4) THEN
WRITE(6,91)
WRITE(10,91)
IPEMPT=1
IPART=C
INTGP=1
IBOUND=2
GO TO 999
ENDIF
IF(MOO.EQ.5) THEN
WRITE(6,92)
WRITE(10,92)
IPART=1
IPEMPT=O
INTGP=1
IBOUND=1
GO TO 999
ENDIF
IF(MOO.EQ.6) THEN
WRITE(6.93)
WRITE(10,93)
IPART=1
```

```
      IPEMPT=0
      INTGP=1
      IBOUND=2
      ENDIF
70    FORMAT(/5X,'A CONTINOUSE SOLUTION BY PREGP IS REQUESTED')
71    FORMAT(//5X,'A CONTINOUS SOLUTION BY PARGP IS REQUESTED')
90    FORMAT(//5X,'LIPREGP AND GOMORY 'S CP METHOD IS SELECTED')
91    FORMAT(//5X,'LIPREGP AND BRACH AND BOUND METHOD IS SELECTED')
92    FORMAT(//5X,'LIPARGP AND GOMORY 'S CP METHOD IS SELECTED')
93    FORMAT(//5X,'LIPARGP AND BRANCH AND BOUND METHOD IS SELECTED')
999   IF(INTGP.EQ.1) THEN
      INTUR=2
      CALL INTERS(INTUR)
      ENDIF
      RETURN
C
C DISPLAY OF MENU 2
C
2     WRITE(6,94)
      WRITE(10,94)
94    FORMAT(15X,'*** DISPLAY OF MENU 2 ***')
      WRITE(6,700)
      WRITE(10,700)
700   FORMAT(25X,'***  MENUE FOR SENSITIVITY ANALYSIS ***')
299   FORMAT(5X,'TO DO NO CHANGES  ENTER 5')
      WRITE(6,100)
      WRITE(10,100)
100   FORMAT(5X,'CHANGE THE RHS VALUES')
      WRITE(6,101)
      WRITE(10,101)
101   FORMAT(5X,'** ENTER 1 **')
      WRITE(6,102)
      WRITE(10,102)
102   FORMAT(5X,'TO ADD A NEW DECISION VARIABLE')
      WRITE(6,103)
      WRITE(10,103)
103   FORMAT(5X,'*** ENTER  2 ***')
      WRITE(6,104)
      WRITE(10,104)
104   FORMAT(5X,'TO ADD A NEW OBJECTIVE FUNCTION')
      WRITE(6,105)
      WRITE(10,105)
105   FORMAT(5X,'** ENTER 3 **')
      WRITE(6,106)
      WRITE(10,106)
106   FORMAT(5X,'TO CHANGE THE COEFFICIENT ASSOCIATED WITH THE
```

```
     +  ITH ROW AND JTH NONBASIC COLUMN')
        WRITE(6,107)
107     FORMAT(5X,'*** ENTER 4 ***')
        WRITE(6,299)
        WRITE(10,299)
        WRITE(6,9)
        WRITE(10,9)
        READ(5,*) ISEN
C
        RETURN
        END
C*****************************************************
C*             SUBROUTINE BINVRS                     *
C*****************************************************
C
        SUBROUTINE BINVRS(SENS,ISKIL)
        DIMENSION IVZAR(100),STOF(100,100),ISDD(100),SZV(100,100)
        DIMENSION ID(100),IBOR(100),TAB(100,100),SENS(100,100)
        DIMENSION IV(100),IB(100),SSIN(100)
        COMMON/B1/TAB,IV,ID,IB,LIT
        COMMON/B3/IPVC,XMAX,NOV,INO
        COMMON/B4/NPRNT,NOC,NOPRO
        COMMON/SS1/IBOR,IVZAR,STOF,ISDD,SZV,SSIN
        REAL IV
C PURPOSE TO DETERMINE THE MATRIX OF B INVERSE
        K=0
        DO 10 L=1,NOC
        DO 20 J=1,NOV
        IF(ID(J).EQ.IBOR(L)) THEN
        K=K+1
        DO 30 LI=1,NOC
30      SENS(LI,K)=TAB(LI,J)
        GO TO 10
        ENDIF
20      CONTINUE
10      CONTINUE
        ISKIL=K
        DO 40 I=1,ISKIL
        WRITE(6,50) (SENS(I,J),J=1,ISKIL)
        WRITE(10,50) (SENS(I,J),J=1,ISKIL)
50      FORMAT(5X,10(F6.2,2X))
40      CONTINUE
        RETURN
        END
C**************************************************
C*            SUBROUTINE SENSTY                   *
```

```
C*************************************************
        SUBROUTINE SENSTY(ISEN,IAM)
        DIMENSION STABB(100,100),SIV(100),ISIB(100),SZZ(100,100)
        DIMENSION ISID(100),SIN(100),LNBVG(100),BMOD(100),SENS(100,100)
        DIMENSION IBOR(100),IVZAR(100),STOF(100,100),SZV(100,100)
        DIMENSION ISDD(100),SSIN(100),SCC(100,100),BBC(100),TAA(100,100)
        DIMENSION IPM(1000),ISIN(100),TBC(100,100)
        DIMENSION TAB(100,100),IV(100),ID(100),IB(100),Z(100,100)
        DIMENSION IN(100),C(100,100),TABB(100,100),ZZ(1,100)
        COMMON/B1/TAB,IV,ID,IB,LIT
        COMMON/B2/Z,IN,IP,IC,IW
        COMMON/B3/IPVC,XMAX,NOV,INO
        COMMON/B4/NPRNT,NOC,NOPRO
        COMMON/B5/NPRO,IPM,TABB,ZZ
        COMMON/B8/C
        COMMON/B9/NOC1,NOC2
        COMMON/SS1/IBOR,IVZAR,STOF,ISDD,SZV,SSIN
        REAL IN,IV
C TO SAVE THE OPTIMAL TABLEAU FOR SENSITIVITY
C
52      FORMAT(2X,'CORRECT',5X,'ENTER',2X,'1:YES',2X,'2:NO')
53      FORMAT(2X,'REENTER AGAIN')
        GO TO (1,2,901),IAM
 1      DO 61 I=1,NOC
        DO 62 J=1,NOV
62      STABB(I,J)=TAB(I,J)
        SIV(I)=IV(I)
        ISIN(I)=IB(I)
61      CONTINUE
        DO 63 I=1,NOPRO
        DO 64 J=1,NOV
        SZZ(I,J)=Z(I,J)
64      ISID(J)=ID(J)
63      SIN(I)=IN(I)
        NCO=NOC
        NVO=NOV
        NPO=NOPRO
901     IF(ISEN.EQ.1) THEN
        WRITE(6,108)
        WRITE(10,108)
108     FORMAT(5X,'TO CHANGE THE RHS VALUES')
        WRITE(6,109)
       WRITE(10,109)
109     FORMAT(5X,'ENTER THE NUMBER OF  CHANGES IN RHS')
        READ(5,*) ICHANG
        WRITE(6,51) ICHANG
```

```
        WRITE(10,51) ICHANG
51      FORMAT('# OF CHANGES =',2X,I4)
C
        WRITE(6,110)
        WRITE(10,110)
110     FORMAT(5X,'ENTER THE ROW NUMBER AND ITS VALUE RESPEVCTIVELY')
        DO 120 I=1,ICHANG
55      READ(5,*) IRUD,PROD
        WRITE(6,54) IRUD,PROD
        WRITE(10,54) IRUD,PROD
54      FORMAT(2X,'ROW=',2X,I4,'RHS=',2X,F8.4)
        WRITE(6,52)
        WRITE(10,52)
        READ(5,*) ICORR
        IF(ICORR.EQ.2) THEN
        WRITE(6,53)
        WRITE(10,53)
        GO TO 55
        ENDIF
120     IVZAR(IRUD)=PROD
        CALL BINVRS(SENS,ISKIL)
        DO 121 I=1,ISKIL
        SUM=0
        DO 122 J=1,ISKIL
122     SUM=SUM+SENS(I,J)*IVZAR(J)
        IV(I)=SUM
121     CONTINUE
C
C EVALUATE THE VALUE OF EACH PRIORITY LEVEL
C
        DO 40 I=1,NOPRO
        SSM=0
        DO 41 KK=1,NOC
        LOK=IB(KK)
41      SSM=SSM+C(I,LOK)*IV(KK)
40      IN(I)=SSM
        CALL PTRG(NOV)
        DO 50 I=1,ISKIL
        IF(IV(I).LT.0) THEN
        NOC2=NOC
        CALL DUALSX(IPROW,LAB,IBALL)
        GO TO 999
        ENDIF
50      CONTINUE
999     RETURN
        ENDIF
```

```
C
C  FOR ADDITION OF NEW DECISION VARIABLES
C
       IF(ISEN.EQ.2) THEN
       WRITE(6,320)
       WRITE(10,320)
320    FORMAT(5X,'A VARIABLE NEED TO BE ADDED')
322    FORMAT(5X,'ENTER ROW #,VAR # START FROM 1 AND THEN ITS VAL')
       WRITE(6,323)
       WRITE(10,323)
323    FORMAT(5X,'ENTER THE NUMBER OF NEW VARIABLES')
       READ(5,*) NNVR
       WRITE(6,56) NNVR
       WRITE(10,56) NNVR
56     FORMAT('# OF NEW VARIABLES=',2X,I4)
       NOVB=NOV
       DO 324 I=1,NNVR
       NOVB1=NOVB+I
       ID(NOVB1)=NOVB1
       DO 998 IPPL=1,NOPRO
998    C(IPPL,NOVB1)=0.
       DO 700 J=1,NOC
700    STOF(J,NOVB1)=0.0
324    CONTINUE
       WRITE(6,321)
       WRITE(10,321)
321    FORMAT(5X,'ENTER THE NUMBER OF NONZERO ELEMENTS')
       READ(5,*) NOCZRO
       WRITE(6,57) NOCZRO
       WRITE(10,57) NOCZRO
57     FORMAT(2X,'# OF NONZERO ELEMENTS =',2X,I4)
       WRITE(6,322)
       WRITE(10,322)
       DO 326 J=1,NOCZRO
59     READ(5,*) IRUW,IRR,VALY1
       WRITE(6,58) IRUW,IRR,VALY1
       WRITE(10,58) IRUW,IRR,VALY1
58     FORMAT(2X,'ROW=',2X,I3,2X,'VARIABLE#',2X,I3,2X,'VALUE',2X,F8.4)
       WRITE(6,52)
       WRITE(10,52)
       READ(5,*) ICORR
       IF(ICORR.EQ.2) THEN
       WRITE(6,53)
       WRITE(10,53)
       GO TO 59
       ENDIF
```

```
      NOVB1=NOV+IRR
      STOF(IRUW,NOVB1)=VALY1
326   CONTINUE
      CALL BINVRS(SENS,ISKIL)
      DO 329 IZB=1,NNVR
      NO=NOV+IZB
      DO 327 LIB=1,ISKIL
      SUM=0
      DO 328 LIC=1,ISKIL
328   SUM=SUM+SENS(LIB,LIC)*STOF(LIC,NO)
      BMOD(LIB)=SUM
327   CONTINUE
      DO 330 I=1,ISKIL
330   TAB(I,NO)=BMOD(I)
329   CONTINUE
      DO 333 IZB=1,NNVR
      NO=NOV+IZB
      DO 331 I=1,NOPRO
      SUM=0
      DO 332 J=1,NOC
      MP=IB(J)
      SUM=SUM+C(I,MP)*TAB(J,NO)
332   CONTINUE
      Z(I,NO)=SUM-C(I,NO)
331   CONTINUE
333   CONTINUE
      NOV=NO
      CALL PTRG(NOV)
      DO 335 INO=1,NOPRO
10    CALL PIVCOL
      IF(XMAX.EQ.0.OR.IPVC.EQ.0) GO TO 335
      CALL PIVROW(IPROW)
      CALL CALC(IPROW)
      CALL PTRG(NOV)
      IF(INO.EQ.1) GO TO 10
      IF(IN(INO).EQ.0) GO TO 335
335   CONTINUE
      RETURN
      ENDIF
C
CC ADD A NEW OBJECTIVE FUNCTION
C
      IF(ISEN.EQ.3) THEN
      WRITE(6,400)
      WRITE(10,400)
400   FORMAT(5X,'YOU ARE IN THE PROCESS OF ADDING A OBJECTIVE FUN')
```

```
65      WRITE(6,401)
        WRITE(10,401)
401     FORMAT(5X,'ENTER # OF NEW CONSTRAINTS AND # OF NEW VARIABLES')
        READ(5,*) KKNOC,KKNOV
        WRITE(6,402) KKNOC,KKNOV
        WRITE(10,402) KKNOC,KKNOV
402     FORMAT(5X,'NO.OF NEW CONST=',2X,I3/5X,'NO. OF NEW VAR=',I3)
        WRITE(6,52)
        WRITE(10,52)
        READ(5,*) ICORR
        IF(ICORR.EQ.2) THEN
        WRITE(6,53)
        WRITE(10,53)
        GO TO 65
        ENDIF
        NOC1=NOC
        NOVP=NOV+1
        NOCP=NOC
        NOC=NOC+KKNOC
        NOV=NOV+KKNOV
        INOV1=NOVP
        DO 403 I=1,KKNOV
        ID(INOV1)=INOV1
        INOV1=INOV1+1
403     CONTINUE
        DO 404 I=NOVP,NOV
        DO 404 J=1,NOCP
404     TAB(J,I)=0.0
        NOCC=NOC1+1
        DO 900 IBOL=NOCC,NOC
        DO 900 IBOK=1,NOV
900     TAB(IBOL,IBOK)=0.
        WRITE(6,406)
        WRITE(10,406)
406     FORMAT(5X,'ENTER NUMBER OF NONZERO ELEMENTS IN THE NEW
     +  CONSTRAINTS')
        READ(5,*) NZRON
        WRITE(6,66) NZRON
        WRITE(10,66) NZRON
66      FORMAT(2X,'# OF NONZERO ELEMENTS =',2X,I5)
        WRITE(6,67)
        WRITE(10,67)
67      FORMAT(2X,'ENTER ROW I,COLUMN J AND ITS VALUE')
        DO 407 I=1,NZRON
        WRITE(6,68) L,M,VALUE
        WRITE(10,68) L,M,VALUE
```

```
68       FORMAT(2X,'ROW I =',2X,I3,'COLUMN J =',2X,I3,'VALUE=',2X,F8.4)
         WRITE(6,52)
         WRITE(10,52)
         READ(5,*) ICORR
         IF(ICORR.EQ.2) THEN
         WRITE(6,53)
         WRITE(10,53)
         GO TO 69
         ENDIF
69       READ(5,*) L,M,VALUE
         TAB(L,M)=VALUE
407      CONTINUE
         WRITE(6,408)
         WRITE(10,408)
408      FORMAT(5X,'ENTER BASIS AND RHS VALUE OF EACH  CONSTRAINT')
         DO 409 I=1,KKNOC
71       READ(5,*) L,IBB,VIV
         WRITE(6,70) L,IBB,VIV
         WRITE(10,70) L,IBB,VIV
70       FORMAT(2X,'ROW=',2X,I3,'BASIS=',2X,I3,'RHS=',2X,F8.4)
         WRITE(6,52)
         WRITE(10,52)
         READ(5,*) ICORR
         IF(ICORR.EQ.2) THEN
         WRITE(6,53)
         WRITE(10,53)
         GO TO 71
         ENDIF
         IB(L)=IBB
         IV(L)=VIV
409      CONTINUE
         WRITE(6,410)
         WRITE(10,401)
410      FORMAT(5X,'ENTER THE PRIORITY LEVEL OF THIS NEW GOAL AND '
     +/5X,'THE NUMBER OF NONZERO ELEMENTS IN THE NEW PRIORITY LEVEL')
         READ(5,*) LPGOAL,NONZO
         WRITE(6,72) LPGOAL,NONZO
         WRITE(10,72) LPGOAL,NONZO
72       FORMAT(2X,'PRIORITY LEVEL',2X,I3,2X,'#OF NONZERO ELEMENTS
     +=',2X,I3)
         WRITE(6,412)
         WRITE(10,412)
412      FORMAT(/5X,'ENTER THE PRIORITY WEIGHTS')
C
C TO ARRANGE THE MATRIX OF PRIORITY WEIGHTS
C
```

```
      IF(LPGOAL.EQ.1) THEN
      DO 415 I=1,NOPRO
      K=I+1
      DO 416 J=1,NOV
      SCC(1,J)=0.0
      IF(J.GE.NOVP) THEN
      SCC(K,J)=0.
      GO TO 416
      ENDIF
      SCC(K,J)=C(I,J)
416   CONTINUE
415   CONTINUE
      GO TO 426
      ENDIF
      NOPP=NOPRO+1
      IF(LPGOAL.EQ.NOPP) THEN
      DO 417 I=1,NOPRO
      DO 417 J=1,NOV
      IF(J.GE.NOVP) THEN
      SCC(I,J)=0.
      GO TO 417
      ENDIF
      SCC(I,J)=C(I,J)
417   CONTINUE
      DO 418 J=1,NOV
418   SCC(NOPP,J)=0.0
      GO TO 426
      ENDIF
      IF(LPGOAL.NE.1.AND.LPGOAL.NE.NOPRO) THEN
      ILP1=LPGOAL-1
      ILP2=LPGOAL+1
      NPPP=NOPRO+1
      DO 420 I=1,NPPP
      IF(I.LE.ILP1) THEN
      DO 421 J=1,NOV
      IF(J.GE.NOVP) THEN
      SCC(I,J)=0.
      GO TO 421
      ENDIF
      SCC(I,J)=C(I,J)
421   CONTINUE
      GO TO 420
      ENDIF
      IF(I.EQ.LPGOAL) THEN
      DO 422 J=1,NOV
      SCC(LPGOAL,J)=0.0
```

```
422     CONTINUE
        GO TO 420
        ENDIF
        K=I-1
        DO 423 J=1,NOV
        IF(J.GE.NOVP) THEN
        SCC(I,J)=C.
        GO TO 423
        ENDIF
        SCC(I,J)=C(K,J)
 423    CONTINUE
 420    CONTINUE
        ENDIF
426     WRITE(6,73)
        WRITE(10,73)
73      FORMAT(2X,'ENTER VAR #, AND PRIORITY WEIGHT OF THIS VARIABLE')
        DO 427 I=1,NONZO
75      READ(5,*) IVAR,IVAL
        WRITE(6,74) IVAR,IVAL
        WRITE(10,74) IVAR,IVAL
74      FORMAT(2X,'VARIABLE #',2X,I3,2X,'PRIORITY WEIGHT=',2X,I4)
        WRITE(6,52)
        WRITE(10,52)
        READ(5,*) ICORR
        IF(ICORR.EQ.2) THEN
        WRITE(6,53)
        WRITE(10,53)
        GO TO 75
        ENDIF
        SCC(LPGOAL,IVAR)=IVAL
427     CONTINUE
        NOC2=NOC
        NOC3=NOC2-NOC1
        NNOC=NOC1+1
        DO 500 NBC=1,NOC3
        DO 501 J=1,NOV
        DO 502 I=1,NOC1
        IF(IB(I).EQ.ID(J)) THEN
        L=I
        IF(TAB(NNOC,J).NE.O.) THEN
        IF(TAB(L,J).EQ.O) GO TO 501
        DD=TAB(NNOC,J)/TAB(L,J)
        DO 503 K=1,NOV
        TAA(L,K)=-DD*TAB(L,K)
        MARY=NOV+1
        TAA(L,MARY)=-DD*IV(L)
```

```
      TBC(NNOC,K)=TAB(NNOC,K)+TAA(L,K)
      TAB(NNOC,K)=TBC(NNOC,K)
503   CONTINUE
      IV(NNOC)=IV(NNOC)+TAA(L,MARY)
      GO TO 501
      ENDIF
      ENDIF
502   CONTINUE
501   CONTINUE
      NNOC=NNOC+1
500   CONTINUE
C
C TO EVALUATE THE VALUE OF PRIORITY LEVELS
      NOPOO=NOPRO+1
      DO 600 K=1,NOPOO
      DO 601 I=1,NOV
      DO 602 J=1,NOC
      IF(IB(J).EQ.ID(I)) THEN
      Z(K,I)=0.0
      GO TO 601
      ENDIF
602   CONTINUE
      BSUM=0.0
      ASUM=0.0
      DO 603 J=1,NOC
      KBK=IB(J)
      ASUM=ASUM+SCC(K,KBK)*TAB(J,I)
      BSUM=BSUM+SCC(K,KBK)*IV(J)
603   CONTINUE
      Z(K,I)=ASUM-SCC(K,I)
      IN(K)=BSUM
601   CONTINUE
600   CONTINUE
      NOPRO=NOPOO
      CALL PTRG(NOV)
C
C TO  FIND THE OPTIMAL SOLUTION
C
      NOPRO=NOPOO
      DO 902 I=1,NOC
      IF(IV(I).LT.0.) THEN
      NOC2=NOC
      CALL DUALSX(IPROW,LAB,IBALL)
      RETURN
      ENDIF
 902  CONTINUE
```

```
      DO 604 INO=1,NOPRO
605   CALL PIVCOL
      IF(XMAX.EQ.O.OR.IPVC.EQ.O) GO TO 604
      CALL PIVROW(IPROW)
      CALL CALC(IPROW)
      CALL PTRG(NOV)
      IF(INO.EQ.1) GO TO 605
      IF(IN(INO).EQ.O) GO TO 604
604   CONTINUE
      RETURN
      ENDIF
C
C TO CHANGE A(I,J) THE COEFFICIENTS OF THE JTH VAR.IN THE ITH ROW
C
C NOTE A(I,J) ARE ASSOCIATED WITH NONBAIC VARIABLES ONLY
C
      IF(ISEN.EQ.4) THEN
      WRITE(6,300)
      WRITE(10,300)
300   FORMAT(5X,'IT IS ONLY POSSIBLE TO CHANGE THE COEFICIENT OF
     +  THE NONBASIC VARIABLES')
C FIND THE LIST OF NONBASIC VARIABLES
      K=O
      DO 301 I=1,NOV
      LBC=O
      DO 302 J=1,NOC
      IF(ID(I).NE.IB(J)) THEN
      LBC=LBC+1
      ENDIF
302   CONTINUE
      IF(LBC.EQ.NOC) THEN
      K=K+1
      LNBVG(K)=ID(I)
      ENDIF
301   CONTINUE
      DO 304 LK=1,K
      WRITE(6,303) LNBVG(LK)
      WRITE(10,303) LNBVG(LK)
303   FORMAT(/5X,I2)
304   CONTINUE
C
      WRITE(6,305)
      WRITE(10,305)
305   FORMAT(/5X,'ENTER THE ROW #, VARIABLE #,AND THEN ITS VALUE')
77    READ(5,*) IZC,IZB,POT
      WRITE(6,76) IZC,IZB,POT
```

```
        WRITE(10,76) IZC,IZB,POT
76      FORMAT(2X,'ROW =',2X,I3,2X,'VAR=',2X,I3,2X,'VALUE=',2X,F8.4)
        WRITE(6,52)
        WRITE(10,52)
        READ(5,*) ICORR
        IF(ICORR.EQ.2) THEN
        WRITE(6,53)
        WRITE(10,53)
        GO TO 77
        ENDIF
        STOF(IZC,IZB)=POT
        CALL BINVRS(SENS,ISKIL)
        DO 306 LIB=1,ISKIL
        SUM=0
        DO 307 LIC=1,ISKIL
307     SUM=SUM+SENS(LIB,LIC)*STOF(LIC,IZB)
        BMOD(LIB)=SUM
306     CONTINUE
        DO 309 I=1,ISKIL
309     TAB(I,IZB)=BMOD(I)
C
C TO FIND THE VALUE OF Z(I,J)
C
        DO 3007 I=1,NOPRO
        SUM=0
        DO 308 J=1,NOC
        MP=IB(J)
        SUM=SUM+C(I,MP)*TAB(J,IZB)
308     CONTINUE
        Z(I,IZB)=SUM-C(I,IZB)
3007     CONTINUE
C
        CALL PTRG(NOV)
        DO 3009 INO=1,NOPRO
310     CALL PIVCOL
        IF(XMAX.EQ.0.OR.IPVC.EQ.0) GO TO 3009
        CALL PIVROW(IPROW)
        CALL CALC(IPROW)
        CALL PTRG(NOV)
        IF(INO.EQ.1) GO TO 310
        IF(IN(INO).EQ.0) GO TO 3009
3009     CONTINUE
        ENDIF
        RETURN
C
2       WRITE(6,3)
```

```
      WRITE(10,3)
3     FORMAT(/5X,'DO YOU WISH TO CONTINUE WITH THE RESULTS '
     +/5X,'ASSOCIATED WITH THE SENSITIVITY ANALYSIS')
      WRITE(6,4)
      WRITE(10,4)
4     FORMAT(/5X,'** ENTER 1 FOR YES *'//5X,'** ENTER 2 FOR NO **')
C
      READ(5,*) NNOOYY
      IF(NNOOYY.EQ.1) RETURN
      IF(NNOOYY.EQ.2) THEN
      NOC=NCO
      NOV=NVO
      NOPRO=NPO
      DO 5  I=1,NOC
      DO 6 J=1,NOV
6     TAB(I,J)=STABB(I,J)
      IV(I)=SIV(I)
      IB(I)=ISIN(I)
5     CONTINUE
C
      DO 7 I=1,NOPRO
      DO 8 J=1,NOV
      Z(I,J)=SZZ(I,J)
8     ID(I)=ISID(I)
7     IN(I)=SIN(I)
      ENDIF
      RETURN
      END
```

APPENDIX B

INTERACTIVE COMPUTER PROGRAM FOR THE

STOCHASTIC VEHICLE ROUTING PROBLEM

```
C*******************************************
C*                                         *
C*     FORTRAN COMPUTER PROGRAM FOR THE    *
C*     STOCHASTIC VEHICLE ROUTING PROBLEM  *
C*                (SVRP)                   *
C*******************************************
C*                                         *
C*       AUTHOR : YAHYA ZARE-MEHRJERDI     *
C*      ADVISOR : DR.M.P.TERRELL           *
C*         DATE : NOVEMBER 1986            *
C*     COMPUTER : IBM 3081D                *
C*                                         *
C*                                         *
C*    SCHOOL OF INDUSTRIAL ENGINEERING     *
C*           AND MANAGEMENT                *
C*    STILLWATER, OK. 74078                *
C*                                         *
C*******************************************
C*
C*****************************************************************
C*   THIS PROGRAM ALLOWS THE USERS TO SOLVE THE FOLLOWING TYPES OF
C*   THE SVRP:
C*
C*   1. DETERMINISTIC VEHICLE ROUTING PROBLEM (DVRP)
C*
C*   2. STOCHASTIC VEHICLE ROUTING PROBLEM HAVING ONLY PROBABILISTIC
C*      CUSTOMER DEMANDS(SVRP)
C*
C*   3. STOCHASTIC VRP WITH PROBABILISTIC CUSTOMER DEMAND AND
C*      TRAVEL AND UNLOAD TIMES OF THE "F" TYPE PROBLEM AND
C*
C*   4. STOCHASTIC VRP WITH PROBABILISTIC CUSTOMER DEMAND AND
C*      TRAVEL AND UNLOAD TIMES OF THE "F" TYPE PROBLEM
C*
C*****************************************************************
C*
C* THE FOLLOWING SUBROUTINGS ARE USED IN THIS PROGRAM:
C*
C* MEMOH =THIS SUBROUTING PROVIDES THE AVAILABLE MENUES
C*        FOR THE USERS
C* DETERM =IT IS USED FOR CONTROLLING THE PROCESS OF PROBLEM
C*         SOLVIG OF THE SVRP
C* TSORT, HEAPSN,SWAPN,PUSHDN AND SAVMAT = THESE SUBROUTINES
C* PERFORM TOGETHER TO PROVIDE THE SORTED SAVINGS FOR THE VRP AND SVRP
C* INPT =THIS SUBROUTINE PROVIDES THE INPUT DATA FOR THE DVRP
C* RTCONT = IT IS USED TO CONSTRUCT THE VEHICLE ROUTES
C* INTR =CHECKS THE EXISTENCE OF ANY INTERIOR STATION IN CONSTRUCTED
C*       VEHICLE ROUTES FOR ROUTING A NEW PAIR OF SELECTED STATIONS
C* COMBND =TO ADD A NEW STATION INTO AN AVAILABLE ROUTE
C* COMBRT = IT IS USED FOR PURPOSE OF COMBINING TWO ROUTES TOGETHER
C* FEASBL =CHECKS THE FEASIBILITY OF THE ROUTES FOR DVRP
```

```
C* CTD    =EVALUATE THE COST ,TIME OR DISTANCE OF THE CONSTRUCTED
C*          VEHICLE ROUTES FOR DVRP
C* CHCKK =EVALUATES THE TOTAL DEMAND OF EACH ROUTE FOR DVRP
C*  WWRT =PRINTS THE FINAL INFORMATIONS FOR EACH CONSTRUCTED ROUTE
C* SWTCH =IT PERFORMS THE TASK OF SWITCHING THE PLACE OF AVAILABLE
C*         ROUTES AFTER TWO ROUTES HAVE BEEN COMBINED TOGETHER
C*  PROB =CONTROL THE PROGRAM FOR THE SVRP OF THE "E" TYPE PROBLEM
C* STINPT=TO READ THE INPUT DATA FOR THE SVRP
C8** PRCHCK=CHECKS THE ROUTE FEASIBILITY FOR THE "E" AND "F" TYPE
C*         PROBLEMS
C* STFSBL=CHECK THE FEASIBILITY OF ADDING A NODE INTO A ROUTE OR
C*         COMBINING TWO ROUTES TOGETHER
C* CONTRL=EVALUATE THE VARIANCE DEMAND,TRAVELLING,AND UNLOADING
C*         TIMES BEFOR ADDITION OF ANY NODE INTO A ROUTE
C*  SOFT =DETERMINE MEAN AND VARIANCE OF TRAVEL TIME OF EACH ROUTE
C*  RUSH =DETERMINE MEAN AND VARIANCE OF DEMAND AND UNLOAD TIMEOF ROUTE
C* STSAVE=DETERMINE THE SORTED SAVIGES FOR THE SVRP OF THE "F"
C*         TYPE PROBLEM
C*  FSBL= CHECK THE FEASIBILITY FOR SVRP WITH ONLY PROBABILISTIC
C*         CUSTOMER DEMANDS
C* FCHECK=IS USED FOR THE SVRP WITH THE PROBABILISTIC DEMAND
C* FAST=DETERMINE TOTAL DEMAND OF A ROUTE BEFORE ADDITION OF A NEW NODE
C* STCONT=EVALUATE THE VARIANCE OF DEMAND
C*  STARS=IS USED FOR SOLVING THE "E" TYPE PROBLEM
C*  STCTD=EVALUATE THE TOTAL ELAPSE TIME FOR SVRP OF THE "F" TYPE
C*         PROBLEM
C***********************************************************
C*   THE FOLLOWING IS THE LIST OF VARIABLES USED IN THIS PROGRAM
C*
C*     NPT = NUMBER OF DEMAND POINTS INCLUDING CENTRAL DEPOT
C*    TCAP = VEHICLE CAPACITY. ALL VEHICLES ARE ASSUMED TO BE
C*           HOMOGENOUES
C*    X(I) = IS THE X COORDINATE OF STATION I
C*    Y(I) = IS THE Y COORDINATE OF STATION I
C*DIST(I,J)= DISTANCE BETWEEN STATIONS I AND J
C*     DDT = STANDS FOR THE DETERMINISTIC VRP
C*     SST = STANDS FOR THE STOCHASTIC VRP
C*    MSVA = ARRAY OF SAVING AFTER SORTING
C*   NSAVE = ARRAY OF SAVING BEFOR SORTING
C*   NB(I) = A POINTER WHICH SHOWS THE FIRST ELEMENT OF EACH VEHICLE
C*           ROUTE
C*   NF(I) = A POINTER WHICH SHOWS THE LAST ELEMENT OF EACH VEHICLE
C*           ROUTE
C*   NR(I) = A POINTER WHICH SHOWS THE NUMBER OF STATIONS ON EACH
C*           VEHICLE ROUTE
C*TDMAND(I)= TOTAL DEMAND OF VEHICLE ROUTE I
```

```
C*       P = NUMBER OF CONSTRUCTED VEHICLE ROUTE
C*    NTRY = IS THE NUMBER OF SORTED SAVINGS WHICH ARE GREATER THAN
C*           ZERO
C*ROUTE(I,J)=INDICATES THE JTH ELEMENT OF ROUTE I
C*   ALPHA = IS THE NORMAL DEVIATE OF THE ROUTE FAILURE PROBABILITY
C*           FOR TRAVEL TIME
C*    BATA = IS THE NORMAL DEVIATE OF THE ROUTE FAILURE PROBABILITY
C*           FOR UNLOAD TIME
C*    ATAH = IS THE NORMAL DEVIATE OF THE ROUTE FAILURE PROBABILITY
C*           FOR THE DEMAND
C*   UTIME = UPPER BOUND OF UNLOAD TIME FOR EACH VEHICLE ROUTE
C*  TTTIME = UPPER BOUND OF TRAVEL TIME FOR EACH VEHICLE ROUTE
C*     IEE = INDICATES THE "E" TYPE PROBLEM
C*     IFF = INDICATES THE "F" TYPE PROBLEM
C*   DELTA = A CONSTANT VALUE USED IN THE ALGORITHM 2 OF "F" TYPE
C*           PROBLEM
C*  IALGOL = INDICATES THE TYPE OF ALGORITHM: 1 OR 2.
C*   BKAMA = INDICATES THE VALUE OF GAMA FOR ALGORITHM 1 OF "F" TYPE
C*           PROBLEM
C*    KPRO = INDICATES THAT THE PROBLEM IS SVRP WITH ONLY PROBABILISTIC
C*           CUSTOMER DEMAND
C*DMAND(I) = MEAN DEMAND OF DEMAND POIT I
C*VDMAND(I)= VARIANCE OF DEMAND POINT I
C*MEAN(I,J)= MEAN TRAVEL TIME BETWEEN STATINS I AND J
C*VARS(I,J)= VARIANCE OF TRAVEL TIME BETWEEN STATIONS I AND J
C* MINE(I) = MEAN UNLOAD TIME OF STATION I
C* VIRS(I) = VARIANCE OF UNLOAD TIME OF STATION I
C*  IPROBL = NUMBER OF PROBLEMS TO BE SOLVED
C* NCUSTOM = NUMBER OF CUSTOMER DEMANDS TO BE CHANGED
C* NLOCAT  = NUMBER OF LOCATIONS WHICH THEIR COORDINATES RAE NEEDED
C*           TO BE CHANGED
C*   IDMN  = MEAN DEMAND OF THE SPECIAL LOCATION THAT NEED TO BE CHANGED
C*   IVDMN = VARIANCE DEMAND OF THE SPECIAL LOCATION THAT NEED TO BE
C*           CHANGED
C*  NCHANG = NUMBER OF NECESARRY CHANGES IN THE TRAVEL TIME
C*TIHAT(I) = AN ARRAY THAT KEEP THE ITH SUBSCRIPT OF THE SORTED
C*           SAVING S(I,J)
C*TJHAT(I) = AN ARRAY THAT KEEP THE JTH SUBSCRIPT OF THE SORTED
C*           SAVING S(I,J)
C*     DD  = TOTAL COST, TIME ,OR DISTANCE
C*TULOAD(I)= TOTAL UNLOAD TIME OF ROUTE I
C*TTRAVL(I)= TOTAL TRAVEL TIME OF ROUTE I
C*WAR(I,J) = THE SAVING IN VARIANCE FOR RANDOM VARIABLE TRAVEL TIME
C*           USING ALGORITHMS(I) AND (II) OF "F" TYPE PROBLEM
C*MAR(I,J) = THE SAVING IN MEAN FOR RANDOM VARIABLE TRAVEL TIME
C*           USING ALGORITHM (I) AND (II) OF "F" TYPE PROBLEM
```

```
C*  TTOTAL = TOTAL TRAVEL TIME FOR "F" TYPE PROBLEM
C*    IDDT = O USING EUCLIDIAN DISTANCE
C*         = 1 USING STRAIGHT LINE DISTANCE
C*    GAMA = IS THE NORMAL DEVIATE OF THE ROUTE FAILURE PROBABILITY
C*           FOR SVRP HAVING ONLY PROBABILISTIC CUSTOMER DEMANDS
C*   IDSTB = 1 FOR DISTRIBUTIONS SUCH AS POISSON,BINOMIAL,GAMMA,
C*             EXPONENTIAL,NEGATIVE BINOMIAL,AND CHI-SQURE
C*           O FOR OTHER DISTRIBUTIONS
C
C
C****************************************************
C*               MAIN PROGRAM                       *
C****************************************************
         DIMENSION NSA(5000),TI(5000),TJ(5000)
         INTEGER TIHAT,TJHAT,TI,TJ,NSA,DIST
         INTEGER FLI,FLJ,FLIJ,FIJ,ROUTE,DMAND,R,P,XX
         INTEGER TCAP,X,Y,T,PP,TT,DDT,SST
         COMMON/A1/X(300),Y(300)
         COMMON/A2/NPT,NW,TCAP,MNP,NTRY
         COMMON/A3/MSVA(5000),NSAVE(5000),XX(5000)
         COMMON/A4/NB(100),NF(100),NR(100),P
         COMMON/A5/DMAND(300),TDMAND(100)
         COMMON/A6/LI,LJ,LI1,LI2,LJ1,LJ2,LRI,LRJ
         COMMON/A7/IBV,IWB
         COMMON/A8/DIST(300,300)
         COMMON/A9/TIHAT(5000),TJHAT(5000),ROUTE(100,100)
         COMMON/A10/ALPHA,BATA,ATAH,UTIME,TTTIME
         COMMON/A12/DDT,SST,IZAR
         COMMON/A15/IEE,IFF,DELTA,IALGOL,BKAMA
         COMMON/A16/KPRO,GAMA
C* READ THE NUMBER OF PROBLEMS TO BE SOLVED
 35      WRITE(6,30)
         WRITE(10,30)
 30      FORMAT(//5X,'NUMBER OF PROBLEMS YOU WISH TO SOLVE')
         READ(5,*) IPROBL
         WRITE(6,31) IPROBL
         WRITE(10,31) IPROBL
 31      FORMAT(//5X,'NUMBER OF PROBLEMS=',2X,I2)
         WRITE(6,3)
         WRITE(10,3)
         READ(5,*) ICORR
         IF(ICORR.EQ.2) THEN
         WRITE(6,4)
         WRITE(10,4)
         GO TO 35
         ENDIF
```

```
          DO 99 I=1,IPROBL
          IF(IPROBL.GE.1) THEN
          WRITE(6,33)
          WRITE(10,33)
 33       FORMAT(//5X,' SELECT ONE OF THE FOLLOWINGS')
          ELSE
          WRITE(6,34)
          WRITE(10,34)
 34       FORMAT(//5X,'ERROR MESSAGE',2X,'REENTER AGAIN')
          GO TO 35
          ENDIF
C
          IFAA=0
          IZAR=1
          DELGAM=0
C
C TO DISPLAY MENU 1
C
          KALL=1
          CALL MEMOH(MOO,KALL,MOH)
C
C PURPOSE TO SOLVE THE DETERMINISTIC VRP
1000      IF(DDT.EQ.1) THEN
          CALL DETERM
          GO TO 999
          ENDIF
C PURPOSE TO SOLVE THE SVRP HAVING ONLY PROBABILISTIC DEMAND
C
          IF(KPRO.EQ.1) THEN
          CALL STATS
          GO TO 999
          ENDIF
C PURPOSE TO SOLVE THE SVRP
501       IF(SST.EQ.1) THEN
C PURPOSE TO SOLVE THE "E" TYPE PROBLEM OF SVRP
          IF(IEE.EQ.1) THEN
          CALL PROB
          GO TO 999
          ENDIF
C PURPOSE TO SOLVE THE "F" TYPE PROBLEM OF SVRP
          IF(IFF.EQ.1) THEN
  8       FORMAT(//15X,'--->',2X,'ENTER A VALUE FOR DELTA')
  9       FORMAT(//15X,'--->','SUGGESTED VALUES ARE'/19X,'.5,1.,1.5,
     +2,2.5,3,3.5,4')
7         WRITE(6,32)
          WRITE(10,32)
```

```
32        FORMAT(//10X,'ENTER YOUR CHOICE OF ALGORITHS "F" TYPE PROB')
          WRITE(6,36)
          WRITE(10,36)
36        FORMAT(//10X,'ENTER 1 ----> ALGORITHM I'/16X,'2 ---->ALGORITHM
     +II')
C
          READ(5,*) IALGOL
          WRITE(6,5) IALGOL
          WRITE(10,5) IALGOL
 5        FORMAT(//5X,'THE SELECTED ALGORITHM IS',2X,I2)
          WRITE(6,3)
          WRITE(10,3)
          READ(5,*) ICORR
          IF(ICORR.EQ.2) THEN
          WRITE(6,4)
          WRITE(10,4)
          GO TO 7
          ENDIF
C TO OBSERVE FOR CHANGES OF ALGORITHM
          IFAA=IFAA+1
          IF(IFAA.GE.2) THEN
          WRITE(6,502)
          WRITE(10,502)
502       FORMAT(5X,'DO YOU WISH TO CHANGE THE VALUE OF DELTA OR GAMA')
          WRITE(6,503)
          WRITE(10,503)
503       FORMAT(5X,'ENTER',2X,'1:YES',2X,'2:NO')
C TO SEE IF THE USER WANTS TO CHANGE THE VALUE OF DELTA OR GAMA
          READ(5,*) DELGAM
          IF(DELGAM.EQ.1) IZAR=2
          ENDIF
          IF(IFAA.EQ.1) THEN
          IFALGL=IALGOL
          ELSE
          IF(IFALGL.EQ.IALGOL) GO TO 888
          IZAR=2
          IFALGL=IALGOL
          ENDIF
C TO USE THE ALGORITHM(II) OF "F" TYPE PROBLEM
888       IF(IALGOL.EQ.2) THEN
12        WRITE(6,8)
          WRITE(10,8)
          WRITE(6,9)
          WRITE(10,9)
          READ(5,*) DELTA
          WRITE(6,6) DELTA
```

```
          WRITE(10,6) DELTA
6         FORMAT(//5X,'DELTA=',F8.3)
          WRITE(6,3)
          WRITE(10,3)
          READ(5,*) ICORR
          IF(ICORR.EQ.2) THEN
          WRITE(6,4)
          WRITE(10,4)
          GO TO 12
          ENDIF
          ENDIF
C TO USE THE ALGORITHM(I) OF "F" TYPE PROBLEM
          IF(IALGOL.EQ.1) THEN
14        WRITE(6,37)
          WRITE(10,37)
37        FORMAT(//10X,'ENTER A VALUE FOR GAMA')
          WRITE(6,38)
          WRITE(10,38)
38        FORMAT(//10X,' 0< GAMA < = 1 ')
          READ(5,*) BKAMA
          WRITE(6,21) BKAMA
          WRITE(10,21) BKAMA
21        FORMAT(//5X,'BKAMA=',2X,F8.3)
          WRITE(6,3)
          WRITE(10,3)
          READ(5,*) ICORR
          IF(ICORR.EQ.2) THEN
          WRITE(6,4)
          WRITE(10,4)
          GO TO 14
          ENDIF
          CALL STATS
          GO TO 999
          ENDIF
          IF(DELTA.EQ.0) THEN
          WRITE(6,11)
          WRITE(10,11)
 11       FORMAT(//15X,'--->',2X,'ZERO IS NOT ACCEPTABLE, TRY AGAIN')
          GO TO 12
          ENDIF
          CALL STATS
          ELSE
          WRITE(6,20)
          WRITE(10,20)
 20       FORMAT(5X,'PLEASE CHECK YOUR FIRST DATA CARD')
          ENDIF
```

```
      ENDIF
C
C TO DO ANY NECESSARY CHANGES FOR THIS PROBLEM BEFOR MOVING TO ANOTHER
C PROBLEM.
C TO DISPLAY MENU 2
999   KALL=2
      CALL MEMOH(MOO,KALL,MOH)
      IF(MOH.EQ.12) GO TO 99
      IF(MOH.EQ.10) THEN
      SST=1
      IEE=1
      IFF=0
      IZAR=4
      CALL PROB
      GO TO 999
      ENDIF
      IF(MOH.EQ.11) GO TO 501
      GO TO 1000
 99   CONTINUE
 3    FORMAT(/5X,'CORRECT',2X,'ENTER',2X,'1:YES',2X,'2:NO')
 4    FORMAT(/5X,'REENTER AGAIN')
      STOP
      END
C**************************************************
C*            SUBROUTINE MEMOH                   *
C**************************************************
C
      SUBROUTINE MEMOH(MOO,KALL,MOH)
      DIMENSION MEAN(300,300),VIRS(300),MINE(300),VARS(300,300)
      DIMENSION VDMAND(300)
      COMMON/A1/X(300),Y(300)
      COMMON/A2/NPT,NW,TCAP,MNP,NTRY
      COMMON/A3/MSVA(5000),NSAVE(5000),XX(5000)
      COMMON/A4/NB(100),NF(100),NR(100),P
      COMMON/A5/DMAND(300),TDMAND(100)
      COMMON/A9/TIHAT(5000),TJHAT(5000),ROUTE(100,100)
      COMMON/A10/ALPHA,BATA,ATAH,UTIME,TTTIME
      COMMON/A12/DDT,SST,IZAR
      COMMON/A15/IEE,IFF,DELTA,IALGOL,BKAMA
      COMMON/A16/KPRO,GAMA
      INTEGER DDT,ROUTE,P,SST,TCAP,X,Y,DMAND
      INTEGER VDMAND,VARS,VIRS,UTIME,TTTIME
3     FORMAT(/5X,'CORRECT',2X,'ENTER',2X,'1:YES',2X,'2:NO')
4     FORMAT(/5X,'REENTER AGAIN')
      GO TO (1,2),KALL
C
```

```
C DISPLAY OF MENU 1
C
1       WRITE(6,8)
        WRITE(10,8)
8       FORMAT(15X,'*** DISPLAY OF MENU 1 *** ')
        WRITE(6,10)
        WRITE(10,10)
10      FORMAT(//5X,'TO SOLVE THE DETERMINISTIC VRP')
        WRITE(6,11)
        WRITE(10,11)
11      FORMAT(5X,' **  ENTER 1 **')
        WRITE(6,20)
        WRITE(10,20)
20      FORMAT(5X,'TO SOLVE A SVRP WITH PROBABILISTIC DEMAND')
        WRITE(6,21)
        WRITE(10,21)
21      FORMAT(5X,'**  ENTER  2 **')
        WRITE(6,30)
        WRITE(10,30)
30      FORMAT(5X,'TO SOLVE SVRP OF "E" TYPE PROBLEM')
        WRITE(6,31)
        WRITE(10,31)
31      FORMAT(5X,'** ENTER  3 **')
        WRITE(6,40)
        WRITE(10,40)
40      FORMAT(5X,'TO SOLVE SVRP OF "F" TYPE PROBLEM')
        WRITE(6,41)
        WRITE(10,41)
41      FORMAT(5X,'** ENTER  4**')
C
        WRITE(6,9)
        WRITE(10,9)
9       FORMAT(//5X,'*** CHOOSE THE OPTION *** ')
        IF(IZAR.EQ.2) THEN
        WRITE(6,621)
        WRITE(10,621)
621     FORMAT(/5X,'ENTER ONLY 3 OR 4 ')
        ENDIF
        READ(5,*) MOO
        WRITE(6,3)
        WRITE(10,3)
        READ(5,*) ICORR
        IF(ICORR.EQ.2) THEN
        WRITE(6,4)
        WRITE(10,4)
        GO TO 1
```

```
        ENDIF
C
        DDT=0
        SST=0
        IEE=0
        IFF=0
        KPRO=0
        IF(MOO.EQ.1) THEN
        WRITE(6,50)
        WRITE(10,50)
        DDT=1
        GO TO 999
        ENDIF
        IF(MOO.EQ.2) THEN
        WRITE(6,60)
        WRITE(10,60)
        SST=1
        KPRO=1
        GO TO 999
        ENDIF
        IF(MOO.EQ.3) THEN
        WRITE(6,70)
        WRITE(10,70)
        SST=1
        IEE=1
        GO TO 999
        ENDIF
        IF(MOO.EQ.4) THEN
        WRITE(6,80)
        WRITE(10,80)
        SST=1
        IFF=1
        GO TO 999
        ENDIF
50      FORMAT(//5X,'A SOLUTION TO DVRP IS REQUIRED')
60      FORMAT(//5X,'A SOLUTION TO SVRP WITH PROBABILISTIC DEMAND'/
     +5X,'IS REQUIRED')
70      FORMAT(//5X,'YOUR PROBLEM IS SVRP OF "E" TYPE PROBLEM ')
80      FORMAT(//5X,'YOUR PROBLEM IS SVRP OF "F" TYPE PROBLEM ')
999     RETURN
 2      CONTINUE
C
        ICHANG=1
C DISPLAY OF MENU 2
519     WRITE(6,7)
        WRITE(10,7)
```

```
7       FORMAT(15X,'*** DISPLAY OF MENU 2***')
        WRITE(6,90)
        WRITE(10,90)
 90     FORMAT(5X,'** DO YOU WISH TO DO ANY CHANGES **')
        WRITE(6,100)
        WRITE(10,100)
 100    FORMAT(5X,'TO CHANGE THE CAPACITY OF TRUCK')
        WRITE(6,101)
        WRITE(10,101)
 101    FORMAT(5X,'** ENTER  1 **')
        WRITE(6,102)
        WRITE(10,102)
 102    FORMAT(5X,'TO CHANGE THE "UTIME" OR "TTTIME" ')
        WRITE(6,103)
        WRITE(10,103)
 103    FORMAT(5X,'** ENTER 2 **')
        WRITE(6,104)
        WRITE(10,104)
 104    FORMAT(5X,'TO CHANGE "ALPHA","BATA" AND "ATAH" ')
        WRITE(6,105)
        WRITE(10,105)
105     FORMAT(5X,'** ENTER 3 **')
        WRITE(6,106)
        WRITE(10,106)
106     FORMAT(5X,'TO CHANGE THE COORDINATE OF LOCATIONS')
        WRITE(6,107)
        WRITE(10,107)
107     FORMAT(5X,'** ENTER 4 **')
        WRITE(6,108)
        WRITE(10,108)
108     FORMAT(5X,'TO CHANGE THE CUSTOMER DEMAND')
        WRITE(6,109)
        WRITE(10,109)
109     FORMAT(5X,'** ENTER 5 **')
        WRITE(6,110)
        WRITE(10,110)
110     FORMAT(5X,'TO CHANGE THE UNLOAD TIME')
        WRITE(6,111)
        WRITE(10,111)
111     FORMAT(5X,' ** ENTER 6 **')
        WRITE(6,112)
        WRITE(10,112)
112     FORMAT(5X,' TO CHANGE THE TRAVEL TIME')
        WRITE(6,113)
        WRITE(10,113)
113     FORMAT(5X,'** ENTER 7 **')
```

```
      WRITE(6,114)
      WRITE(10,114)
114   FORMAT(5X,'** TO DO NO CHANGES ENTER 8 ***')
      WRITE(6,622)
      WRITE(10,622)
622   FORMAT(5X,'TO CHANGE ALGORITHM I INTO II OR VISE VERSA')
      WRITE(6,623)
      WRITE(10,623)
623   FORMAT(5X,'*** ENTER 9 ***')
      IF(ICHANG.EQ.1) THEN
      WRITE(6,91)
      WRITE(10,91)
      READ(5,*) NCHA
      ENDIF
      IF(NCHA.GT.1) THEN
      WRITE(6,501)
      WRITE(10,501)
      ENDIF
      WRITE(6,25)
      WRITE(10,25)
25    FORMAT(15X,'*** CHOOSE THE  OPTION ***')
518   READ(5,*) MOH
      WRITE(6,624) MOH
      WRITE(10,624) MOH
624   FORMAT(/5X,'YOUR OPTION =',2X,I3)
      WRITE(6,3)
      WRITE(10,3)
      READ(5,*) ICORR
      IF(ICORR.EQ.2) THEN
      WRITE(6,4)
      WRITE(10,4)
      GO TO 518
      ENDIF
      IF(MOH.EQ.9) THEN
      IZAR=2
      GO TO 130
      ENDIF
      IF(MOH.EQ.8) THEN
      WRITE(6,618)
      WRITE(10,618)
618   FORMAT(/5X,'DO YOU WISH TO SOLVE THIS PROBLEM BY ANOTHER'/
     +5X,'METHOD. POSIBLE SELECTIONS ARE "F"-->"E",AND "E"-->"F"')
      WRITE(6,619)
      WRITE(10,619)
619   FORMAT(/5X,'ENTER',2X,'1:F--->E',2X,'2:E--->F',2X,'3:NO')
      READ(5,*) KYNOO
```

```
C PURPOSE TO CHANGE THE "F" TYPE PROBLEM INTO "E" TYPE
        IF(KYNOO.EQ.1) THEN
        MOH=10
        IZAR=4
        ENDIF
C PURPOSE TO CHANGE THE "E" TYPE PROBLEM INTO "F" TYPE
        IF(KYNOO.EQ.2) THEN
        SST=1
        IFF=1
        IEE=0
        MOH=11
        IZAR=2
        ENDIF
        IF(KYNOO.EQ.3) THEN
        MOH=12
        RETURN
        ENDIF
        GO TO 130
        ENDIF
91      FORMAT(5X,'ENTER THE NUMBER OF CHANGES')
 501    FORMAT(5X,'CHANG IN COORDINATION OR TRAVAL. TIME COMES LAST')
        IF(MOH.EQ.1) THEN
        WRITE(6,120)
        WRITE(10,120)
120     FORMAT(5X,'ENTER THE NEW CAPACITY OF TRUCK')
502     READ(5,*) TCAP
        WRITE(6,115) TCAP
        WRITE(10,115) TCAP
115     FORMAT(//5X,'THE NEW CAPACITY OF TRUCK =',2X,I4)
        WRITE(6,3)
        WRITE(10,3)
        READ(5,*) ICORR
        IF(ICORR.EQ.2) THEN
        WRITE(6,4)
        WRITE(10,4)
        GO TO 502
        ENDIF
        IZAR=3
        DO 200 I=1,NTRY
200     NSAVE(I)=MSVA(I)
        GO TO 130
        ENDIF
        IF(MOH.EQ.2) THEN
503     WRITE(6,121)
        WRITE(10,121)
121     FORMAT(5X,'ENTER NEW VALUES FOR UTIME AND TTTIME')
```

```
      READ(5,*) UTIME ,TTTIME
      WRITE(6,3)
      WRITE(10,3)
      READ(5,*) ICORR
      IF(ICORR.EQ.2) THEN
      WRITE(6,4)
      WRITE(10,4)
      GO TO 503
      ENDIF
      WRITE(6,117) UTIME ,TTTIME
      WRITE(10,117) UTIME,TTTIME
117   FORMAT(//5X,'UTIME=',2X,I5,2X,'TTTIME=',2X,I5)
      IZAR=3
      DO 201 I=1,NTRY
201   NSAVE(I)=MSVA(I)
      GO TO 130
      ENDIF
      IF(MOH.EQ.3) THEN
504   WRITE(6,122)
      WRITE(10,122)
122   FORMAT(5X,'ENTER VALUES FOR ALPHA , BATA AND ATAH')
      READ(5,*) ALPHA,BATA,ATAH
      WRITE(6,118)
      WRITE(10,118)
118   FORMAT(//5X,'THE NEW VALUES FOR ALPHA ,BATA AND ATAH ARE')
      WRITE(6,119) ALPHA,BATA,ATAH
      WRITE(10,119) ALPHA,BATA,ATAH
119   FORMAT(//5X,'ALPHA=',2X,F6.3,2X,'BATA=',2X,F6.3,2X,'ATAH='
     +,2X,F6.3)
      IZAR=3
      WRITE(6,3)
      READ(5,*) ICORR
      IF(ICORR.EQ.2) THEN
      WRITE(6,4)
      WRITE(10,4)
      GO TO 504
      ENDIF
      GO TO 130
      ENDIF
      IF(MOH.EQ.4) THEN
505   WRITE(6,123)
      WRITE(10,123)
123   FORMAT(5X,'ENTER THE = OF LOCATIONS NEED TO BE CHANGED')
      READ(5,*) NLOCAT
      WRITE(6,3)
      WRITE(10,3)
```

```
      READ(5,*) ICORR
      IF(ICORR.EQ.2) GO TO 505
      DO 140 I=1,NLOCAT
506   WRITE(6,141)
      WRITE(10,141)
141   FORMAT(5X,'ENTER LOCATION,THEN ITS COORDINATES X AND Y')
      READ(5,*) LOCAT,JXJ,JYJ
      WRITE(6,507) LOCAT,JXJ,JYJ
      WRITE(10,507) LOCAT,JXJ,JYJ
507   FORMAT(5X,'LOCATION=',2X,I3,1X,'X=',2X,I3,2X,'Y=',2X,I3)
      WRITE(6,3)
      WRITE(10,3)
      READ(5,*) ICORR
      IF(ICORR.EQ.2) THEN
      WRITE(6,4)
      WRITE(10,4)
      GO TO 506
      ENDIF
      X(LOCAT)=JXJ
      Y(LOCAT)=JYJ
140   CONTINUE
      WRITE(6,202)
      WRITE(10,202)
202   FORMAT(25X,'** PLEASE WAIT **')
      IZAR=2
      GO TO 130
      ENDIF
      IF(MOH.EQ.5) THEN
508   WRITE(6,124)
      WRITE(10,124)
124   FORMAT(5X,'ENTER THE # OF CUSTOMER DEMAND POINTS TO BE CHANGED')
      READ(5,*) NCUSTM
      WRITE(6,42) NCUSTM
      WRITE(10,42) NCUSTM
42    FORMAT(//5X,'NUMBER OF CHANGES =',2X,I5)
      WRITE(6,3)
      WRITE(10,3)
      READ(5,*) ICORR
      IF(ICORR.EQ.2) GO TO 508
      DO 142 I=1,NCUSTM
510   WRITE(6,143)
      WRITE(10,143)
143   FORMAT(5X,'ENTER CUSTOMER # ,MEAN AND THEN VARIANCE OF DEMAND')
      READ(5,*) NCSTM,IDMN,IVDMN
      WRITE(6,509) NCSTM,IDMN,IVDMN
      WRITE(10,509) NCSTM,IDMN,IVDMN
```

```
509     FORMAT(5X,'CUSTOMER #',2X,I3,2X,'MEAN=',2X,I3,2X,'VAR=',2X,I3)
        WRITE(6,3)
        WRITE(10,3)
        READ(5,*) ICORR
        IF(ICORR.EQ.2) THEN
        WRITE(6,4)
        WRITE(10,4)
        GO TO 510
        ENDIF
        DMAND(NCSTM)=IDMN
        VDMAND(NCSTM)=IVDMN
142     CONTINUE
        IZAR=3
        DO 203 I=1,NTRY
203     NSAVE(I)=MSVA(I)
        GO TO 130
        ENDIF
        IF(MOH.EQ.6) THEN
511     WRITE(6,125)
        WRITE(10,125)
125     FORMAT(5X,'ENTER # OF CUSTOMERS WITH NEW UNLOAD TIME VALUES')
        READ(5,*) NCUSTM
        WRITE(6,517) NCUSTM
        WRITE(10,517) NCUSTM
517     FORMAT(5X,'# OF CUSTOMERS WITH NEW UNLOAD TIME =',2X,I3)
        WRITE(6,3)
        WRITE(10,3)
        READ(5,*) ICORR
        IF(ICORR.EQ.2) GO TO 511
        DO 144 I=1,NCUSTM
513     WRITE(6,145)
        WRITE(10,145)
145      FORMAT(5X,'ENTER CUSTOMER #,MEAN AND VAR OF UNLOAD TIME')
        READ(5,*) NCSTM,IDMN,IVDMN
        WRITE(6,512) NCSTM,IDMN,IVDMN
        WRITE(10,512) NCSTM,IDMN,IVDMN
512     FORMAT(5X,'CUSTOMER #',2X,I3,2X,'MEAN=',1X,I3,2X,'VAR=',2X,I3)
        WRITE(6,3)
        WRITE(10,3)
        READ(5,*) ICORR
        IF(ICORR.EQ.2) THEN
        WRITE(6,4)
        WRITE(10,4)
        GO TO 513
        ENDIF
        MINE(NCSTM)=IDMN
```

```
      VIRS(NCSTM)=IVDMN
144   CONTINUE
      IZAR=3
      DO 204 I=1,NTRY
204   NSAVE(I)=MSVA(I)
      GO TO 130
      ENDIF
      IF(MOH.EQ.7) THEN
      WRITE(6,126)
      WRITE(10,126)
126   FORMAT(5X,'ENTER THE # OF CHANGES OF TRAVEL TIME')
      READ(5,*) NCHANG
      WRITE(6,514) NCHANG
      WRITE(10,514) NCHANG
514   FORMAT(5X,'#OF CHANGES =',2X,I3)
      DO 147 I=1,NCHANG
516   WRITE(6,146)
      WRITE(10,146)
146   FORMAT(5X,'ENTER I,J,MEAN AND VARIANCE OF TRAVEL TIME')
      READ(5,*) K,L,KB,KZ
      WRITE(6,515) K,L,KB,KZ
      WRITE(10,515) K,L,KB,KZ
515   FORMAT(/5X,'I=',2X,I3,1X,'J=',2X,I3,'MEAN=',2X,I3,1X,'VAR=',
     +2X,I3)
      WRITE(6,3)
      WRITE(10,3)
      READ(5,*) ICORR
      IF(ICORR.EQ.2) THEN
      WRITE(6,4)
      WRITE(10,4)
      GO TO 516
      ENDIF
      MEAN(K,L)=KB
      VARS(K,L)=KZ
      MEAN(L,K)=MEAN(K,L)
      VARS(L,K)=VARS(K,L)
147   CONTINUE
      IZAR=2
      GO TO 130
      ENDIF
130   ICHANG=0
      NCHA=NCHA-1
      IF(NCHA.GE.1) GO TO 519
      DO 616 I=1,NW
      ROUTE(I,1)=I
      ROUTE(I,2)=1
```

```
      MNP1=MNP-1
      DO 617 J=3,MNP1
617   ROUTE(I,J)=0
      NB(I)=0
      NF(I)=0
616   NR(I)=2
      RETURN
      END
C
C*****************************************************
C*             SUBROUTINE DETERM                     *
C*****************************************************
C
       SUBROUTINE DETERM
C***
       DIMENSION NSA(5000),TI(5000),TJ(5000)
       INTEGER TIHAT,TJHAT,TI,TJ,NSA,DIST
       INTEGER FLI,FLJ,FLIJ,FIJ,ROUTE,DMAND,R,P,XX
       INTEGER TCAP,X,Y,T,PP,TT
       INTEGER  DDT,SST
       COMMON/A1/X(300),Y(300)
       COMMON/A2/NPT,NW,TCAP,MNP,NTRY
       COMMON/A3/MSVA(5000),NSAVE(5000),XX(5000)
       COMMON/A4/NB(100),NF(100),NR(100),P
       COMMON/A5/DMAND(300),TDMAND(100)
       COMMON/A6/LI,LJ,LI1,LI2,LJ1,LJ2,LRI,LRJ
       COMMON/A7/IBV,IWB
       COMMON/A8/DIST(300,300)
       COMMON/A9/TIHAT(5000),TJHAT(5000),ROUTE(100,100)
       COMMON/A12/DDT,SST,IZAR
C****   READ IN IDDT AS 0 OR 1.
C****         IDDT= 0 IF USING THE EUCLIDIAN DISTANCE
C****             = 1  1F USING THE STRAIGHT LINE DISTANCE
C****
       GO TO (4,5,8),IZAR
 4     WRITE(6,1)
       WRITE(10,1)
 1     FORMAT(2X,'-->',5X,'ENTER 0 FOR EUCLIDIAN DISTANCE'//
     + 2X,'-->',5X,'ENTER 1 FOR LINEAR DISTANCE')
       READ(5,*) IDDT
       CALL INPT(IDDT)
 5     CALL SAVMAT
       TT=NTRY
       CALL TSORT(NSAVE,TIHAT,TJHAT,NTRY)
C***   SET THE TOTAL DEMAND OF EACH ROUTE TO ZERO
       DO 50 I=1,NTRY
```

```
 50       MSVA(I)=NSAVE(I)
 8        DO 7 P=1,NW
 7        TDMAND(P)=0
          T=1
11        P=1
          R=3
          IX=5
          PP=P
          ITI=TIHAT(T)
          JTJ=TJHAT(T)
          TDMAND(P)=TDMAND(P)+DMAND(ITI)+DMAND(JTJ)
          IF(TDMAND(P).LE.TCAP) THEN
          ROUTE(P,R)=TIHAT(T)
          NB(P)=ROUTE(P,R)
          R=R+1
          ROUTE(P,R)=TJHAT(T)
          NF(P)=ROUTE(P,R)
          NR(P)=R
          ENDIF
          K=T+1
          IF(TDMAND(P).GT.TCAP) THEN
          NSAVE(K-1)=0
          TDMAND(P)=0
          T=K
          GO TO 11
          ENDIF
C***      CONSTRUCT THE ROUTE
          DO 10 T=K,TT
          NSAVE(T-1)=0
          IYOUTH=1
          CALL INTR(IN,PP,T,IYOUTH)
          IF(IN.EQ.1) GO TO 10
          PP=P
          CALL RTCONT(PP,T)
 10       CONTINUE
          IYOUTH=2
          CALL INTR(IN,PP,T,IYOUTH)
          CALL WWRT(PP)
          RETURN
          END
C*************************************************
C*              SUBROUTINE TSORT                *
C*************************************************
          SUBROUTINE TSORT (NSAVE,TIHAT,TJHAT,NTRY)
          DIMENSION TIHAT(5000),TJHAT(5000),TI(5000),TJ(5000),NSAVE(5000)
          DIMENSION NSA(5000)
```

```
      INTEGER TIHAT,TJHAT,TI,TJ,NSA,NSAVE
C**   TO SORT IN DECREASING ORDER
      CALL HEAPSN(NSAVE,TIHAT,TJHAT,NTRY)
      DO 50 J=1,NTRY
      KK=NTRY+1-J
      NSA(KK)=NSAVE(J)
      TI(KK)=TIHAT(J)
      TJ(KK)=TJHAT(J)
 50   CONTINUE
      DO 60 I=1,NTRY
      NSAVE(I)=NSA(I)
      TIHAT(I)=TI(I)
      TJHAT(I)=TJ(I)
60    CONTINUE
      WRITE(6,10)
10    FORMAT(10X,'SORTED SAVINGS')
      RETURN
      END
C*******************************************************
C*           SUBROUTINE INPT                           *
C*******************************************************
      SUBROUTINE INPT(IDDT)
      DIMENSION VDMAND(300),MEAN(300,300),VARS(300,300)
      DIMENSION MINE(300),VIRS(300)
      INTEGER TCAP,X,DDT,SST,Y,VDMAND,DMAND,TDMAND
      COMMON/A1/X(300),Y(300)
      COMMON/A2/NPT,NW,TCAP,MNP,NTRY
      COMMON/A5/DMAND(300),TDMAND(100)
      COMMON/A8/DIST(300,300)
      COMMON/A12/DDT,SST,IZAR
      COMMON/A14/MEAN,VARS,MINE,VIRS,VDMAND
      COMMON/A16/KPRO,GAMA
78    WRITE(6,1)
      WRITE(10,1)
1     FORMAT(5X,'--->',2X,'ENTER THE NUMBER OF STOP POINTS'/
    +15X,'INCLUDING THE TERMINAL AND TRUCK CAPACITY RESPECTIVELY')
      READ(5,*) NPT,TCAP
      WRITE(6,2) NPT,TCAP
      WRITE(10,2) NPT,TCAP
  2   FORMAT(5X,'NUMBER OF DEMAND POINTS=',I3//
    +5X,'CAPACITY OF TRUCK=',I5)
     WRITE(6,76)
     WRITE(10,76)
     READ(5,*) ICORR
     IF(ICORR.EQ.2) THEN
     WRITE(6,77)
```

```
      WRITE(10,77)
      GO TO 78
      ENDIF
      IF(IDDT.EQ.1) GO TO 13
      WRITE(6,3)
      WRITE(10,3)
 3    FORMAT(5X,'--->',5X,'ENTER THE EUCLIDIAN DISTANCE'/
     +18X,'FOR ALL STOP POINTS AND TERMINALS')
      WRITE(6,4)
      WRITE(10,4)
  4   FORMAT(5X,'--->',5X,'ENTER EUCLIDIAN DIST. FOR TERMINAL FIRST')
      DO 10 I=1,NPT
79    READ(5,*) X(I),Y(I)
      WRITE(6,16) X(I),Y(I)
      WRITE(10,16) X(I),Y(I)
      WRITE(6,76)
      READ(5,*) ICORR
      IF(ICORR.EQ.2) THEN
      WRITE(6,77)
      WRITE(10,77)
      GO TO 79
      ENDIF
 10   CONTINUE
C***
C**   WRITE THE EUCLIDIAN DISTANCE
C**
      WRITE(6,11)
      WRITE(10,11)
 11   FORMAT(10X,'EUCLIDIAN DISTANCE',//16X,'X',8X,'Y')
      DO 12  I=1,NPT
      WRITE(6,16) X(I),Y(I)
      WRITE(10,16) X(I),Y(I)
 16   FORMAT(14X,I4,8X,I4)
 12   CONTINUE
C****
C**  EVALUATE THE DISTANCE BETWEEN POINTS I AND J
C**
      DO 20 I=1,NPT
      DO 20 J=1,NPT
      IF(I.EQ.J) DIST(I,J)=0
      IF(I.GE.J) GO TO 20
      WW=FLOAT((X(I)-X(J))**2+(Y(I)-Y(J))**2)
      DIST(I,J)=SQRT(WW)
      DIST(J,I)=DIST(I,J)
 20   CONTINUE
 13   IF(IDDT.EQ.0) GO TO 95
```

```
      WRITE(6,19)
      WRITE(10,19)
 19   FORMAT(5X,'ENTER THE LINEAR DISTANCE BETWEEN THE POINTS')
      DO 35 I=1,NPT
      READ(5,*) (DIST(I,J),J=I,NPT)
 35   CONTINUE
      DO 36 I=2,NPT
      K=I-1
      DO 37 J=1,K
      DIST(I,J)=DIST(J,I)
 37   CONTINUE
 36   CONTINUE
 95   IF(DDT.EQ.1) THEN
      WRITE(6,7)
      WRITE(10,7)
 7    FORMAT(5X,'--->',5X,'ENTER THE CUSTOMER DEMANDS')
      DO 8  I=2,NPT
81    READ(5,*) DMAND(I)
      WRITE(6,26) I,DMAND(I)
      WRITE(10,26) I,DMAND(I)
      WRITE(6,76)
      WRITE(10,76)
      READ(5,*) ICORR
      IF(ICORR.EQ.2) THEN
      WRITE(6,77)
      WRITE(10,77)
      GO TO 81
      ENDIF
 8    CONTINUE
C****
C**    WRITE THE DEMANDS
C***
      WRITE(6,24)
      WRITE(10,24)
 24   FORMAT(2X,'DEMAND POINT',8X,'DEMAND')
      DO 75 I=2,NPT
      WRITE(6,26) I,DMAND(I)
      WRITE(10,26) I,DMAND(I)
 26   FORMAT(8X,I3,10X,I5)
 75   CONTINUE
      GO TO 111
      ENDIF
      IF(KPRO.EQ.1) THEN
      WRITE(6,142)
      WRITE(10,142)
142   FORMAT(//10X,'ENTER THE VALUE OF GAMA')
```

```
82      READ(5,*) GAMA
        WRITE(6,141) GAMA
        WRITE(10,141) GAMA
141     FORMAT(//20X,'GAMA=',5X,F10.4)
        WRITE(6,76)
        WRITE(10,76)
        READ(5,*) ICORR
        IF(ICORR.EQ.2) THEN
        WRITE(6,77)
        WRITE(10,77)
        GO TO 82
        ENDIF
        WRITE(6,112)
        WRITE(10,112)
 112    FORMAT(//10X,'---',2X,'ENTER MEAN AND VARIANCE OF DEMAND')
        DO 113  I=2,NPT
83      READ(5,*) DMAND(I),VDMAND(I)
        WRITE(6,116) I,DMAND(I),VDMAND(I)
        WRITE(10,116) I,DMAND(I),VDMAND(I)
        WRITE(6,76)
        WRITE(10,76)
        READ(5,*) ICORR
        IF(ICORR.EQ.2) THEN
        WRITE(6,77)
        WRITE(10,77)
        GO TO 83
        ENDIF
113     CONTINUE
C** WRITE THE DEMAND
        WRITE(6,114)
        WRITE(10,114)
 114    FORMAT(//2X,'DEMAND POINT',8X,'MEAN DEMAND',8X,'VAR DEMAND')
        DO 115 I=2,NPT
        WRITE(6,116) I,DMAND(I),VDMAND(I)
        WRITE(10,116) I,DMAND(I),VDMAND(I)
116     FORMAT(//11X,I2,12X,I5,12X,I5)
115     CONTINUE
        ENDIF
111     WRITE(6,38)
        WRITE(10,38)
 38     FORMAT(10X,'DISTANCE')
        ITDD=0
        DO 41 I=2,NPT
        ITDD=ITDD+DMAND(I)
 41     CONTINUE
        NW=(ITDD/TCAP)+20
```

```
      NNW=NW-20
      MNP=(NPT/NNW)+1
      MNP=MNP+7
76    FORMAT(/2X,'CORRECT',2X,'ENTER',1X,'1::YES',2X,'2:NO')
77    FORMAT(/2X,'REENTER AGAIN')
      RETURN
      END
C************************************************
C*              SUBROUTINE HEAPSN               *
C************************************************
      SUBROUTINE HEAPSN(XX,POS,PPOSS,N)
      INTEGER XX(5000),POS(5000),PPOSS(5000)
      N2=N/2
      DO 10 J=1,N2
      I=N2+1-J
10    CALL PUSHDN (XX,POS,PPOSS,I,N)
      N1=N-1
      DO 20 JJ=1,N1
      I=N1+1-JJ
      CALL SWAPN(XX(1),XX(I+1),POS(1),POS(I+1),PPOSS(1),
     1 PPOSS(I+1))
20    CALL PUSHDN(XX,POS,PPOSS,1,I)
      RETURN
      END
C***********************************************
C*          SUBROUTINE SWAPN                   *
C*******************************************
C
      SUBROUTINE SWAPN(I,J,P,Q,R,S)
      INTEGER P,Q,R,S
      K=I
      I=J
      J=K
      Z=P
      P=Q
      Q=Z
      T=R
      R=S
      S=T
      RETURN
      END
C**********************************************
C*            SUBROUTINE PUSHDN               *
C**********************************************
      SUBROUTINE PUSHDN(XX,POS,PPOSS,I,N)
      INTEGER XX(5000),POS(5000),PPOSS(5000)
```

```
      LOGICAL FIN
      FIN=.FALSE.
      K=XX(I)
      Z=POS(I)
      T=PPOSS(I)
      J=I*2
10    CONTINUE
      IF(J.LE.N.AND..NOT.FIN) THEN
      IVV=J+1
      IF(IVV.LE.N) THEN
      IF(J.LT.N.AND.XX(J).LT.XX(J+1)) J=J+1
      ENDIF
      IF(K.GE.XX(J)) THEN
      FIN=.TRUE.
      ELSE
      XX(J/2)=XX(J)
      POS(J/2)=POS(J)
      PPOSS(J/2)=PPOSS(J)
      J=J*2
      ENDIF
      XX(J/2)=K
      POS(J/2)=Z
      PPOSS(J/2)=T
      GO TO 10
      ENDIF
      RETURN
      END
C**********************************************
C*          SUBROUTINE SAVMAT                *
C**********************************************
C*
      SUBROUTINE SAVMAT
C**  THIS SUBROUTINE  CONSTRUCT THE SAVING MATRIX AND THE
C**  INITIAL SOLUTION TO THE PROBLEM
      DIMENSION ISAVE(300,300)
      INTEGER TCAP,X,P,TIHAT,TJHAT,ROUTE
      COMMON/A2/NPT,NW,TCAP,MNP,NTRY
      COMMON/A3/MSVA(5000),NSAVE(5000),XX(5000)
      COMMON/A4/NB(100),NF(100),NR(100),P
      COMMON/A9/TIHAT(5000),TJHAT(5000),ROUTE(100,100)
      COMMON/A8/DIST(300,300)
      NTRY=0
      DO 60 I=2,NPT
      DO 60 J=I,NPT
      ISAVE(I,J)=-99999
      IF(DIST(I,J).EQ.0) GO TO 70
```

```
       ISAVE(I,J)=DIST(I,1)+DIST(1,J)-DIST(I,J)
 70    ISAVE(J,I)=ISAVE(I,J)
 60    CONTINUE
       DO 80 I=2,NPT
       ISAVE(I,1)=-99999
       ISAVE(1,I)=ISAVE(I,1)
 80    CONTINUE
C**
C**  CONSTRUCT THE INITIAL SOLUTION OR INITIAL ROUTE BY
C**  THE FOLLOWING MATRIX.PUT THE ARRAY ISAVE INTO THE
C**  NEW ARRAY NSAVE WHICH IS ONE DIMENSIONAL.
       L=0
       IPT=NPT-1
       DO 100 I=2,IPT
       K=I+1
       DO 100 J=K,NPT
       IF(ISAVE(I,J).LE.0) GO TO 100
       L=L+1
       NSAVE(L)=ISAVE(I,J)
       TIHAT(L)=I
       TJHAT(L)=J
 100   CONTINUE
       NTRY=L
       DO 170 I=1,NW
       ROUTE(I,1)=I
       ROUTE(I,2)=1
       ROUTE(I,MNP)=1
       MNP1=MNP-1
       DO 180 J=3,MNP1
 180   ROUTE(I,J)=0
       NB(I)=0
       NF(I)=0
       NR(I)=2
170    CONTINUE
       DO 13 I=1,NW
       WRITE(6,23) (ROUTE(I,J),J=1,MNP)
       WRITE(10,23) (ROUTE(I,J),J=1,MNP)
 23    FORMAT(5X,30(I2,2X))
 13    CONTINUE
       RETURN
       END
C*********************************************
C*           SUBROUTINE RTCONT              *
C*********************************************
C
       SUBROUTINE RTCONT(PP,T)
```

```
      DIMENSION VARS(300,300),MINE(300),MEAN(300,300)
      DIMENSION VIRS(300),VDMAND(300)
      INTEGER P,T,R,PP,ROUTE,TIHAT,TJHAT,DMAND,TDMAND
      INTEGER DDT,SST,VARS,VIRS,VDMAND
      COMMON/A4/NB(100),NF(100),NR(100),P
      COMMON/A5/DMAND(300),TDMAND(100)
      COMMON/A6/LI,LJ,LI1,LI2,LJ1,LJ2,LRI,LRJ
      COMMON/A9/TIHAT(5000),TJHAT(5000),ROUTE(100,100)
      COMMON/A12/DDT,SST,IZAR
      COMMON/A14/MEAN,MINE,VARS,VIRS,VDMAND
      COMMON/A15/IEE,IFF,DELTA,IALGOL,BKAMA
      COMMON/A16/KPRO,GAMA
      COMMON/A17/KDMAND,KTULOD,KTTRVL
C**   LRI= INDICATES ROUTE  LRI
C**   LRJ=INDICATES ROUTE LRJ
C**   LI1 =INDICATES THAT TIHAT(T) IS EQUAL TO THE NB(KPP)
C**   LI2 = INDICATES THAT TIHAT(T) IS EQUAL TO THE NF(KPP)
C**   LJ1=INDICATES THAT TJHAT(T) IS EQUAL TO THE NB(KPP)
C**   LJ2= INDICATES THAT TJHAT(T) IS EQUAL TO THE NF(KPP)
      LRI=0
      LRJ=0
      LI1=0
      LI2=0
      LJ1=0
      LJ2=0
      LI=0
      LJ=0
      KPP=PP
      DO 10 KPP=1,PP
      IF(NB(KPP).EQ.TIHAT(T).OR.NF(KPP).EQ.TIHAT(T)) THEN
      LI=1
      LRI=KPP
      IF(NB(KPP).EQ.TIHAT(T)) LI1=1
      IF(NF(KPP).EQ.TIHAT(T)) LI2=1
      IF(NB(KPP).EQ.TJHAT(T).OR.NF(KPP).EQ.TJHAT(T)) RETURN
      ENDIF
      IF(NB(KPP).EQ.TJHAT(T).OR.NF(KPP).EQ.TJHAT(T) ) THEN
      LJ=1
      LRJ=KPP
      IF(NB(KPP).EQ.TJHAT(T)) LJ1=1
      IF(NF(KPP).EQ.TJHAT(T)) LJ2=1
      IF(NB(KPP).EQ.TIHAT(T).OR.NF(KPP).EQ.TIHAT(T)) RETURN
      ENDIF
 10   CONTINUE
     IF(LI.EQ.0.AND.LJ.EQ.0) THEN
      P=PP+1
```

```
        R=3
        ROUTE(P,R)=TIHAT(T)
        NB(P)=ROUTE(P.R)
        R=R+1
        ROUTE(P,R)=TJHAT(T)
        NF(P)=ROUTE(P,R)
        PP=P
        NR(P)=R
        RETURN
        ENDIF
        IF(LI.EQ.1.AND.LJ.EQ.1) THEN
        CALL COMBRT(IVB,IWB,IXBB,IYBB,PP,T)
        IF(IVB.EQ.O.AND.IWB.EQ.O.AND.IXBB.EQ.O.AND.IYBB.EQ.O)
     +  RETURN
        CALL SWTCH(IVB,IWB,IXBB,IYBB,PP)
        RETURN
        ENDIF
C* TO COMBINE TWO ROUTES TOGETHER
        IF(LI.EQ.1.AND.LJ.EQ.O)  THEN
        CALL COMBND(T)
        RETURN
        ENDIF
C* TO ADD A NODE INTO AN EXISTING ROUTE
        IF(LI.EQ.O.AND.LJ.EQ.1) THEN
        CALL COMBND(T)
        ENDIF
        RETURN
        END
C*************************************************
C*        SUBROUTINE INTR                        *
C*************************************************
C*
          SUBROUTINE INTR(IN,PP,T,IYOUTH)
          INTEGER PP,TCAP,TIHAT,TJHAT,ROUTE,T
          COMMON/A2/NPT,NW.TCAP,MNP,NTRY
          COMMON/A4/NB(100),NF(100),NR(100),P
          COMMON/A9/TIHAT(5000),TJHAT(5000),ROUTE(100,100)
          GO TO (1,2),IYOUTH
1         IN=O
          DO 20 I=1,PP
          NNR=NR(I)-1
          IF(NNR.LT.4) GO TO 20
          DO 10 J=4,NNR
          IF(ROUTE(I,J).EQ.TIHAT(T).OR.ROUTE(I,J).EQ.TJHAT(T)) IN=1
 10      CONTINUE
 20      CONTINUE
```

```
         RETURN
 2       ICANCL=0
         DO 30 I=2,NPT
         DO 40 J=1,PP
         MNOPP=NR(J)
         DO 50 K=3,MNOPP
         IF(I.EQ.ROUTE(J,K)) GO TO 30
50       CONTINUE
40       CONTINUE
         ICANCL=I
         P=PP+1
         R=3
         ROUTE(P,R)=ICANCL
         NB(P)=ROUTE(P,R)
         NF(P)=ROUTE(P,R)
         NR(P)=R
         PP=P
30       CONTINUE
         RETURN
         END
C*************************************************
C*              SUBROUTINE COMBND               *
C*************************************************
C**
         SUBROUTINE COMBND(T)
         INTEGER TCAP,P,DMAND,TIHAT,TJHAT,ROUTE,FLI,FLJ
         INTEGER FLIJ,FIJ,T,PP,DDT,SST
         COMMON/A2/NPT,NW,TCAP,MNP,NTRY
         COMMON/A4/NB(100),NF(100),NR(100),P
         COMMON/A5/DMAND(300),TDMAND(100)
         COMMON/A6/LI,LJ,LI1,LI2,LJ1,LJ2,LRI,LRJ
         COMMON/A9/TIHAT(5000),TJHAT(5000),ROUTE(100,100)
         COMMON/A12/DDT,SST,IZAR
         COMMON/A15/IEE,IFF,DELTA,IALGOL,BKAMA
         COMMON/A16/KPRO,GAMA
         COMMON/A17/ KDMAND,KTULOD,KTTRVL
         IF(LI.EQ.1) THEN
         IF(LI1.EQ.1) THEN
         IF(DDT.EQ 1) THEN
         IX=1
         CALL FEASBL(IX,FLI,FLJ,FLIJ,FIJ,T,IZX)
         GO TO 5
         ENDIF
         IF(KPRO.EQ.1) THEN
         IX=1
         CALL FSBL(IX,FLI,FLJ,FLIJ,FIJ,T,IZX,KDMAND)
```

```
          GO TO 5
          ENDIF
          IF(SST.EQ 1.AND.IEE.EQ.1) THEN
          IX=1
          CALL STFSBL(IX,FLI,FLJ,FLIJ,FIJ,T,IZX,KDMAND,KTULOD,KTTRVL)
          GO TO 5
          ENDIF
          IF(SST.EQ.1.AND.IFF.EQ.1) THEN
          IX=1
          CALL FSBL(IX,FLI,FLJ,FLIJ,FIJ,T,IZX,KDMAND)
          GO TO 5
          ENDIF
5         IF(FLI.EQ.1) THEN
C**    ADD A NODE IN FRONT OF ROUTE LRI
          LOC=4
          NOV1=NR(LRI)+1
          NVV=NOV1+1
          NVZ=NVV-LOC
          DO 10 K=1,NVZ
          I=NVV-K
          J=I-1
 10       ROUTE(LRI,I)=ROUTE(LRI,J)
          ROUTE(LRI,3)=TJHAT(T)
          NF(LRI)=ROUTE(LRI,NOV1)
          NB(LRI)=TJHAT(T)
          NR(LRI)=NOV1
          ENDIF
          RETURN
          ENDIF
          IF(LI2.EQ.1) THEN
          IF(DDT.EQ.1) THEN
          IX=1
          CALL FEASBL(IX,FLI,FLJ,FLIJ,FIJ,T,IZX)
          GO TO 15
          ENDIF
          IF(KPRO.EQ.1) THEN
          IX=1
          CALL FSBL(IX,FLI,FLJ,FLIJ,FIJ,T,IZX,KDMAND)
          GO TO 15
          ENDIF
          IF(SST.EQ.1.AND.IEE.EQ.1) THEN
          IX=1
C * PURPOSE TO CHECK THE FEASIBILITY OF VEHICLE ROUTES
          CALL STFSBL(IX,FLI,FLJ,FLIJ,FIJ,T,IZX,KDMAND,KTULOD,KTTRVL)
           GO TO 15
           ENDIF
```

```
          IF(SST.EQ.1.AND.IFF.EQ.1) THEN
          IX=1
          CALL FSBL(IX,FLI,FLJ,FLIJ,FIJ,T,IZX,KDMAND)
          GO TO 15
          ENDIF
15        IF(FLI.EQ.1)  THEN
C** ADD A NODE AT THE END OF ROUTE OF LRI
          LOC=NR(LRI)+1
          ROUTE(LRI,LOC)=TJHAT(T)
          NR(LRI)=LOC
          NF(LRI)=TJHAT(T)
          ENDIF
          ENDIF
          RETURN
          ENDIF
          IF(LJ.EQ.1) THEN
          IF(LJ1.EQ.1) THEN
          IF(DDT.EQ.1) THEN
          IX=2
          CALL FEASBL(IX,FLI,FLJ,FLIJ,FIJ,T,IZX)
          GO TO 20
          ENDIF
          IF(KPRO.EQ.1) THEN
          IX=2
          CALL FSBL(IX,FLI,FLJ,FLIJ,FIJ,T,IZX,KDMAND)
          GO TO 20
          ENDIF
          IF(SST.EQ.1.AND.IEE.EQ.1) THEN
          IX=2
          CALL STFSBL(IX,FLI,FLJ,FLIJ,FIJ,T,IZX,KDMAND,KTULOD,KTTRVL)
          GO TO 20
          ENDIF
          IF(SST.EQ.1.AND.IFF.EQ.1) THEN
          IX=2
          CALL FSBL(IX,FLI,FLJ,FLIJ,FIJ,T,IZX,KDMAND)
          GO TO 20
          ENDIF
20        IF(FLJ.EQ.1) THEN
C**    ADD A NODE IN THE FRONT OF ROUTE LRJ
          LOC=4
          NOV2=NR(LRJ)+1
          NWW=NOV2+1
          NWZ=NWW-LOC
          DO 30 K=1,NWZ
          I=NWW-K
          J=I-1
```

```
30        ROUTE(LRJ,I)=ROUTE(LRJ,J)
          ROUTE(LRJ,3)=TIHAT(T)
          NF(LRJ)=ROUTE(LRJ,NOV2)
          NR(LRJ)=NOV2
          NB(LRJ)=TIHAT(T)
          ENDIF
          RETURN
          ENDIF
          IF(LJ2.EQ.1) THEN
          IF(DDT.EQ.1) THEN
          IX=2
          CALL FEASBL(IX,FLI,FLJ,FLIJ,FIJ,T,IZX)
          GO TO 25
          ENDIF
          IF(KPRO.EQ.1) THEN
          IX=2
          CALL FSBL(IX,FLI,FLJ,FLIJ,FIJ,T,IZX,KDMAND)
          GO TO 25
          ENDIF
          IF(SST.EQ.1.AND.IEE.EQ.1) THEN
          IX=2
          CALL STFSBL(IX,FLI,FLJ,FLIJ,FIJ,T,IZX,KDMAND,KTULOD,KTTRVL)
          GO TO 25
          ENDIF
          IF(SST.EQ.1.AND.IFF.EQ.1) THEN
          IX=2
          CALL FSBL(IX,FLI,FLJ,FLIJ,FIJ,T,IZX,KDMAND)
          GO TO 25
          ENDIF
25        IF(FLJ.EQ.1) THEN
C** ADD A NODE AT THE END OF ROUTE LRJ
          LOC=NR(LRJ)+1
          ROUTE(LRJ,LOC)=TIHAT(T)
          NR(LRJ)=LOC
C** NODE NB DOES NOT CHANGE
          NF(LRJ)=TIHAT(T)
          ENDIF
          ENDIF
          ENDIF
          RETURN
          END
C*****************************************************
C*             SUBROUTINE COMBRT                     *
C*****************************************************
        SUBROUTINE COMBRT(IVB,IWB,IXBB,IYBB,PP,T)
        DIMENSION TULOAD(100),TTRAVL(100)
```

```
      INTEGER TIHAT,TJHAT,ROUTE,P,FLI,FLJ,FLIJ,FIJ
      INTEGER DMAND,TDMAND,PP,T,DDT,SST
      COMMON/A4/NB(100),NF(100),NR(100),P
      COMMON/A5/DMAND(300),TDMAND(100)
      COMMON/A6/LI,LJ,LI1,LI2,LJ1,LJ2,LRI,LRJ
      COMMON/A9/TIHAT(5000),TJHAT(5000),ROUTE(100,100)
      COMMON/A12/DDT,SST,IZAR
      COMMON/A15/IEE,IFF,DELTA,IALGOL,BKAMA
      COMMON/A16/KPRO,GAMA
      COMMON/A17/KDMAND,KTULOD,KTTRVL
      IVB=0
      IWB=0
      IXBB=0
      IYBB=0
      IF(LI2.EQ.1.AND.LJ2.EQ.1) THEN
      IF(DDT.EQ.1) THEN
      IX=3
      CALL FEASBL (IX,FLI,FLJ,FLIJ,FIJ,T,IZX)
      GO TO 41
      ENDIF
      IF(KPRO.EQ.1) THEN
      IX=3
      CALL FSBL(IX,FLI,FLJ,FLIJ,FIJ,T,IZX,KDMAND)
      GO TO 41
      ENDIF
      IF(SST.EQ.1.AND.IEE.EQ.1) THEN
      IX=3
      CALL STFSBL(IX,FLI,FLJ,FLIJ,FIJ,T,IZX,KDMAND,KTULOD,KTTRVL)
      GO TO 41
      ENDIF
      IF(SST.EQ.1.AND.IFF.EQ.1) THEN
      IX=3
      CALL FSBL(IX,FLI,FLJ,FLIJ,FIJ,T,IZX,KDMAND)
      GO TO 41
      ENDIF
CC*   FOR CHECK
C**
41    IF(FLIJ.EQ.1) THEN
      IXBB=1
C** PUT ROUTE LRJ IN ROUTE LRI
      LC=NR(LRI)+1
      LD=NR(LRJ)-2
      LB=LC+LD-1
      J=NR(LRJ)
      NF(LRI)=NB(LRJ)
      DO 10 I=LC,LB
```

```
      ROUTE(LRI,I)=ROUTE(LRJ,J)
      ROUTE(LRJ,J)=0
      J=J-1
 10   CONTINUE
C**   NB(LRI) DOES NOT CHANGE.
      NR(LRI)=LB
      NR(LRJ)=2
      NB(LRJ)=0
      NF(LRJ)=0
      ELSE
      RETURN
      ENDIF
      IF(DDT.EQ.1) THEN
      TDMAND(LRI)=TDMAND(LRI)+TDMAND(LRJ)
      TDMAND(LRJ)=0
      RETURN
      ENDIF
      IF(KPRO.EQ.1) THEN
      TDMAND(LRI)=KDMAND
      TDMAND(LRJ)=0
      RETURN
      ENDIF
      IF(SST.EQ.1.AND.IEE.EQ.1) THEN
      TDMAND(LRI)=KDMAND
      TDMAND(LRJ)=0
      TULOAD(LRI)=KTULOD
      TULOAD(LRJ)=0
      TTRAVL(LRI)=KTTRVL
      TTRAVL(LRJ)=0
      RETURN
      ENDIF
      IF(SST.EQ.1.AND.IFF.EQ.1) THEN
      TDMAND(LRI)=KDMAND
      TDMAND(LRJ)=0
      ENDIF
      RETURN
      ENDIF
      IF(LI2.EQ.1.AND.LJ1.EQ 1) THEN
      IF(DDT EQ.1) THEN
      IX=3
      CALL FEASBL(IX,FLI,FLJ,FLIJ,FIJ,T,IZX)
      GO TO 42
      ENDIF
      IF(KPRO.EQ.1) THEN
      IX=3
      CALL FSBL(IX,FLI,FLJ,FLIJ,FIJ,T,IZX,KDMAND)
```

```
      GO TO 42
      ENDIF
      IF(SST.EQ.1.AND.IEE.EQ.1) THEN
      IX=3
      CALL STFSBL(IX,FLI,FLJ,FLIJ,FIJ,T,IZX,KDMAND,KTULOD,KTTRVL)
      GO TO 42
      ENDIF
C**
      IF(SST.EQ.1.AND.IFF.EQ.1) THEN
      IX=3
       CALL FSBL(IX,FLI,FLJ,FLIJ,FIJ,T,IZX,KDMAND)
      GO TO 42
      ENDIF
42    IF(FLIJ.EQ.1) THEN
      IVB=1
      LC=NR(LRI)+1
      LD=NR(LRJ)-2
      LB=LD+LC-1
      J=3
     DO 15 I=LC,LB
      ROUTE(LRI,I)=ROUTE(LRJ,J)
      ROUTE(LRJ,J)=0
      J=J+1
 15   CONTINUE
      NF(LRI)=NF(LRJ)
      NR(LRI)=LB
      NR(LRJ)=2
      NB(LRJ)=0
      NF(LRJ)=0
      ELSE
      RETURN
      ENDIF
      IF(DDT.EQ.1) THEN
      TDMAND(LRI)=TDMAND(LRI)+TDMAND(LRJ)
      TDMAND(LRJ)=0
      RETURN
      ENDIF
      IF(KPRO.EQ.1) THEN
      TDMAND(LRI)=KDMAND
      TDMAND(LRJ)=0
      RETURN
      ENDIF
      IF(SST.EQ.1.AND.IEE.EQ.1) THEN
      TDMAND(LRI)=KDMAND
      TDMAND(LRJ)=0
      TULOAD(LRI)=KTULOD
```

```
      TULOAD(LRJ)=0
      TTRAVL(LRI)=KTTRVL
      TTRAVL(LRJ)=0
      RETURN
      ENDIF
      IF(SST.EQ.1.AND.IFF.EQ.1) THEN
      TDMAND(LRI)=KDMAND
      TDMAND(LRJ)=0
      ENDIF
      RETURN
      ENDIF
C
      IF(LI1.EQ.1.AND.LJ2.EQ.1) THEN
      IF(DDT.EQ.1) THEN
      IX=4
      CALL FEASBL(IX,FLI,FLJ,FLIJ,FIJ,T,IZX)
      GO TO 43
      ENDIF
      IF(KPRO.EQ.1) THEN
      IX=4
      CALL FSBL(IX,FLI,FLJ,FLIJ,FIJ,T,IZX,KDMAND)
      GO TO 43
      ENDIF
      IF(SST.EQ.1.AND.IEE.EQ.1) THEN
      IX=4
      CALL STFSBL(IX,FLI,FLJ,FLIJ,FIJ,T,IZX,KDMAND,KTULOD,KTTRVL)
      GO TO 43
      ENDIF
      IF(SST.EQ.1.AND.IFF.EQ.1) THEN
      IX=4
      CALL FSBL(IX,FLI,FLJ,FLIJ,FIJ,T,IZX,KDMAND)
      GO TO 43
      ENDIF
43    IF(FIJ.EQ.1) THEN
      IWB=1
      LC=NR(LRJ)+1
      LD=NR(LRI)-2
      LB=LD+LC-1
      J=3
      DO 30 I=LC,LB
      ROUTE(LRJ,I)=ROUTE(LRI,J)
      ROUTE(LRI,J)=0
      J=J+1
 30   CONTINUE
      NF(LRJ)=NF(LRI)
C** NB(LRJ) DOES NOT CHANGE.
```

```
      NR(LRJ)=LB
      NR(LRI)=2
      NB(LRI)=0
      NF(LRI)=0
      ELSE
      RETURN
      ENDIF
      IF(DDT.EQ.1) THEN
      TDMAND(LRJ)=TDMAND(LRJ)+TDMAND(LRI)
      TDMAND(LRI)=0
      RETURN
      ENDIF
      IF(KPRO.EQ.1) THEN
      TDMAND(LRJ)=KDMAND
      TDMAND(LRI)=0
      RETURN
      ENDIF
      IF(SST.EQ.1.AND.IEE.EQ.1) THEN
      TDMAND(LRJ)=KDMAND
      TDMAND(LRI)=0
      TULOAD(LRJ)=KTULOD
      TULOAD(LRI)=0
      TTRAVL(LRJ)=KTTRVL
      TTRAVL(LRI)=0
      RETURN
      ENDIF
      IF(SST.EQ.1.AND.IFF.EQ.1) THEN
      TDMAND(LRJ)=KDMAND
      TDMAND(LRI)=0
      ENDIF
      ENDIF
      RETURN
      END
C*****************************************************
C*          SUBROUTINE FEASBL                       *
C*****************************************************
C*
      SUBROUTINE FEASBL(IX,FLI,FLJ,FLIJ,FIJ,T,IZX)
      INTEGER TCAP,DMAND,TIHAT,TJHAT,ROUTE,FLI,FLJ,FLIJ,FIJ
      INTEGER TDMAND,P,T,PP
      COMMON/A2/NPT,NW,TCAP,MNP,NTRY
      COMMON/A4/NB(100),NF(100),NR(100),P
      COMMON/A5/DMAND(300),TDMAND(100)
      COMMON/A6/LI,LJ,LI1,LI2,LJ1,LJ2,LRI,LRJ
      COMMON/A9/TIHAT(5000),TJHAT(5000),ROUTE(100,100)
      GO TO (1,2,3,4,5),IX
```

```
1       IYZ=1
        ISET=1
        CALL CHCKK(LRI,IYZ,ISET)
        JTJ=TJHAT(T)
        KDMAND=TDMAND(LRI)+DMAND(JTJ)
        IF(KDMAND.LE.TCAP) THEN
        FLI=1
        ELSE
        FLI=0
        ENDIF
        RETURN
 2      IYZ=1
        ISET=1
        CALL CHCKK(LRJ,IYZ,ISET)
        ITI=TIHAT(T)
        KDMAND=TDMAND(LRJ)+DMAND(ITI)
        IF(KDMAND.LE.TCAP) THEN
        FLJ=1
        ELSE
        FLJ=0
        ENDIF
        RETURN
 3      IYZ=1
        ISET=1
        CALL CHCKK(LRI,IYZ,ISET)
        IYZ=1
        CALL CHCKK(LRJ,IYZ,ISET)
        KDMAND=TDMAND(LRI)+TDMAND(LRJ)
        IF(KDMAND.LE.TCAP) THEN
        FLIJ=1
        ELSE
        FLIJ=0
        ENDIF
        RETURN
4       IYZ=1
        ISET=1
        CALL CHCKK(LRI,IYZ,ISET)
        ISET=1
        IYZ=1
        CALL CHCKK(LRJ,IYZ,ISET)
        KDMAND=TDMAND(LRI)+TDMAND(LRJ)
        IF(KDMAND.LE.TCAP)  THEN
        FIJ=1
        ELSE
        FIJ=0
        ENDIF
```

```
      RETURN
5     ITI=TIHAT(T)
      JTJ=TJHAT(T)
      TDMAND(P)=TDMAND(P)+DMAND(ITI)+DMAND(JTJ)
      IZX=0
      IF(TDMAND(P).LE.TCAP) IZX=1
      RETURN
      END
C***********************************************
C*          SUBROUTINE CTD                     *
C***********************************************
        SUBROUTINE CTD(PP,DD,IDD)
        INTEGER PP,P,ROUTE,TIHAT,TJHAT
        REAL IDD
        DIMENSION IDD(100)
        COMMON/A4/NB(100),NF(100),NR(100),P
        COMMON/A8/DIST(300,300)
        COMMON/A9/TIHAT(5000),TJHAT(5000),ROUTE(100,100)
        DD=0
        DO 10 I=1,PP
        IDD(I)=0
        NN=NR(I)
        DO 20 J=2,NN
        K=ROUTE(I,J)
        L=ROUTE(I,J+1)
        LL=J+1
        IF(LL.GT.NN) THEN
        IDD(I)=IDD(I)+DIST(K,1)
        ELSE
        IDD(I)=IDD(I)+DIST(K,L)
        ENDIF
 20     CONTINUE
        DD=DD+IDD(I)
 10     CONTINUE
        RETURN
        END
C***********************************************
C*          SUBROUTINE CHCKK                   *
C***********************************************
C
        SUBROUTINE CHCKK(PP,IYZ,ISET)
        INTEGER DMAND,TDMAND,PP,ROUTE,TIHAT,TJHAT,P
        REAL IDD
        DIMENSION IDD(100)
        COMMON/A4/NB(100),NF(100),NR(100),P
        COMMON/A5/DMAND(300),TDMAND(100)
```

```
          COMMON/A9/TIHAT(5000),TJHAT(5000),ROUTE(100,100)
          GO TO (1,2), IYZ
1         NN=NR(PP)
          TDMAND(PP)=0
          DO 20 J=3,NN
          K=ROUTE(PP,J)
 20       TDMAND(PP)=TDMAND(PP)+DMAND(K)
          RETURN
C**
 2        DO 40 I=ISET,PP
          TDMAND(I)=0
          NN=NR(I)
          DO 30 J=3,NN
          K=ROUTE(I,J)
          IF(K.EQ.0) GO TO 40
 30       TDMAND(I)=TDMAND(I)+DMAND(K)
 40       CONTINUE
          RETURN
          END
C**********************************************
C*             SUBROUTINE WWRT                *
C**********************************************
C**
         SUBROUTINE WWRT(PP)
         DIMENSION MOLE(100),IDD(100),TULOAD(100),TTRAVL(100)
         INTEGER TULOAD,TTRAVL,DDT,SST,TULDD,TTDD
         INTEGER P,ROUTE,TDMAND,DMAND,TIHAT,TJHAT,PP
         REAL IDD
         COMMON/A2/NPT,NW,TCAP,MNP,NTRY
         COMMON/A4/NB(100),NF(100),NR(100),P
         COMMON/A5/DMAND(300),TDMAND(100)
         COMMON/A9/TIHAT(5000),TJHAT(5000),ROUTE(100,100)
         COMMON/A12/DDT,SST,IZAR
         COMMON/A13/TULOAD,TTRAVL
         COMMON/A15/IEE,IFF,DELTA,IALGOL,BKAMA
         COMMON/A16/KPRO,GAMA
         WRITE(6,10)
         WRITE(10,10)
 10      FORMAT(5X,'THE ROUTES ARE THE FOLLOWINGS')
         MNK=MNP-1
         DO 20 I=1,PP
        WRITE(6,30) I,(ROUTE(I,J),J=2,MNK),ROUTE(I,MNP)
        WRITE(10,30) I,(ROUTE(I,J),J=2,MNK),ROUTE(I,MNP)
30      FORMAT(//5X,'ROUTE ',2X,I3,2X,'-->',2X,30(1X,I2))
20      CONTINUE
        IF(DDT.EQ.1.OR.IEE.EQ.1.OR.KPRO.EQ.1) THEN
```

```
      CALL CTD(PP,DD,IDD)
      ENDIF
      IF(IFF.EQ.1) THEN
      CALL STCTD(PP,DD)
      WRITE(6,41) DD
      WRITE(10,41) DD
 41   FORMAT(//30X,'TOTAL ELAPSE TIME OF WHOLE SYSTEM IS',2X,F10.4)
      GO TO 42
      ENDIF
      KKDD=DD
      WRITE(6,40) KKDD
      WRITE(10,40) KKDD
 40   FORMAT(10X,'TOTAL DISTANCE IS',1X,I8)
      DO 50 I=1,PP
      MOLE(I)=IDD(I)
      WRITE(6,60) I,MOLE(I)
      WRITE(10,60) I,MOLE(I)
60    FORMAT(//5X,'DISTANCE ROUTE',2X,I3,'IS',I6)
50    CONTINUE
42    IYZ=2
      ISET=1
      IF(DDT.EQ.1) THEN
      CALL CHCKK(PP,IYZ,ISET)
      ENDIF
      IF(KPRO.EQ.1) THEN
      CALL FCHECK(PP,IYZ,ISET)
      GO TO 99
      ENDIF
      IF(SST.EQ.1) THEN
      CALL PRCHCK(PP,IYZ,ISET)
      TULDD=0
      TTDD=0
      WRITE(6,51)
      WRITE(10,51)
 51   FORMAT(//30X,'TOTAL UNLOAD TIME OF EACH ROUTE')
      DO 52 I=1,PP
      TULDD=TULDD+TULOAD(I)
      WRITE(6,53) I ,TULOAD(I)
      WRITE(10,53) I,TULOAD(I)
 53   FORMAT(//30X,'UNLOAD TIME',2X,I3,2X,'IS',2X,I8)
 52   CONTINUE
      WRITE(6,101) TULDD
      WRITE(10,101) TULDD
101   FORMAT(//30X,'TOTAL UNLOAD TIME OF WHOLE SYSTEM=',2X,I6)
      WRITE(6,54)
      WRITE(10,54)
```

```
 54     FORMAT(//30X,'TOTAL TRAVEL TIME OF EACH ROUTE')
        DO 55 I=1,PP
        TTDD=TTDD+TTRAVL(I)
        WRITE(6,56) I,TTRAVL(I)
        WRITE(10,56) I,TTRAVL(I)
 56     FORMAT(//30X,'TRAVLING TIME ',2X,I3,2X,'IS',2X,I8)
 55     CONTINUE
        WRITE(6,102) TTDD
        WRITE(10,102) TTDD
102     FORMAT(//30X,'TOTAL TRAVEL TIME OF WHOLE SYSTEM=',2X,I6)
        ENDIF
99      WRITE(6,70)
        WRITE(10,70)
70      FORMAT(//20X,'TOTAL DEMAND OF EACH ROUTE')
        DO 80 I=1,PP
        WRITE(6,90) I,TDMAND(I)
        WRITE(10,90) I,TDMAND(I)
90      FORMAT(30X,'DEMAND ROUTE',2X,I3,'IS',I8)
80      CONTINUE
        K=PP
        WRITE(6,100) K
        WRITE(10,100) K
 100    FORMAT(//30X,'NUMBER OF THE REQUIRED VEHICLES',2X,'IS',
     1  2X,I2)
        RETURN
        END
C************************************************
C*              SUBROUTINE SWTCH                *
C************************************************
C**
          SUBROUTINE SWTCH(IVB,IWB,IXBB,IYBB,PP)
          DIMENSION KROOT(100,100),KNR(100),KNF(100),KNB(100)
          INTEGER P,T,R,PP,ROUTE,TIHAT,TJHAT,DMAND,TDMAND
          INTEGER DDT,SST
          COMMON/A4/NB(100),NF(100),NR(100),P
          COMMON/A5/DMAND(300),TDMAND(100)
          COMMON/A6/LI,LJ,LI1,LI2,LJ1,LJ2,LRI,LRJ
          COMMON/A9/TIHAT(5000),TJHAT(5000),ROUTE(100,100)
          COMMON/A12/DDT,SST,IZAR
          COMMON/A16/KPRO,GAMA
          DO 5 I=1,PP
          DO 15 J=1 ,20
C**  VALUE 20 IN THE ABOVE DO IS STANDING FOR MNP
  15      KROOT(I,J)=0
          KNR(I)=2
          KNB(I)=0
```

```
5        KNF(I)=0
         MEET=0
         DO 10 I=1,PP
         IF(IXBB.EQ.1.AND.I.EQ.LRJ) GO TO 10
         IF(IYBB.EQ.1.AND.I.EQ.LRJ) GO TO 10
         IF(IVB.EQ.1.AND.I.EQ.LRJ) GO TO 10
         IF(IWB.EQ.1.AND.I.EQ.LRI) GO TO 10
         NRI= NR(I)
         MEET=MEET+1
         DO 20 J=3,NRI
20       KROOT(MEET,J)=ROUTE(I,J)
         KNR(MEET)=NR(I)
         KNF(MEET)=NF(I)
         KNB(MEET)=NB(I)
10       CONTINUE
         DO 50 I=1,PP
         NRI=NR(I)
         DO 60 J=3,NRI
60       ROUTE(I,J)=0
         NR(I)=2
         NF(I)=0
         NB(I)=0
50       CONTINUE
         PP=MEET
         P=MEET
         DO 30 I=1,MEET
         KNRI=KNR(I)
         DO 40 J=3,KNRI
40       ROUTE(I,J)=KROOT(I,J)
         NR(I)=KNR(I)
         NB(I)=KNB(I)
         NF(I)=KNF(I)
30       CONTINUE
         ISET=1
         IYZ=2
         IF(DDT.EQ.1) THEN
         CALLCHCKK(PP,IYZ,ISET)
         RETURN
         ENDIF
         CALL FCHECK(PP,IYZ,ISET)
         RETURN
         ENDIF
         IF(SST.EQ.1) THEN
         CALL PRCHCK(PP,IYZ,ISET)
         ENDIF
         RETURN
```

```
           END
C**************************************************
C*           SUBROUTINE PROB                      *
C**************************************************
         SUBROUTINE PROB
         INTEGER TIHAT,TJHAT,TI,TJ,NSA,TCAP,X,Y,T,PP,TT
         INTEGER ROUTE,TDMAND
         INTEGER TULOAD,TTRAVL,VDMAND,DMAND
         INTEGER VARS,VIRS,P,UTIME,TTTIME,DDT,SST
         REAL LOAD,KMAND,LTRAV
         DIMENSION NSA(5000),TI(5000),TJ(5000)
         DIMENSION TULOAD(100),TTRAVL(100),MEAN(300,300)
         DIMENSION VARS(300,300),MINE(300),VIRS(300),VDMAND(300)
         COMMON/A1/X(300),Y(300)
         COMMON/A2/NPT,NW,TCAP,MNP,NTRY
         COMMON/A3/MSVA(5000),NSAVE(5000),XX(5000)
         COMMON/A4/NB(100),NF(100),NR(100),P
         COMMON/A5/DMAND(300),TDMAND(100)
         COMMON/A6/LI,LJ,LI1,LI2,LJ1,LJ2,LRI,LRJ
         COMMON/A7/IBV,IWB
         COMMON/A8/DIST(300,300)
         COMMON/A9/TIHAT(5000),TJHAT(5000),ROUTE(100,100)
         COMMON/A10/ALPHA,BATA,ATAH,UTIME,TTTIME
         COMMON/A12/DDT,SST,IZAR
         COMMON/A13/TULOAD,TTRAVL
         COMMON/A14/MEAN,VARS,MINE,VIRS,VDMAND
         COMMON/A15/IEE,IFF,DELTA,IALGOL,BKAMA
         GO TO (4,5,8,4),IZAR
 4       CALL STINPT
 5       CALL SAVMAT
         TT=NTRY
         CALL TSORT(NSAVE,TIHAT,TJHAT,NTRY)
C**
C**   SET TOTAL DEMAND OF EACH ROUTE TO ZERO
C**   SET TOTAL TRAVELING TIME OF EACH ROUTE TO ZERO
C**   SET TOTAL UNLOADING TIME OF EACH ROUTE TO ZERO
C**
        DO 50 I=1,NTRY
50      MSVA(I)=NSAVE(I)
 8      DO 7 P=1,NW
        TDMAND(P)=0
        TTRAVL(P)=0
        TULOAD(P)=0
 7      CONTINUE
        T=1
 11     P=1
```

```
      R=3
      IX=5
      PP=P
      ITI=TIHAT(T)
      JTJ=TJHAT(T)
C**
C** TO DETERMINE THE DEMAND OF DEMAND POINTS ITI AND JTJ
C**
       KMAND=VDMAND(ITI)+VDMAND(JTJ)
       KMAND=SQRT(KMAND)
       TDMAND(P)=TDMAND(P)+(ATAH * KMAND)+DMAND(ITI)+DMAND(JTJ)
C**
C**  TO DETERMINE THE TOTAL UNLOADING TIME OF DEMANC POINTS ITJ,JTJ
       LOAD=VIRS(ITI)+VIRS(JTJ)
       LOAD=SQRT(LOAD)
       TULOAD(P)=TULOAD(P)+(BATA*LOAD)+MINE(ITI)+MINE(JTJ)
C**
C**  TO DETERMINE THE TOTAL TRAVELING TIME FOR DEMAND POINTS ITI
C**  AND JTJ
C**
      LTRAV=VARS(1,ITI)+VARS(JTJ,1)+VARS(ITI,JTJ)
      LTRAV=SQRT(LTRAV)
      TTRAVL(P)=TTRAVL(P)+MEAN(1,ITI)+MEAN(JTJ,1)+MEAN(ITI,JTJ)
     + +(ALPHA*LTRAV)
      IF(TDMAND(P).LE.TCAP.AND.TULOAD(P).LE.UTIME.AND.TTRAVL(P).
     + LE.TTTIME) THEN
       ROUTE(P,R)=TIHAT(T)
       NB(P)=ROUTE(P,R)
       R=R+1
       ROUTE(P,R)=TJHAT(T)
       NF(P)=ROUTE(P,R)
       NR(P)=R
       ENDIF
       K=T+1
       IF(TDMAND(P).GT.TCAP.OR.TULOAD(P).GT.UTIME.OR.TTRAVL(P).
     +  GT.TTTIME) THEN
       NSAVE(K-1)=0
       TDMAND(P)=0
       TTRAVL(P)=0
       TULOAD(P)=C
       T=K
       GO TO 11
       ENDIF
C**
C** TO CONSTRUCT A ROUTE
C**
```

```
      DO 10 T=K,TT
      NSAVE(T-1)=0
      IYOUTH=1
      CALL INTR(IN,PP,T,IYOUTH)
      IF(IN.EQ.1) GO TO 10
      PP=P
      MNK=MNP-1
      CALL RTCONT(PP,T)
 10   CONTINUE
      IYOUTH=2
      CALL INTR(IN,PP,T,IYOUTH)
      CALL WWRT(PP)
      RETURN
      END
C**************************************************
C***           SUBROUTINE STINPT
C**************************************************
C**
        SUBROUTINE STINPT
        INTEGER  DMAND,VDMAND,TCAP,VARS,VIRS,UTIME,TTTIME
        INTEGER X,Y,TDMAND
        DIMENSION MEAN(300,300) ,VARS(300,300),MINE(300)
        DIMENSION VIRS(300),VDMAND(300)
        COMMON/A1/X(300),Y(300)
        COMMON/A2/NPT,NW,TCAP,MNP,NTRY
        COMMON/A3/MSVA(5000),NSAVE(5000),XX(5000)
        COMMON/A5/DMAND(300),TDMAND(100)
        COMMON/A8/DIST(300,300)
        COMMON/A10/ALPHA,BATA,ATAH,UTIME,TTTIME
        COMMON/A12/DDT,SST,IZAR
        COMMON/A14/MEAN,VARS,MINE,VIRS,VDMAND
        COMMON/A15/IEE,IFF,DELTA,IALGOL,BKAMA
C**   IDSTB =1 STANDS FOR DISTRIBUTIONS SUCH AS POISSON ,BINOMIAL,
C**            EXPONENTIAL ,GAMMA,NEGATIVE BINOMIAL,AND CHI-SQUARE
C**   IDSTB=0  STANDS FOR OTHER DISTRIBUTIONS.
C**
      IF(IZAR.EQ.4) GO TO 10
      WRITE(6,80)
      WRITE(10,80)
80    FORMAT(5X,'ENTER THE TYPE OF DISTRIBUTION FUNCTIONS'/
     +'ENTER 1 FOR EXPON,BINOMIAL,CHI-SQURE,POISSON,NEG-BINO,GAMMA')
      WRITE(6,81)
      WRITE(10,81)
81    FORMAT(5X,' 0 OTHERWISE')
204   READ(5,*) IDSTB
      WRITE(6,200)
```

```
        WRITE(10,200)
        READ(5,*) ICORR
        IF(ICORR.EQ.2) THEN
        WRITE(6,201)
        WRITE(10,201)
        GO TO 204
        ENDIF
205     WRITE(6,1)
        WRITE(10,1)
 1      FORMAT(5X,'-->','ENTER THE NUMBER OF STOP POINTS INCLUDING'/
    +'THE TERMINAL AND TRUCK CAPACITY RESPECTIVELY')
C**
        READ(5,*) NPT,TCAP
        WRITE(6,101) NPT,TCAP
        WRITE(10,101) NPT,TCAP
101     FORMAT(//20X,I5,5X,I5)
        WRITE(6,200)
        WRITE(10,200)
        READ(5,*) ICORR
        IF(ICORR.EQ.2) THEN
        WRITE(6,201)
        WRITE(10,201)
        GO TO 205
        ENDIF
206     WRITE(6,3)
        WRITE(10,3)
3       FORMAT(5X,'--->',2X,'ENTER THE TOTAL UNLOAD TIME AND TOTAL'/
    +'TRAVELING TIME FOR EACH ROUTE')
        READ(5,*) UTIME,TTTIME
        WRITE(6,220) UTIME,TTTIME
        WRITE(10,220) UTIME,TTTIME
220     FORMAT(2X,'UTIME=',1X,I3,2X,'TTTIME=',1X,I3)
        WRITE(6,200)
        WRITE(10,200)
        READ(5,*) ICORR
        IF(ICORR.EQ.2) THEN
        WRITE(6,201)
        WRITE(10,201)
        GO TO 206
        ENDIF
C**
C**  ALPHA = PROBABILITY OF ROUTE FAILING FOR VIOLATING THE TOTAL
C**          TRAVELING TIME.
C**   BATA = PROBABILITY OF ROUTE FAILING FOR VAIOLATING THE TOTAL
C**          UNLOADING TIME
C**   ATAH = PROBABILITY OF ROUTE FAILING FOR VIOLATING THE CAPACITY
```

```
C**         OF TRUCK.
C**
207     WRITE(6,5)
        WRITE(10,5)
5       FORMAT(5X,'ENTER VALUES OF ALPHA,BATA,ATAH RESPECTIVELY')
C
         READ(5,*) ALPHA,BATA,ATAH
         WRITE(6,11237) ALPHA,BATA,ATAH
         WRITE(10,11237) ALPHA,BATA,ATAH
11237    FORMAT(//20X,F10.4,2X,F10.4,2X,F10.4)
         WRITE(6,200)
         WRITE(10,200)
         READ(5,*) ICORR
         IF(ICORR.EQ.2) THEN
         WRITE(6,201)
         WRITE(10,201)
         GO TO 207
         ENDIF
         GO TO 15
 10      WRITE(6,2)
         WRITE(10,2)
2        FORMAT(//10X,'--->',2X,'ENTER 0 FOR EUCLIDIAN DISTANCE'/
     +14X,'1 FOR LINEAR DISTANCE')
         READ(5,*) IEUC
         IF(IEUC.EQ.1) THEN
209      WRITE(6,12)
         WRITE(10,12)
 12      FORMAT(5X,'--->','ENTER THE LINEAR DISTANCE OR COST MATRIX')
        DO 11 I=1,NPT
        READ(5,*) (DIST(I,J),J=I,NPT)
        WRITE(6,42) (DIST(I,J),J=1,NPT)
        WRITE(10,42) (DIST(I,J),J=1,NPT)
        WRITE(6,200)
        WRITE(10,200)
        READ(5,*) ICORR
        IF(ICORR.EQ.2) THEN
        WRITE(6,201)
        WRITE(10,201)
        GO TO 209
        ENDIF
 11     CONTINUE
        DO 14 I=2,NPT
        K=I-1
        DO 13 J=1,K
        DIST(I,J)=DIST(J,I)
 13     CONTINUE
```

```
 14     CONTINUE
        ELSE
210     WRITE(6,4)
        WRITE(10,4)
4       FORMAT(//10X,'--->',2X,'ENTER THE EUCLIDIAN DISTANCE'/
     +10X,'WITH THE COORDINATE OF TERMINAL POINT FIRST')
        DO 6 I=1,NPT
        READ(5,*) X(I),Y(I)
        WRITE(6,9) X(I),Y(I)
        WRITE(10,9) X(I),Y(I)
        WRITE(6,200)
        WRITE(10,200)
        READ(5,*) ICORR
        IF(ICORR.EQ.2) THEN
        WRITE(6,201)
        WRITE(10,201)
        GO TO 210
        ENDIF
 6      CONTINUE
C**
C** WRITE THE EUCLIDIAN DISTANCE
C**
        WRITE(6,7)
        WRITE(10,7)
7       FORMAT(10X,'EUCLIDIAN DISTANCE'//16X,'X',10X,'Y')
        DO 8 I=1,NPT
        WRITE(6,9) X(I),Y(I)
        WRITE(10,9) X(I),Y(I)
 9      FORMAT(14X,I4,8X,I4)
 8      CONTINUE
C**
C**   EVALUATE THE DISTANCE BETWEEN POINTS I AND J
        DO 91 I=1,NPT
        DO 91  J=1,NPT
        IF(I.EQ.J) DIST(I,J)=0
        IF(I.GE.J) GO TO 91
        SOB=FLOAT((X(I)-X(J))**2+(Y(I)-Y(J))**2)
        DIST(I,J)=SQRT(SOB)
        DIST(J,I)=DIST(I,J)
 91     CONTINUE
        ENDIF
        IF(IZAR.EQ.4) RETURN
        IF(IEE.EQ.1) GO TO 223
15      CONTINUE
        WRITE(6,16)
        WRITE(10,16)
```

```
 16     FORMAT(5X,'ENTER THE MEAN TRAVEL TIME BETWEEN I AND J')
        DO 17 I=1,NPT
211     READ(9,*) (MEAN(I,J),J=I,NPT)
        WRITE(10,45) (MEAN(I,J),J=1,NPT)
        WRITE(6,200)
        WRITE(10,200)
C       READ(5,*) ICORR
        ICORR=1
        IF(ICORR.EQ.2) THEN
        WRITE(6,201)
        WRITE(10,201)
        GO TO 211
        ENDIF
17      CONTINUE
        WRITE(6,102)
        WRITE(10,102)
102     FORMAT(5X,'ENTER THE VARIANCE OF TRAVEL TIME BETWEEN I , J')
        DO 52 I=1,NPT
212     READ(9,*) (VARS(I,J),J=I,NPT)
        WRITE(10,48) (VARS(I,J),J=1,NPT)
        WRITE(6,200)
        WRITE(10,200)
        IF(ICORR.EQ.2) THEN
        WRITE(6,201)
        WRITE(10,201)
        GO TO 212
        ENDIF
 52     CONTINUE
        DO 30 I=2,NPT
        K=I-1
        DO 31 J=1,K
        MEAN(I,J)=MEAN(J,I)
        VARS(I,J)=VARS(J,I)
 31     CONTINUE
  30    CONTINUE
        WRITE(6,18)
        WRITE(10,18)
 18     FORMAT(5X,'--->','ENTER THE MEAN AND VARIANCE OF UNLOAD
     +  TIME FOR EACH DEMAND POINT I')
        DO 19 I=2,NPT
213     READ(9,*) MINE(I),VIRS(I)
        ICORR=1
        WRITE(6,221) MINE(I),VIRS(I)
        WRITE(10,221) MINE(I),VIRS(I)
221     FORMAT(/2X,'MEAN=',2X,I4,'VAR=',2X,I4)
        WRITE(6,200)
```

```
      WRITE(10,200)
      IF(ICORR.EQ.2) THEN
      WRITE(6,201)
      WRITE(10,201)
      GO TO 213
      ENDIF
 19   CONTINUE
      WRITE(6,20)
      WRITE(10,20)
20    FORMAT(5X,'--->','ENTER THE MEAN AND VARIANCE OF THE '/
     +9X,'DEMAND POINT I')
      DO 21 I=2,NPT
214   READ(9,*) DMAND(I),VDMAND(I)
      WRITE(6,222) DMAND(I),VDMAND(I)
      WRITE(10,222) DMAND(I),VDMAND(I)
222   FORMAT(2X,'DMAND=',1X,I4,2X,'VDMAND=',2X,I4)
      WRITE(6,200)
      WRITE(10,200)
      ICORR=1
      IF(ICORR.EQ.2) THEN
      WRITE(6,201)
      WRITE(10,201)
      GO TO 214
      ENDIF
 21   CONTINUE
      IF(IEE.EQ.1) GO TO 10
223   WRITE(6,22)
      WRITE(10,22)
      IF(IEE.EQ.1) THEN
 22   FORMAT(//15X,'DISTANCE MATRIX ')
      DO 25 I=1 ,NPT
      WRITE(6,42) (DIST(I,J),J=1,NPT)
      WRITE(10,42) (DIST(I,J),J=1,NPT)
 42   FORMAT(1X,15F5.1)
 25   CONTINUE
      ENDIF
      WRITE(6,43)
      WRITE(10,43)
43    FORMAT(//15X,'MEAN TRAVEL TIME')
 45   FORMAT(1X,20I4)
      WRITE(6,46)
      WRITE(10,46)
 46   FORMAT(//15X,'VARIANCE TRAVEL TIME')
48   FORMAT(1X,20I4)
      WRITE(6,26)
      WRITE(10,26)
```

```
 26      FORMAT(5X,'DEMAND POINT',2X,'DEMAND',10X,'VARIANCE DEMAND',
     +   10X,'MEAN UNLOADING',10X,'VARIANCE UNLOADING')
         DO 27 I=2,NPT
         WRITE(6,28) I,DMAND(I),VDMAND(I),MINE(I),VIRS(I)
         WRITE(10,28) I,DMAND(I),VDMAND(I),MINE(I),VIRS(I)
  28     FORMAT(10X,I3,2X,I5,19X,I5,19X,I5,22X,I5)
 27      CONTINUE
         IF(IDSTB.EQ.1) GO TO 50
         ITDD=0
         IVDD=0
         DO 41 I=2,NPT
         ITDD=ITDD+DMAND(I)
         IVDD=IVDD+VDMAND(I)
 41      CONTINUE
         BB=FLOAT(IVDD)
         IVDD=SQRT(BB)
         IVDD=ATAH*IVDD
         ITOTAL=IVDD+ITDD
         NW=(ITOTAL/TCAP)+10
         NNW=NW-10
         MNP=(NPT/NNW)+1
         MNP=MNP+5
         GO TO 99
50       WRITE(6,777)
         WRITE(10,777)
777      FORMAT(5X,'ENTER THE VALUE OF SAI')
         READ(5,*) ISAI
         BK=(ATAH**4)*(ISAI**2)+4*TCAP*(ATAH**2)*ISAI
         BKK=SQRT(BK)
         TCAPBR=(2*TCAP+(ATAH**2)*ISAI-BKK)*.5
         WRITE(6,61) TCAPBR
         WRITE(10,61) TCAPBR
61       FORMAT(//15X,'ARTIFICIAL CAPAVITY OF TRUCK=',F8.3)
C**
C** TOTAL MEAN DEMAND ON EACH ROUTE MUST BE LESS THAN TCAP=TCAPBR
C**
        ITDD=0
        DO 51 I=2,NPT
        ITDD=ITDD+DMAND(I)
 51     CONTINUE
        NW=(ITDD/TCAP)+10
        NNW=NW-10
        MNP=(NPT/NNW)+1
        MNP=MNP+5
99      CONTINUE
200     FORMAT(/2X,'CORRECT',2X,'1:YES',2X,'2:NO')
```

```
201     FORMAT(/2X,'REENTER AGAIN')
        RETURN
        END
C***********************************************
C*          SUBROUTINE PRCHCK                  *
C***********************************************
          SUBROUTINE PRCHCK(PP,IYZ,ISET)
          INTEGER DMAND,TDMAND,PP,ROUTE,TIHAT,TJHAT,P
          INTEGER VDMAND,MEAN,VARS,MINE,VIRS
          REAL KVD,KVVRS,KWVARS
          INTEGER TULOAD,TTRAVL,UTIME,TTTIME
          DIMENSION IDD(100),VDMAND(300),MEAN(300,300)
          DIMENSION VARS(300,300),MINE(300),VIRS(300)
          DIMENSION TULOAD(100),TTRAVL(100)
          COMMON/A4/NB(100),NF(100),NR(100),P
          COMMON/A5/DMAND(300),TDMAND(100)
          COMMON/A6/LI,LJ,LI1,LI2,LJ1,LJ2,LRI,LRJ
          COMMON/A9/TIHAT(5000),TJHAT(5000),ROUTE(100,100)
          COMMON/A10/ALPHA,BATA,ATAH,UTIME,TTTIME
          COMMON/A11/VD,IMEAN,VVRS,KMP,WVARS,KMON
          COMMON/A13/TULOAD,TTRAVL
          COMMON/A14/MEAN,VARS,MINE,VIRS,VDMAND
          GO TO (1,2),IYZ
 1        NN=NR(PP)
          TDMAND(PP)=0
          TTRAVL(PP)=0
          TULOAD(PP)=0
          VD=0
          DO 20 J=3,NN
          K=ROUTE(PP,J)
  20      VD=VD+VDMAND(K)
          KVD=SQRT(VD)
          KVD=ATAH*KVD
          DO 30 J=3,NN
          K=ROUTE(PP,J)
 30       TDMAND(PP)=TDMAND(PP)+DMAND(K)
          IMEAN=TDMAND(PP)
C**  TOTAL DEMAND OF ROUTE PP,CNSIDERING MEAN AND VARIANCE
          TDMAND(PP)=TDMAND(PP)+KVD
C** TO CALCULATE TOTAL STANDARED DEVIATION OF UNLOAD TIME
          VVRS=0
          DO 40 J=3,NN
          K=ROUTE(PP,J)
40        VVRS=VVRS+VIRS(K)
          KVVRS=SQRT(VVRS)
          KVVRS=BATA*KVVRS
```

```
        DO 50 J=3,NN
        K=ROUTE(PP,J)
50      TULOAD(PP)=TULOAD(PP)+MINE(K)
        KMP=TULOAD(PP)
C**   TOTAL UNLOAD TIME CONSIDERING MEAN AND VARIANCE
        TULOAD(PP)=TULOAD(PP)+KVVRS
C**
C***TO CALCULATE TOTAL TRAVEL TIME OF ROUTE PP.
C**
        WVARS=0
        DO 60 J=3,NN
        K=J-1
        KA=ROUTE(PP,K)
        KB=ROUTE(PP,J)
 60     WVARS=WVARS+VARS(KA,KB)
        KG=ROUTE(PP,NN)
        WVARS=WVARS+VARS(KG,1)
        KWVARS=SQRT(WVARS)
        KWVARS=ALPHA*KWVARS
        KMEN=0
        DO 70 J=3,NN
        K=J-1
        LA=ROUTE(PP,K)
        LB=ROUTE(PP,J)
70      KMEN=KMEN+MEAN(LA,LB)
        LG=ROUTE(PP,NN)
        KMON =KMEN+MEAN(LG,1)
        TTRAVL(PP)=TTRAVL(PP)+KMON+KWVARS
        RETURN
C*****
C***
 2      DO 80 I=ISET,PP
        TDMAND(I)=0
        NN=NR(I)
        VD=0
        DO 90 J=3,NN
        K=ROUTE(I,J)
 90     VD=VD+VDMAND(K)
        KVD=SQRT(VD)
        KVD=ATAH*KVD
        DO 100 J=3,NN
        K=ROUTE(I,J)
100     TDMAND(I)=TDMAND(I)+DMAND(K)
        TDMAND(I)=TDMAND(I)+KVD
  80    CONTINUE
C** TO FIND THE TOTAL UNLOAD TIME OF EACH CONSTRUCTED ROUTE
```

```
      DO 110 I=ISET,PP
      TULOAD(I)=0
      NN=NR(I)
      VVRS=0
      DO 120 J=3,NN
      K=ROUTE(I,J)
 120  VVRS=VVRS+VIRS(K)
      KVVRS=SQRT(VVRS)
      KVVRS=BATA*KVVRS
      DO 130 J=3,NN
      K=ROUTE(I,J)
 130  TULOAD(I)=TULOAD(I)+MINE(K)
      TULOAD(I)=TULOAD(I)+KVVRS
 110  CONTINUE
C** TO FIND THE TOTAL TRAVEL TIME FOR EACH CONSTRUCTED ROUTE
C**
       DO 140 I=ISET,PP
       TTRAVL(I)=0
       NN=NR(I)
C** TO CALCULATE TOTAL VARIANCE OF TRAVEL TIME FOR ROUTEI
      WVARS=0
      DO 150 J=3,NN
      K=J-1
      KA=ROUTE(I,K)
      KB=ROUTE(I,J)
 150  WVARS=WVARS+VARS(KA,KB)
      KG=ROUTE(I,NN)
      WVARS=WVARS+VARS(KG,1)
      KWVARS=SQRT(WVARS)
      KWVARS=ALPHA*KWVARS
C** TO CALCULATE TOTAL MEAN TRAVEL TIME OF ROUTE I
      KMEN=0
      DO 160 J=3,NN
      K=J-1
      LA=ROUTE(I,K)
      LB=ROUTE(I,J)
 160  KMEN=KMEN+MEAN(LA,LB)
      LG=ROUTE(I,NN)
C**
C** TO FIND TOTAL MEAN TRAVEL TIME OF ROUTE I
      KMON=KMEN+MEAN(LG,1)
      TTRAVL(I)=TTRAVL(I)+KMON+KWVARS
140   CONTINUE
      RETURN
      END
C***********************************************
```

```
C*             UBROUTINE STFSBL                        *
C*************************************************
      SUBROUTINE STFSBL(IX,FLI,FLJ,FLIJ,FIJ,T,IZX,KDMAND,KTULOD,
     +   KTTRVL)
      DIMENSION MEAN(300,300),VARS(300,300),MINE(300)
      DIMENSION TULOAD(100),TTRAVL(100)
      DIMENSION VIRS(300),VDMAND(300)
      INTEGER TCAP,DMAND,TIHAT,TJHAT,ROUTE,FLI,FLJ,FLIJ,FIJ
      INTEGER TDMAND,P,T,PP,UTIME,TTTIME
      INTEGER VARS,VIRS,VDMAND,TULOAD,TTRAVL
      REAL KVD,KVVRS,LOAD,KMAND,LTRAV,KWVARS
      COMMON/A2/NPT,NW,TCAP,MNP,NTRY
      COMMON/A4/NB(100),NF(100),NR(100),P
      COMMON/A5/DMAND(300),TDMAND(100)
      COMMON/A6/LI,LJ,LI1,LI2,LJ1,LJ2,LRI,LRJ
      COMMON/A9/TIHAT(5000),TJHAT(5000),ROUTE(100,100)
      COMMON/A10/ALPHA,BATA,ATAH,UTIME,TTTIME
      COMMON/A11/VD,IMEAN,VVRS,KMP,WVARS,KMON
      COMMON/A13/TULOAD,TTRAVL
      COMMON/A14/MEAN,VARS,MINE,VIRS,VDMAND
      ITI=TIHAT(T)
      JTJ=TJHAT(T)
      GO TO (1,2,3,4),IX
 1    IYZ=1
      ISET=1
      CALL PRCHCK(LRI,IYZ,ISET)
      IRAS=ITI
      IRAK=JTJ
      KRAFT=LRI
      CALL CONTRL(IRAS,IRAK,KRAFT,KDMAND,KTULOD,KTTRVL)
     IF(KDMAND.LE.TCAP.AND.KTULOD.LE.UTIME.AND.KTTRVL.LE.TTTIME)
     + THEN
     FLI=1
     ELSE
     FLI=0
     ENDIF
     RETURN
 2   IYZ=1
     ISET=1
     CALL PRCHCK(LRJ,IYZ,ISET)
     IRAK=ITI
     IRAS=JTJ
     KRAFT=LRJ
     CALL CONTRL(IRAS,IRAK,KRAFT,KDMAND,KTULOD,KTTRVL)
     IF(KDMAND.LE.TCAP.AND.KTULOD.LE.UTIME.AND.KTTRVL.LE.TTTIME)
     + THEN
```

```
      FLJ=1
      ELSE
      FLJ=0
      ENDIF
      RETURN
3     IYZ=1
      ISET=1
      CALL PRCHCK(LRI,IYZ,ISET)
      M1=IMEAN
      V1=VD
      N1=KMP
      W1=VVRS
      MYN1=KMON
      VD1=WVARS
      CALL PRCHCK(LRJ,IYZ,ISET)
      M2=IMEAN
      V2=VD
      N2=KMP
      W2=VVRS
      MYN2=KMON
      VD2=WVARS
      CALL SOFT(KTTRVL,VD1,VD2,MYN1,MYN2,ITI,JTJ)
      CALL RUSH(M1,M2,N1,N2,V1,V2,W1,W2,KDMAND,KTULOD)
      IF(KDMAND.LE.TCAP.AND.KTULOD.LE.UTIME.AND.KTTRVL.LE.TTTIME)
     + THEN
      FLIJ=1
      ELSE
      FLIJ=0
      ENDIF
      RETURN
4     IYZ=1
      ISET=1
      CALL PRCHCK(LRI,IYZ,ISET)
      M1=IMEAN
      V1=VD
      N1=KMP
      W1=VVRS
      MYN1=KMON
      VD1=WVARS
      CALL PRCHCK(LRJ,IYZ,ISET)
      M2=IMEAN
      V2=VD
      N2=KMP
      W2=VVRS
      MYN2=KMON
      VD2=WVARS
```

```
      CALL SOFT(KTTRVL,VD1,VD2,MYN1,MYN2,ITI,JTJ)
      CALL RUSH(M1,M2,N1,N2,V1,V2,W1,W2,KDMAND,KTULOD)
      IF(KDMAND.LE.TCAP.AND.KTULOD.LE.UTIME.AND.KTTRVL.LE.TTTIME)
     +   THEN
      FIJ=1
      ELSE
      FIJ=0
      ENDIF
      RETURN
      END
C*********************************************
C*          SUBROUTINE CONTRL                *
C*********************************************
C**
      SUBROUTINE CONTRL(IRAS,IRAK,KRAFT,KDMAND,KTULOD,KTTRVL)
      INTEGER DMAND,VDMAND,TCAP,VARS,VIRS,UTIME,TTTIME
      REAL KVD,KVVRS,KWVARS
      INTEGER TULOAD ,TTRAVL,TDMAND
      DIMENSION MEAN(300,300),VARS(300,300),MINE(300)
      DIMENSION VIRS(300),VDMAND(300)
      DIMENSION TULOAD(100),TTRAVL(100)
      COMMON/A5/DMAND(300),TDMAND(100)
      COMMON/A10/ALPHA,BATA,ATAH,UTIME,TTTIME
      COMMON/A11/VD,IMEAN,VVRS,KMP,WVARS,KMON
      COMMON/A13/TULOAD,TTRAVL
      COMMON/A14/MEAN,VARS,MINE,VIRS,VDMAND
C**
C**   TO EVALUATE THE VARIANCE OF DEMAND
      VDNEW=VD+VDMAND(IRAK)
C**
C**  TO EVALUATE THE VARIANCE OF UNLOADING TIME
      VVRSNU=VVRS+VIRS(IRAK)
C**
C**  TO EVALUATE THE VARIANCE OFTRAVELING TIME
      WVARSU=WVARS+VARS(1,IRAK)+VARS(IRAS,IRAK)-VARS(IRAS,1)
C**
      KVD=SQRT(VDNEW)
      KVD=ATAH*KVD
      KDMAND=IMEAN+DMAND(IRAK)+KVD
C**
C**
      KVVRS=SQRT(VVRSNU)
      KVVRS=BATA*KVVRS
      KTULOD=KMP+MINE(IRAK)+KVVRS
C**
C**
```

```
      KWVARS=SQRT(WVARSU)
      KWVARS=ALPHA*KWVARS
      KTTRVL=KMON+MEAN(1,IRAK)+MEAN(IRAS,IRAK)+KWVARS-MEAN(IRAS,1)
      RETURN
      END
C*********************************************
C*            SUBROUTINE SOFT               *
C*********************************************
        SUBROUTINE SOFT(KTTRVL,VD1,VD2,MYN1,MYN2,ITI,JTJ)
        INTEGER VARS,VIRS,VDMAND
        REAL KWVARS
        DIMENSION MEAN(300,300),VARS(300,300)
        DIMENSION MINE(300),VIRS(300),VDMAND(300)
        COMMON/A10/ALPHA,BATA,ATAH,UTIME,TTTIME
        COMMON/A14/MEAN,VARS,MINE,VIRS,VDMAND
        MEN=MEAN(1,ITI)+MEAN(1,JTJ)
        MENS=MEN-MEAN(ITI,JTJ)
        MEANTL=MYN1+MYN2-MENS
        VD3=VARS(1,ITI)+VARS(1,JTJ)
        SVD3=VD3-VARS(ITI,JTJ)
        VD=VD1+VD2-SVD3
        KWVARS=SQRT(VD)
        KWVARS=ALPHA*KWVARS
        KTTRVL=MEANTL+KWVARS
        RETURN
        END
C**********************************************
C*           SUBROUTINE RUSH                 *
C**********************************************
      SUBROUTINE RUSH(M1,M2,N1,N2,V1,V2,W1,W2,KDMAND,KTULOD)
      COMMON/A10/ALPHA,BATA,ATAH,UTIME,TTTIME
      REAL IV3,IW3
      M3=M1+M2
      V3=V2+V1
      IV3=SQRT(V3)
      IV3=ATAH*IV3
      KDMAND=M3+IV3
      N3=N1+N2
      W3=W1+W2
      IW3=SQRT(W3)
      IW3=BATA*IW3
      KTULOD=N3+IW3
      RETURN
      END
C***********************************************
C*            SUBROUTINE STSAVE                *
```

```
C*************************************************
C** THIS SUBROUTINE   CONSTRUCT THE SAVING MATRIX FOR PROBLEM F WHEN
C** TIME IS CONSERNED.
         DIMENSION VARS(300,300),MINE(300),VIRS(300),VDMAND(300)
         DIMENSION MEAN(300,300) ,ISAVE(300,300),WAR(100,100)
         DIMENSION MAR(100,100)
         INTEGER TCAP,X,P,TIHAT,TJHAT,ROUTE,VARS,VIRS,VDMAND
         COMMON/A2/NPT,NW,TCAP,MNP,NTRY
         COMMON/A3/MSVA(5000),NSAVE(5000),XX(5000)
         COMMON/A4/NB(100),NF(100),NR(100),P
         COMMON/A8/DIST(300,300)
         COMMON/A9/TIHAT(5000),TJHAT(5000),ROUTE(100,100)
         COMMON/A14/MEAN,VARS,MINE,VIRS,VDMAND
         COMMON/A15/IEE,IFF,DELTA,IALGOL,BKAMA
         IF(IALGOL.EQ.2) THEN
         ISIGMA=0
         DO 10 I=2,NPT
         DO 10 J=2,NPT
         ISIGMA=ISIGMA+VARS(I,J)
10       CONTINUE
         K=NPT-1
         KK=K*K
         IBAR=ISIGMA/(KK*DELTA)
         ENDIF
         DO 15 I=2,NPT
         DO 15 J=2 ,NPT
         WAR(I,J)=0
         MAR(I,J)=0
 15      CONTINUE
         DO 20 I=2,NPT
         DO 30 J=I,NPT
C PURPOSE TO DETERMINE THE LIST OF SAVINGS FOR BOTH ALGORITHMS
C OF "F" TYPE PROBLEM
        ISAVE(I,J)=-99999
        IF(MEAN(I,J).EQ.0) GO TO 40
        MAR(I,J)=MEAN(I,1)+MEAN(1,J)-MEAN(I,J)
C PURPOSE TO DETERMINE THE SAVINGS FOR ALGORITHM(I)
       IF(IALGOL.EQ.1) THEN
       MAR(I,J)=BKAMA*MAR(I,J)
       ENDIF
C
 40    IF(VARS(I,J).EQ.0) GO TO 50
       WAR(I,J)=VARS(I,1)+VARS(1,J)+VARS(I,J)
C
       IF(IALGOL.EQ.1) THEN
       WAR(I,J)=(1-BKAMA)*SQRT(WAR(I,J))
```

```
      ENDIF
C
 50   MAR(J,I)=MAR(I,J)
      WAR(J,I)=WAR(I,J)
      IF(WAR(I,J).EQ.O) GO TO 30
      IF(IALGOL.EQ.2) THEN
      ISAVE(I,J)=MAR(I,J)+IBAR/SQRT(WAR(I,J))
      ENDIF
      IF(IALGOL.EQ.1) THEN
      ISAVE(I,J)=MAR(I,J)+WAR(I,J)
      ENDIF
C
      ISAVE(J,I)=ISAVE(I,J)
 30   CONTINUE
 20   CONTINUE
      IF(IALGOL.EQ.1.AND.BKAMA.EQ.1) THEN
      ISAVE(I,J)=MAR(I,J)
      ISAVE(J,I)=ISAVE(I,J)
      ENDIF
      DO 60 I=2,NPT
      ISAVE(I,1)=-99999
      ISAVE(1,I)=ISAVE(I,1)
 60   CONTINUE
C**
C**
C**PUT ARRAY ISAVE INTO A ONE DIMENSIONAL ARRAY
       L=O
       IPT=NPT-1
       DO 100 I=2,IPT
       K=I+1
       DO 100 J=K,NPT
       IF(ISAVE(I,J).LE.O)  GO TO 100
       L=L+1
       NSAVE(L)=ISAVE(I,J)
       TIHAT(L)=I
       TJHAT(L)=J
 100   CONTINUE
       NTRY=L
       DO 170 I=1,NW
       ROUTE(I,1)=I
       ROUTE(I,MNP)=1
       ROUTE(I,2)=1
       MNP1=MNP-1
       DO 180 J=3,MNP1
 180   ROUTE(I,J)=O
       NB(I)=O
```

```
      NF(I)=0
      NR(I)=2
170   CONTINUE
      DO 13 I=1,NW
      WRITE(6,23) (ROUTE(I,J),J=1,MNP)
 23   FORMAT(5X,30(I2,2X))
 13   CONTINUE
      RETURN
      END
C***************************************************
C*                SUBROUTINE FSBL                  *
C***************************************************
      SUBROUTINE FSBL(IX,FLI,FLJ,FLIJ,FIJ,T,IZX,KDMAND)
      DIMENSION MEAN(300,300),VARS(300,300)
      DIMENSION MINE(300),TULOAD(100),TTRAVL(100),VIRS(300)
      DIMENSION VDMAND(300)
      INTEGER TCAP,DMAND,TIHAT,TJHAT,ROUTE,FLI,FLJ,FLIJ,FIJ
      INTEGER TDMAND,P,T,PP,XX,UTIME,TTTIME
      INTEGER VARS,VIRS,VDMAND,TULOAD,TTRAVL
      COMMON/A2/NPT,NW,TCAP,MNP,NTRY
      COMMON/A3/MSVA(5000),NSAVE(5000),XX(5000)
      COMMON/A4/NB(100),NF(100),NR(100),P
      COMMON/A5/DMAND(300),TDMAND(100)
      COMMON/A6/LI,LJ,LI1,LI2,LJ1,LJ2,LRI,LRJ
      COMMON/A7/IBV,IWB
      COMMON/A9/TIHAT(5000),TJHAT(5000),ROUTE(100,100)
      COMMON/A10/ALPHA,BATA,ATAH,UTIME,TTTIME
      COMMON/A11/VD,IMEAN,VVRS,KMP,WVARS,KMON
      COMMON/A13/TULOAD,TTRAVL
      COMMON/A14/MEAN,VARS,MINE,VIRS,VDMAND
      COMMON/A15/IEE,IFF,DELTA,IALGOL,BKAMA
      COMMON/A16/KPRO,GAMA
C PURPOSE OF THIS SUBROUTINE IS TO DETERMINE THE FEASIBILITY
C FOR THE SVRP WHEN CUSTOMER DEMANDS ARE ONLY PROBABILISTICS
      IYZ=1
      ISET=1
      ITI=TIHAT(T)
      JTJ=TJHAT(T)
      GO TO (1,2,3,4),IX
1     CALL FCHECK(LRI,IYZ,ISET)
      IRAK=ITI
      IRAS=JTJ
      KRAFT=LRI
      CALL STCONT(IRAS,IRAK,KRAFT,KDMAND)
      IF(KDMAND.LE.TCAP) THEN
      FLI=1
```

```
      ELSE
      FLI=0
      ENDIF
      RETURN
 2    CALL FCHECK (LRJ,IYZ,ISET)
      IRAS=ITI
      IRAK=JTJ
      KRAFT=LRJ
      CALL STCONT(IRAS,IRAK,KRAFT,KDMAND)
      IF(KDMAND.LE.TCAP) THEN
      FLJ=1
      ELSE
      FLJ=0
      ENDIF
      RETURN
 3    CALL FCHECK(LRI,IYZ,ISET)
      M1=IMEAN
      V1=VD
      CALL FCHECK(LRJ,IYZ,ISET)
      M2=IMEAN
      V2=VD
      CALL SFAST(M1,M2,V1,V2,KDMAND)
      IF(KDMAND.LE.TCAP) THEN
      FLIJ=1
      ELSE
      FLIJ=0
      ENDIF
      RETURN
C**
C**
 4    CALL FCHECK(LRI,IYZ,ISET)
      M1=IMEAN
      V1=VD
      CALL FCHECK(LRJ,IYZ,ISET)
      M2=IMEAN
      V2=VD
      CALL SFAST(M1,M2,V1,V2,KDMAND)
      IF(KDMAND.LE.TCAP) THEN
      FIJ=1
      ELSE
      FIJ=0
      ENDIF
      RETURN
      END
C*************************************************
C*         SUBROUTINE FCHECK                     *
```

```
C**************************************************
        SUBROUTINE FCHECK(PP,IYZ,ISET)
        DIMENSION IDD(100),VDMAND(300),MEAN(300,300)
        DIMENSION VARS(300,300),MINE(300),VIRS(300)
        DIMENSION TULOAD(100),TTRAVL(100)
        INTEGER DMAND,TDMAND,PP,ROUTE,TIHAT,TJHAT,P
        INTEGER VDMAND,MEAN,VARS,MINE,VIRS
        INTEGER TULOAD,TTRAVL,UTIME,TTTIME
        REAL KVD
        COMMON/A4/NB(100),NF(100),NR(100),P
        COMMON/A5/ DMAND(300),TDMAND(100)
        COMMON/A6/LI,LJ,LI1,LI2,LJ1,LJ2,LRI,LRJ
        COMMON/A9/TIHAT(5000),TJHAT(5000),ROUTE(100,100)
        COMMON/A10/ALPHA,BATA,ATAH,UTIME,TTTIME
        COMMON/A11/VD,IMEAN,VVRS,KMP,WVARS,KMON
        COMMON/A13/TULOAD,TTRAVL
        COMMON/A14/MEAN,VARS,MINE,VIRS,VDMAND
        COMMON/A15/IEE,IFF,DELTA,IALGOL,BKAMA
        COMMON/A16/KPRO,GAMA
        IF(KPRO.EQ.1) ATAH=GAMA
C
        GO TO (1,2), IYZ
 1      NN=NR(PP)
        TDMAND(PP)=0
        VD=0
        DO 10 J=3,NN
        K=ROUTE(PP,J)
 10     VD=VD+VDMAND(K)
        KVD=SQRT(VD)
        KVD=ATAH*KVD
        DO 20 J=3,NN
        K=ROUTE(PP,J)
 20     TDMAND(PP)=TDMAND(PP)+DMAND(K)
        IMEAN=TDMAND(PP)
C**  TOTAL DEMAND OF ROUTE PP  CONSIDERING MEAN AND VARIANCE
C**  OF ALL DEMAND POINTS LOCATED ON THIS ROUTE.
         TDMAND(PP)=TDMAND(PP)+KVD
         RETURN
 2       DO 80 I=ISET,PP
         TDMAND(I)=0
         NN=NR(I)
         VD=0
         DO 90 J=3,NN
         K=ROUTE(I,J)
 90      VD=VD+VDMAND(K)
         KVD=SQRT(VD)
```

```
      KVD=ATAH*KVD
      DO 100 J=3,NN
      K=ROUTE(I,J)
100   TDMAND(I)=TDMAND(I)+DMAND(K)
      TDMAND(I)=TDMAND(I)+KVD
 80   CONTINUE
      RETURN
      END
C************************************************
C*            SUBROUTINE SFAST                  *
C************************************************
      SUBROUTINE SFAST(M1,M2,V1,V2,KDMAND)
      COMMON/A10/ALPHA,BATA,ATAH,UTIME,TTTIME
C**
      REAL IV3
      COMMON/A16/KPRO,GAMA
      IF(KPRO.EQ.1) ATAH=GAMA
      M3=M1+M2
      V3=V1+V2
      IV3=SQRT(V3)
      IV3=ATAH*IV3
      KDMAND=M3+IV3
      RETURN
      END
C************************************************
C*               SUBROUTINE STCONT              *
C************************************************
      SUBROUTINE STCONT(IRAS,IRAK,KRAFT,KDMAND)
      DIMENSION MEAN(300,300),VARS(300,300),MINE(300)
      DIMENSION VIRS(300),VDMAND(300),TULOAD(100),TTRAVL(100)
      INTEGER DMAND,VDMAND,TCAP,VARS,VIRS,UTIME,TTTIME
      INTEGER TULOAD,TTRAVL,TDMAND
      REAL KVD
      COMMON/A5/DMAND(300),TDMAND(100)
      COMMON/A10/ALPHA,BATA,ATAH,UTIME,TTTIME
      COMMON/A11/VD,IMEAN,VVRS,KMP,WVARS,KMON
      COMMON/A13/TULOAD,TTRAVL
      COMMON/A14/MEAN,VARS,MINE,VIRS,VDMAND
      COMMON/A15/IEE,IFF,DELTA,IALGOL,BKAMA
      COMMON/A16/KPRO,GAMA
      IF(KPRO.EQ.1) ATAH=GAMA
C**
C**  TO EVALUATE THE VARIANCE OF DEMAND
      VDNEW=VD+VDMAND(IRAS)
      KVD=SQRT(VDNEW)
      KVD=ATAH*KVD
```

```
      KDMAND=IMEAN+DMAND(IRAS)+KVD
      RETURN
      END
C*************************************************
C*              SUBROUTINE STATS                 *
C*************************************************
      SUBROUTINE STATS
C**
      DIMENSION MEAN(300,300),VARS(300,300),MINE(300)
      DIMENSION TULOAD(100),TTRAVL(100),VIRS(300)
      DIMENSION VDMAND(300)
      INTEGER TCAP,DMAND,TIHAT,TJHAT,ROUTE,FLI,FLJ,FLIJ,FIJ
      INTEGER TDMAND,P,T,PP,TT,XX,UTIME,TTTIME
      INTEGER VARS,VIRS,VDMAND,TULOAD,TTRAVL,DDT,SST
      REAL KMAND
      COMMON/A2/NPT,NW,TCAP,MNP,NTRY
      COMMON/A3/MSVA(5000),NSAVE(5000),XX(5000)
      COMMON/A4/NB(100),NF(100),NR(100),P
      COMMON/A5/DMAND(300),TDMAND(100)
      COMMON/A6/LI,LJ,LI1,LI2,LJ1,LJ2,LRI,LRJ
      COMMON/A7/IBV,IWB
      COMMON/A9/TIHAT(5000),TJHAT(5000),ROUTE(100,100)
      COMMON/A10/ALPHA,BATA,ATAH,UTIME,TTTIME
      COMMON/A11/VD,IMEAN,VVRS,KMP,WVARS,KMON
      COMMON/A13/TULOAD,TTRAVL
      COMMON/A14/MEAN,VARS,MINE,VIRS,VDMAND
      COMMON/A15/IEE,IFF,DELTA,IALGOL,BKAMA
      COMMON/A12/DDT,SST,IZAR
      COMMON/A16/KPRO,GAMA
      COMMON/A17/KDMAND,KTULOD,KTTRVL
C**
      IF(KPRO.EQ.1) THEN
      GO TO (20,21,22),IZAR
20    WRITE(6,4)
 4    FORMAT(//10X,'---',2X,'ENTER 0 FOR EUCLIDIAN DISTANCE AND
     +1 FOR LINEAR DISTANCE')
      READ(5,*) IDDT
      CALL INPT(IDDT)
21    CALL SAVMAT
      GO TO 9
      ENDIF
C**
      GO TO (30,31,22),IZAR
30    CALL STINPT
31    CALL STSAVE
9     TT=NTRY
```

```
          CALL TSORT(NSAVE,TIHAT,TJHAT,NTRY)
C**    SET THE TOTAL DEMAND OF EACH ROUTE TO ZERO
          DO 50 I=1,NTRY
50        MSVA(I)=NSAVE(I)
22        DO 7 P=1,NW
          TDMAND(P)=0
 7        CONTINUE
          T=1
 11       P=1
          R=3
          PP=P
          ITI=TIHAT(T)
          JTJ=TJHAT(T)
C**     TO DETERMINE THE DEMAND POINTS OF ITI AND JTJ.
C**
          IF(KPRO.EQ.1) ATAH=GAMA
          KMAND=VDMAND(ITI)+VDMAND(JTJ)
          KMAND=SQRT(KMAND)
          TDMAND(P)=TDMAND(P)+(ATAH*KMAND)+DMAND(ITI)+DMAND(JTJ)
          IF(TDMAND(P).LE.TCAP) THEN
          ROUTE(P,R)=TIHAT(T)
          NB(P)=ROUTE(P,R)
          R=R+1
          ROUTE(P,R)=TJHAT(T)
          NF(P)=ROUTE(P,R)
          NR(P)=R
          ENDIF
          K=T+1
          IF(TDMAND(P).GT.TCAP) THEN
          NSAVE(K-1)=0
          TDMAND(P)=0
          T=K
          GO TO 11
          ENDIF
C**      TO CONSTRUCT A ROUTE
          DO 10 T=K,TT
          NSAVE(T-1)=0
          IYOUTH=1
          CALL INTR(IN,PP,T,IYOUTH)
          IF(IN.EQ.1) GO TO 10
          PP=P
          MNK=MNP-1
          CALL RTCONT(PP,T)
 10       CONTINUE
          IYOUTH=2
          CALL INTR(IN,PP,T,IYOUTH)
```

```
      CALL WWRT(PP)
      RETURN
      END
C**************************************************
C**          SUBROUTINE  STCTD
C**************************************************
C**
      SUBROUTINE STCTD(PP,TTOTAL )
      DIMENSION IDD(100),VDMAND(300),MEAN(300,300)
      DIMENSION TTRAVL(100),VARS(300,300),MINE(300)
      DIMENSION VIRS(300),TULOAD(100)
      INTEGER TULOAD,TTRAVL,UTIME,TTTIME,VDMAND,MEAN,VARS
      INTEGER  MINE,VIRS,DMAND,TDMAND,PP,ROUTE,P
      INTEGER  TIHAT,TJHAT
      COMMON/A4/NB(100),NF(100),NR(100),P
      COMMON/A5/DMAND(300),TDMAND(100)
      COMMON/A6/LI,LJ,LI1,LI2,LJ1,LJ2,LRI,LRJ
      COMMON/A9/TIHAT(5000),TJHAT(5000),ROUTE(100,100)
      COMMON/A10/ALPHA,BATA,ATAH,UTIME,TTTIME
      COMMON/A11/VD,IMEAN,VVRS,KMP,WVARS,KMON
      COMMON/A13/TULOAD,TTRAVL
      COMMON/A14/MEAN,VARS,MINE,VIRS,VDMAND
C**
      TTOTAL=0.
      IYZ=2
      ISET=1
      CALL PRCHCK(PP,IYZ,ISET)
      DO 10 I=1,PP
      TTOTAL=TTOTAL+TULOAD(I)+TTRAVL(I)
 10   CONTINUE
      RETURN
      END
```

APPENDIX C

DATA FOR TEST PROBLEMS 1, 2, AND 3

TABLE XXIV

50 NODE PROBLEM

TEST PROBLEM #1

i	x_i	y_i	μ_i	σ_i^2	i	x_i	y_i	μ_i	σ_i^2
1	37	52	7	3	26	27	68	7	2
2	49	49	30	100	27	30	48	25	9
3	52	64	16	5	28	43	67	14	4
4	20	26	9	9	29	58	48	6	4
5	40	30	21	18	30	58	27	19	40
6	21	47	15	6	31	37	69	11	2
7	17	63	19	14	32	38	46	12	4
8	31	62	23	15	33	46	10	23	11
9	52	33	11	3	34	61	33	26	27
10	51	21	5	1	35	62	63	17	6
11	42	41	19	14	36	63	69	6	1
12	31	32	29	53	37	32	22	9	2
13	5	25	23	33	38	45	35	15	14
14	12	42	21	49	39	59	15	14	4
15	36	16	10	6	40	5	6	7	3
16	52	41	15	9	41	10	17	27	81
17	27	23	3	0	42	21	10	13	3
18	17	33	41	67	43	5	64	11	2
19	13	13	9	2	44	30	15	16	28
20	57	58	28	87	45	39	10	10	11
21	62	42	8	2	46	32	39	5	1
22	42	57	8	2	47	25	32	25	25
23	16	57	16	7	48	25	55	17	8
24	8	52	10	6	49	48	28	18	13
25	7	38	28	16	50	56	37	10	3

Central Depot is at $x_o = 30$, $y_o = 40$

Vehicle Capacity is $Q = 160$

TABLE XXV

75 NODE PROBLEM

TEST PROBLEM #2

i	x_i	y_i	μ_i	σ_i^2	i	x_i	y_i	μ_i	σ_i^2
1	22	22	18	7	44	21	48	17	12
2	36	26	26	27	45	50	30	21	18
3	21	45	11	2	46	51	42	27	46
4	45	35	30	18	47	50	15	19	40
5	55	20	21	28	48	48	21	20	11
6	33	34	19	23	49	12	38	5	1
7	50	50	15	5	50	15	56	22	54
8	55	45	16	28	51	29	39	12	3
9	26	59	29	17	52	54	38	19	40
10	40	66	26	75	53	55	57	22	30
11	55	65	37	38	54	67	41	16	7
12	35	51	16	16	55	10	70	7	5
13	62	35	12	4	56	6	25	26	27
14	62	57	31	107	57	65	27	14	4
15	62	24	8	2	58	40	60	21	9
16	21	36	19	10	59	70	64	24	19
17	33	44	20	8	60	64	4	13	3
18	9	56	13	11	61	36	6	15	25
19	62	48	15	6	62	30	20	18	9
20	66	14	22	30	63	20	30	11	3
21	44	13	28	87	64	15	5	28	22
22	26	13	12	9	65	50	70	9	3
23	11	28	6	2	66	57	72	37	152
24	7	43	27	81	67	45	42	30	36
25	17	64	14	22	68	38	33	10	4
26	41	46	18	36	69	50	4	8	7
27	55	34	17	6	70	66	8	11	8
28	35	16	29	23	71	59	5	3	0
29	52	26	13	7	72	35	60	1	0
30	43	26	22	19	73	27	24	6	1
31	31	76	25	17	74	40	20	10	11
32	22	53	28	22	75	40	37	20	44
33	26	29	27	81					
34	50	40	19	10					
35	55	50	10	6					
36	54	10	12	3					
37	60	15	14	4					
38	47	66	24	64					
39	30	60	16	7					
40	30	50	33	30					
41	12	17	15	9					
42	15	14	11	2					
43	16	19	18	13					

Central Depot is at $x_o = 40$, $y_o = 40$

Vehicle Capacity is $Q = 140$

TABLE XXVI

50 NODE PROBLEM

TEST PROBLEM #3

i	x_i	y_i	Demand μ_i	Demand σ_i^2	Mean Unload Time*	i	x_i	y_i	Demand μ_i	Demand σ_i^2	Mean Unload Time*
1	37	52	7	3	7	26	27	68	7	2	7
2	49	49	30	100	30	27	30	48	25	9	25
3	52	64	16	5	16	28	43	67	14	4	14
4	20	26	9	9	9	29	58	48	6	4	6
5	40	30	21	18	21	30	58	27	19	40	19
6	21	47	15	6	15	31	37	69	11	2	11
7	17	63	19	14	19	32	38	46	12	4	12
8	31	62	23	15	23	33	46	10	23	11	23
9	52	33	11	3	11	34	61	33	26	27	26
10	51	21	5	1	5	35	62	63	17	6	17
11	42	41	19	14	19	36	63	69	6	1	6
12	31	32	29	53	29	37	32	22	9	2	9
13	5	25	23	33	23	38	45	35	15	14	15
14	12	42	21	49	21	39	59	15	14	4	14
15	36	16	10	6	10	40	5	6	7	3	7
16	52	41	15	9	15	41	10	17	27	81	27
17	27	23	3	0	3	42	21	10	13	3	13
18	17	33	41	67	41	43	5	64	11	2	11
19	13	13	9	2	9	44	30	15	16	28	16
20	57	58	28	87	28	45	39	10	10	11	10
21	62	42	8	2	8	46	32	39	5	1	5
22	42	57	8	2	8	47	25	32	25	25	25
23	16	57	16	7	16	48	25	55	17	8	17
24	8	52	10	6	10	49	48	28	18	13	18
25	7	38	28	16	28	50	56	37	10	3	10

Central Depot is at $x_o = 30$, $y_o = 40$

Vehicle Capacity is Q = 160

*Unload time is poisson distributed (mean = variance)

TABLE XXVII

50 NODE PROBLEM

TEST PROBLEM #3

MEAN TRAVEL TIME

0	19	28	42	23	19	16	34	29	30
36	17	12	37	24	32	29	23	20	41
41	41	27	29	33	30	36	12	38	37
40	38	14	43	41	50	55	24	21	48
53	39	40	44	32	40	5	14	21	28
34									
19	0	17	26	40	29	23	30	16	32
43	17	28	53	35	46	25	39	36	57
28	35	11	28	37	42	25	12	22	28
42	23	10	54	39	35	40	39	25	54
70	56	56	44	48	53	19	30	17	34
32									
28	17	0	21	47	28	36	44	29	22
36	15	32	63	48	45	13	43	45	64
17	20	15	43	52	55	37	25	25	13
31	30	16	49	26	25	32	41	20	45
76	63	60	58	49	51	26	38	32	28
19									
42	26	21	0	62	46	45	45	28	40
54	33	48	76	57	63	30	60	59	80
12	31	17	47	57	65	33	35	14	23
47	21	30	68	41	15	17	58	38	62
92	78	77	59	67	69	41	53	37	46
35									
23	40	47	62	0	27	28	47	48	42
40	34	18	21	24	25	45	12	12	20
61	56	48	40	37	24	54	31	59	55
48	58	35	39	52	70	75	18	34	51
32	19	22	52	20	32	24	12	38	36
48									
19	29	28	46	27	0	33	51	42	17
20	16	14	45	39	20	22	20	30	41
42	33	35	46	49	43	51	27	47	33
24	49	22	28	28	50	57	16	11	32
53	42	36	61	24	27	17	21	37	12
23									
16	23	36	45	28	33	0	22	24	43
50	29	24	35	15	44	40	32	20	44
48	52	30	16	19	22	29	13	38	47
53	35	23	56	53	55	59	35	35	62
55	41	47	31	42	52	19	21	13	42
46									

TABLE XXVII (continued)

34	30	44	45	47	51	22	0	19	58
67	42	43	50	28	63	52	52	38	63
51	62	33	10	20	35	16	26	34	55
68	28	35	75	66	56	58	55	50	79
72	58	66	17	62	71	36	41	16	59
59									
29	16	29	28	48	42	24	19	0	45
57	31	38	57	36	58	38	50	41	65
34	47	17	21	33	43	11	19	18	39
56	14	23	67	53	40	42	51	39	68
77	62	66	34	59	66	30	39	14	48
45									
30	32	22	40	42	17	43	58	45	0
17	18	28	60	52	31	12	35	44	55
33	19	34	54	60	57	54	34	45	22
13	49	25	31	13	40	48	30	11	26
68	56	49	70	37	34	28	35	44	10
9									
36	43	36	54	40	20	50	67	57	17
0	29	30	58	56	21	27	31	46	49
47	31	47	63	66	59	66	44	59	36
14	62	36	17	21	55	62	25	21	14
61	52	41	78	29	22	34	36	54	12
23									
17	17	15	33	34	16	29	42	31	18
29	0	20	51	39	33	14	31	34	51
30	27	22	39	45	45	40	19	34	23
28	37	10	40	27	38	44	28	11	40
64	50	47	55	37	40	15	26	29	20
20									
12	28	32	48	18	14	24	43	38	28
30	20	0	35	28	23	30	14	19	34
47	42	35	37	39	32	46	22	47	40
35	47	21	34	39	55	61	15	20	42
47	33	31	52	23	31	11	10	31	23
33									
37	53	63	76	21	45	35	50	57	60
58	51	35	0	25	41	62	29	20	20
76	74	61	43	35	18	60	43	70	72
66	68	49	55	70	85	90	35	52	68
25	14	29	49	35	47	39	28	46	54
65									

TABLE XXVII (continued)

24	35	48	57	24	39	15	28	36	52
56	39	28	25	0	45	51	32	15	37
60	62	43	21	15	10	39	25	50	58
61	47	34	59	62	68	72	36	43	68
47	33	42	30	41	53	27	22	25	49
56									
32	46	45	63	25	20	44	63	58	31
21	33	23	41	45	0	38	16	33	30
59	47	52	57	57	46	66	42	64	49
32	66	39	16	39	67	74	11	28	30
42	34	22	71	10	11	31	26	51	23
37									
29	25	13	30	45	22	40	52	38	12
27	14	30	62	51	38	0	39	46	60
24	15	25	50	57	57	47	30	36	14
21	41	20	40	17	31	39	36	14	35
73	61	55	65	43	43	27	37	39	19
9									
23	39	43	60	12	20	32	52	50	35
31	31	14	29	32	16	39	0	19	23
58	50	47	45	44	32	56	33	59	50
40	59	33	30	45	66	73	9	28	42
36	24	20	58	13	24	23	14	41	28
41									
20	36	45	59	12	30	20	38	41	44
46	34	19	20	15	33	46	19	0	27
59	58	44	31	28	16	46	26	54	55
52	52	32	47	55	67	73	25	36	57
38	23	31	42	29	41	22	12	31	40
50									
41	57	64	80	20	41	44	63	65	55
49	51	34	20	37	30	60	23	27	0
78	71	66	55	50	33	71	49	77	71
59	76	52	42	65	87	93	28	49	58
15	8	13	64	23	34	41	29	55	48
62									
41	28	17	12	61	42	48	51	34	33
47	30	47	76	60	59	24	58	59	78
0	23	21	52	62	67	40	37	22	15
40	30	29	62	33	11	18	55	34	54
91	77	74	65	63	64	40	52	41	40
28									

TABLE XXVII (continued)

41	35	20	31	56	33	52	62	47	19
31	27	42	74	62	47	15	50	58	71
23	0	32	61	68	69	55	42	40	11
21	47	32	45	13	28	35	46	25	35
83	72	65	76	53	50	39	48	50	26
12									
27	11	15	17	48	35	30	33	17	34
47	22	35	61	43	52	25	47	44	66
21	32	0	34	44	50	25	20	15	25
43	18	17	59	39	28	32	46	29	57
78	64	64	48	55	59	27	39	23	38
32									
29	28	43	47	40	46	16	10	21	54
63	39	37	43	21	57	50	45	31	55
52	61	34	0	14	28	21	22	37	54
64	32	32	69	64	58	61	49	46	75
65	51	59	18	56	65	31	34	14	54
56									
33	37	52	57	37	49	19	20	33	60
66	45	39	35	15	57	57	44	28	50
62	68	44	14	0	19	32	29	48	63
70	43	39	70	70	69	72	49	51	78
58	45	55	17	54	65	35	34	23	58
63									
30	42	55	65	24	43	22	35	43	57
59	45	32	18	10	46	57	32	16	33
67	69	50	28	19	0	46	33	58	65
65	54	41	60	68	75	79	38	48	71
41	28	40	34	42	54	33	25	32	53
61									
36	25	37	33	54	51	29	16	11	54
66	40	46	60	39	66	47	56	46	71
40	55	25	21	32	46	0	27	22	47
64	15	32	76	61	45	46	58	48	77
81	67	72	29	66	74	38	46	18	57
53									
12	12	25	35	31	27	13	26	19	34
44	19	22	43	25	42	30	33	26	49
37	42	20	22	29	33	27	0	30	36
44	29	12	52	44	45	49	34	26	55
61	47	49	38	42	49	14	23	13	35
36									

TABLE XXVII (continued)

38	22	25	14	59	47	38	34	18	45
59	34	47	70	50	64	36	59	54	77
22	40	15	37	48	58	22	30	0	32
54	10	28	71	49	26	27	58	41	68
89	74	76	48	67	71	39	50	28	50
42									

37	28	13	23	55	.33	47	55	39	22
36	23	40	72	58	49	14	50	55	71
15	11	25	54	63	65	47	36	32	0
28	38	27	50	21	21	28	47	25	42
84	71	66	69	54	53	36	47	43	29
16									

40	42	31	47	48	24	53	68	56	13
14	28	35	66	61	32	21	40	52	59
40	21	43	64	70	65	64	44	54	28
0	59	36	27	11	46	53	34	21	17
71	61	51	80	39	33	37	43	54	15
15									

38	23	30	21	58	49	35	28	14	49
62	37	47	68	47	66	41	59	52	76
30	47	18	32	43	54	15	29	10	38
59	0	30	74	54	33	34	59	44	72
87	73	76	41	68	73	39	49	25	53
47									

14	10	16	30	35	22	23	35	23	25
36	10	21	49	34	39	20	33	32	52
29	32	17	32	39	41	32	12	28	27
36	30	0	47	34	38	43	32	18	47
65	51	50	48	41	46	14	25	21	27
27									

43	54	49	68	39	28	56	75	67	31
17	40	34	55	59	16	40	30	47	42
62	45	59	69	70	60	76	52	71	50
27	74	47	0	35	69	76	25	33	19
52	47	32	84	23	11	41	39	62	24
37									

41	39	26	41	52	28	53	66	53	13
21	27	39	70	62	39	17	45	55	65
33	13	39	64	70	68	61	44	49	21
11	54	34	35	0	39	46	40	22	24
77	67	58	79	46	41	38	46	53	19
10									

TABLE XXVII (continued)

50	35	25	15	70	50	55	56	40	40
55	38	55	85	68	67	31	66	67	87
11	28	28	58	69	75	45	45	26	21
46	33	38	69	39	0	10	63	42	60
99	86	83	71	72	72	49	60	48	48
35									
55	40	32	17	75	57	59	58	42	48
62	44	61	90	72	74	39	73	73	93
18	35	32	61	72	79	46	49	27	28
53	34	43	76	46	10	0	70	49	67
05	92	89	72	78	79	54	66	51	55
42									
24	39	41	58	18	16	35	55	51	30
25	28	15	35	36	11	36	9	25	28
55	46	46	49	49	38	58	34	58	47
34	59	32	25	40	63	70	0	25	36
40	30	22	62	11	19	23	17	43	23
36									
21	25	20	38	34	11	35	50	39	11
21	11	20	52	43	28	14	28	36	49
34	25	29	46	51	48	48	26	41	25
21	44	18	33	22	42	49	25	0	32
62	50	44	62	32	33	19	27	36	12
16									
48	54	45	62	51	32	62	79	68	26
14	40	42	68	68	30	35	42	57	58
54	35	57	75	78	71	77	55	68	42
17	72	47	19	24	60	67	36	32	0
68	61	48	90	37	27	46	48	65	23
29									
53	70	76	92	32	53	55	72	77	68
61	64	47	25	47	42	73	36	38	15
91	83	78	65	58	41	81	61	89	84
71	87	65	52	77	99	105	40	62	68
0	17	22	72	34	44	54	42	66	60
74									
39	56	63	78	19	42	41	58	62	56
52	50	33	14	33	34	61	24	23	8
77	72	64	51	45	28	67	47	74	71
61	73	51	47	67	86	92	30	50	61
17	0	18	59	27	38	40	28	52	50
63									

TABLE XXVII (continued)

40	56	60	77	22	36	47	66	66	49
41	47	31	29	42	22	55	20	31	13
74	65	64	59	55	40	72	49	76	66
51	76	50	32	58	83	89	22	44	48
22	18	0	70	15	24	40	29	57	41
56									
44	44	58	59	52	61	31	17	34	70
78	55	52	49	30	71	65	58	42	64
65	76	48	18	17	34	29	38	48	69
80	41	48	84	79	71	72	62	62	90
72	59	70	0	69	79	47	48	29	70
72									
32	48	49	67	20	24	42	62	59	37
29	37	23	35	41	10	43	13	29	23
63	53	55	56	54	42	66	42	67	54
39	68	41	23	46	72	78	11	32	37
34	27	15	69	0	15	31	24	51	29
43									
40	53	51	69	32	27	52	71	66	34
22	40	31	47	53	11	43	24	41	34
64	50	59	65	65	54	74	49	71	53
33	73	46	11	41	72	79	19	33	27
44	38	24	79	15	0	38	34	59	27
41									
5	19	26	41	24	17	19	36	30	28
34	15	11	39	27	31	27	23	22	41
40	39	27	31	35	33	38	14	39	36
37	39	14	41	38	49	54	23	19	46
54	40	40	47	31	38	0	14	23	26
31									
14	30	38	53	12	21	21	41	39	35
36	26	10	28	22	26	37	14	12	29
52	48	39	34	34	25	46	23	50	47
43	49	25	39	46	60	66	17	27	48
42	28	29	48	24	34	14	0	30	31
40									
21	17	32	37	38	37	13	16	14	44
54	29	31	46	25	51	39	41	31	55
41	50	23	14	23	32	18	13	28	43
54	25	21	62	53	48	51	43	36	65
66	52	57	29	51	59	23	30	0	45
46									

TABLE XXVII (continued)

28	34	28	46	36	12	42	59	48	10
12	20	23	54	49	23	19	28	40	48
40	26	38	54	58	53	57	35	50	29
15	53	27	24	19	48	55	23	12	23
60	50	41	70	29	27	26	31	45	0
17									
34	32	19	35	48	23	46	59	45	9
23	20	33	65	56	37	9	41	50	62
28	12	32	56	63	61	53	36	42	16
15	47	27	37	10	35	42	36	16	29
74	63	56	72	43	41	31	40	46	17
0									

TABLE XXVIII

50 NODE PROBLEM

TEST PROBLEM #3

VARIANCE TRAVEL TIME

0 6 1 1 2 0	0 2 4 2 4	3 2 5 0 6	10 3 7 3 2	2 3 3 0 4	1 2 7 9 7	0 4 7 5 0	2 5 2 4 0	2 3 2 2 5	0 7 1 1 4
0 1 2 9 7 7	0 0 8 1 8	2 2 2 2 9	3 0 3 5 8	2 8 4 8 1	1 5 4 4 8	2 3 4 5 0	0 7 2 8 1	1 2 2 5 1	2 4 0 7 1
3 7	2 0	0 6	1 9	9 2	3 11	7 0	6 0	5 9	5 1
0 1 10 0	4 5 10	2 0 0	4 4 13	8 2 0	6 4 6	6 6 4	1 6 8	4 3 5	2 2 5
10 13 1 5 3 7	3 2 0 4 10	1 8 2 0 9	0 3 2 10 5	1 5 5 6 15	5 1 1 2 10	3 2 2 2 0	3 14 8 5 8	3 9 1 9 2	3 11 1 11 0
2 5 0 11 0 9	2 3 10 12 3	9 3 2 4 2	1 4 0 6 7	0 5 7 1 0	5 2 1 7 7	1 8 11 2 0	10 0 2 0 0	5 0 12 0 8	1 0 9 10 3
1 4 9 2 8 3	1 0 1 6 9	3 1 7 2 1	5 6 0 6 12	5 5 4 2 2	0 0 4 8 5	0 5 12 0 0	10 0 2 2 0	10 4 8 0 6	2 1 3 2 0
0 2 3 8 8 5	2 2 2 7 6	7 1 4 5 11	3 3 2 5 4	1 1 4 4 5	0 9 0 7 0	0 9 2 4 0	2 7 2 5 2	4 2 8 3 2	7 6 9 9 10

TABLE XXVIII (continued)

2	0	6	3	10	10	2	0	4	3
7	1	1	11	2	13	9	12	3	6
12	3	1	0	4	0	2	1	5	12
4	4	0	11	3	2	3	4	9	16
10	2	4	1	6	13	6	3	0	9
9									
2	1	5	3	5	10	4	4	0	2
12	7	8	9	5	3	2	3	5	10
0	8	0	4	5	10	2	0	0	8
10	2	5	11	11	9	0	12	5	14
10	11	13	3	5	5	6	7	3	10
2									
0	2	5	3	1	2	7	3	2	0
4	0	1	13	8	0	0	0	1	13
7	1	0	9	13	3	1	3	0	2
0	1	2	4	2	8	2	1	1	0
16	4	.8	12	5	3	0	7	6	1
0									
6	1	7	13	5	4	2	7	12	4
0	6	6	4	11	2	0	5	4	4
5	0	4	12	14	14	14	9	3	2
2	0	3	2	3	7	7	3	0	2
2	7	1	11	5	1	1	6	12	2
0									
2	0	0	2	3	0	2	1	7	0
6	0	4	4	2	6	1	1	4	12
3	3	1	4	10	10	3	2	6	5
0	5	0	2	3	8	2	3	0	2
14	3	6	5	1	3	0	4	1	2
4									
2	2	6	8	3	1	1	1	8	1
6	4	0	6	1	3	6	1	3	1
0	6	7	5	7	3	5	5	8	5
4	7	4	8	9	8	1	0	4	1
8	3	4	1	2	7	1	0	6	2
5									
3	0	9	3	4	6	3	11	9	13
4	4	6	0	4	8	9	3	0	4
4	14	4	0	2	1	13	3	4	4
7	2	2	1	6	10	19	6	9	6
6	2	6	7	2	1	8	6	7	1
1									

TABLE XXVIII (continued)

3	8	2	5	5	5	1	2	5	6
11	2	1	4	0	2	5	2	0	5
1	14	7	2	2	1	0	4	3	3
5	11	1	13	1	12	7	3	1	14
3	0	9	0	10	2	5	3	3	9
0									
2	5	11	1	2	0	9	13	3	0
2	6	3	8	2	0	0	2	0	2
11	1	0	7	4	0	0	5	14	12
5	16	7	1	2	1	10	1	5	7
5	4	4	7	1	1	6	4	4	0
6									
4	3	0	2	8	5	9	9	2	0
0	1	6	9	5	0	0	0	9	13
0	0	3	11	9	13	3	1	3	3
0	4	2	8	0	1	1	8	2	2
1	2	11	15	9	7	5	5	3	4
1									
5	7	0	14	0	0	7	12	3	0
5	1	1	3	2	2	0	0	1	3
8	11	1	7	7	5	2	1	3	9
2	0	4	4	7	15	17	0	6	9
8	3	2	7	0	5	5	2	9	0
8									
3	2	9	9	0	4	2	3	5	1
4	4	3	0	0	0	9	1	0	5
13	8	1	7	5	3	6	1	4	1
12	8	0	9	10	12	9	3	1	5
6	1	1	8	4	1	2	1	7	6
3									
7	4	1	11	0	1	6	6	10	13
4	12	1	4	5	2	13	3	5	0
7	11	11	10	8	7	1	6	7	5
12	6	9	1	9	1	5	2	2	7
3	0	2	12	0	7	5	2	6	6
12									
1	2	0	1	0	9	3	12	0	7
5	3	0	4	1	11	0	8	13	7
0	4	0	6	7	3	7	6	4	1
4	1	1	0	0	1	3	13	6	0
4	7	8	13	9	14	1	1	1	7
0									

TABLE XXVIII (continued)

4	8	4	0	10	1	2	3	8	1
0	3	6	14	14	1	0	11	8	11
4	0	2	0	1	11	11	6	1	0
4	2	1	3	2	4	5	8	2	4
15	11	13	17	1	1	2	1	1	0
0									
5	2	2	2	2	7	4	1	0	0
4	1	7	4	7	0	3	1	1	11
0	2	0	1	7	5	5	2	0	3
9	0	1	13	7	3	6	3	1	1
10	4	8	9	7	14	0	0	4	4
7									
7	3	4	2	0	0	2	0	4	9
12	4	5	0	2	7	11	7	7	10
6	0	1	0	1	4	2	2	0	12
6	3	2	4	14	0	5	5	3	12
5	1	2	3	10	1	4	0	1	6
0									
3	4	8	5	7	4	4	4	5	13
14	10	7	2	2	4	9	7	5	8
7	1	7	1	0	4	5	2	7	14
2	10	3	14	15	5	16	2	4	3
4	2	2	3	3	13	7	7	5	2
3									
7	4	6	1	1	4	0	0	10	3
14	10	3	1	1	0	13	5	3	7
3	11	5	4	4	0	5	7	2	15
7	5	1	3	14	14	8	6	0	12
8	6	4	4	3	10	1	3	3	10
13									
7	4	6	2	11	12	2	2	2	1
14	3	5	13	0	0	3	2	6	1
7	11	5	2	5	5	0	2	1	6
8	0	6	12	12	10	8	2	2	5
19	11	0	5	6	9	9	3	3	10
11									
2	2	1	8	2	2	2	1	0	3
9	2	5	3	4	5	1	1	1	6
6	6	2	2	2	7	2	0	6	6
0	2	2	5	7	9	4	6	1	9
5	11	3	8	5	10	0	2	3	7
2									

TABLE XXVIII (continued)

2	2	4	1	12	8	8	5	0	0
3	6	8	4	3	14	3	3	4	7
4	1	0	0	7	2	1	6	0	4
7	1	6	11	1	4	4	13	2	3
1	1	14	11	11	13	7	3	1	7
0									
1	0	2	1	9	3	9	12	8	2
2	5	5	4	3	12	3	9	1	5
1	0	3	12	14	15	6	6	4	0
5	9	4	0	1	3	5	2	4	4
18	12	3	16	10	8	7	10	8	1
2									
1	9	1	5	11	2	8	4	10	0
2	0	4	7	5	5	0	2	12	12
4	4	9	6	2	7	8	0	7	5
0	0	0	3	1	3	8	1	2	3
7	12	3	15	6	3	3	6	7	1
2									
2	1	5	4	12	6	7	4	2	1
0	5	7	2	11	16	4	0	8	6
1	2	0	3	10	5	0	2	1	9
0	0	2	8	0	2	7	0	6	3
8	12	12	2	9	6	1	9	6	3
6									
0	2	0	0	4	2	5	0	5	2
3	0	4	2	1	7	2	4	0	9
1	1	1	2	3	1	6	2	6	4
0	2	0	11	3	2	4	3	1	7
10	12	0	0	8	6	0	3	4	4
1									
3	5	4	10	6	6	5	11	11	4
2	2	8	1	13	1	8	4	9	1
0	3	13	4	14	3	12	5	11	0
3	8	11	0	8	0	10	1	4	0
10	7	4	13	4	2	4	3	0	5
0									
0	8	2	6	1	2	4	3	11	2
3	3	9	6	1	2	0	7	10	9
0	2	7	14	15	14	12	7	1	1
1	0	3	8	0	3	11	2	4	1
7	14	6	11	3	7	3	6	7	0
2									

TABLE XXVIII (continued)

9 7 1 3 20 2	4 8 4 2 12	4 8 3 2 13	2 10 0 0 7	7 12 5 3 0	8 1 14 0 13	7 1 10 1 5	2 15 9 10 9	9 12 4 2 3	8 1 3 3 0
5 7 3 8 2 3	5 2 5 7 17	6 1 6 4 1	2 19 5 10 14	2 7 16 11 0	0 10 8 1 15	4 1 8 0 12	3 17 4 12 1	0 9 4 3 1	2 5 5 9 12
4 3 13 1 1 8	8 3 8 0 0	6 0 3 3 0	5 6 5 1 2	0 3 2 2 1	2 1 6 10 1	5 8 2 12 4	4 0 6 0 1	12 3 13 1 3	1 2 2 8 4
2 0 6 2 5 1	5 0 2 6 8	3 4 1 1 1	9 9 3 4 8	0 1 4 4 6	0 5 0 2 1	3 2 2 3 3	9 6 1 1 2	5 1 2 0 3	1 2 4 2 1
1 2 0 3 0 5	7 2 4 3 15	2 1 1 7 6	11 6 12 0 0	10 14 3 1 7	2 7 12 3 0	9 2 5 9 4	16 9 9 8 9	14 5 3 2 8	0 7 4 0 4
2 2 4 7 0 15	7 14 15 8 1	10 8 10 10 1	3 6 5 10 1	0 3 4 7 6	8 5 8 20 4	8 1 19 2 11	10 8 5 1 5	10 6 1 5 11	16 3 18 0 1
4 7 7 12 1 15	8 3 11 12 0	10 3 4 12 0	10 2 1 7 12	3 0 2 14 3	9 4 6 12 1	6 2 11 17 0	2 3 11 0 4	11 1 1 8 0	4 0 12 15 1

TABLE XXVIII (continued)

6	9	0	9	2	1	11	4	13	8
1	6	4	6	9	4	11	2	1	2
8	13	8	2	2	4	0	3	14	3
3	12	0	4	6	13	1	0	1	6
1	0	0	16	1	3	8	3	13	7
7									
2	8	13	5	7	12	4	1	3	12
11	5	1	7	0	7	15	7	8	12
13	17	9	3	3	4	5	8	11	16
15	2	0	13	11	7	14	2	8	0
1	12	16	0	7	6	10	11	3	16
7									
4	1	0	15	0	2	5	6	5	5
5	1	2	2	10	1	9	0	4	0
9	1	7	10	3	3	6	5	11	10
6	9	8	4	3	0	0	1	6	7
6	3	1	7	0	1	0	2	8	3
1									
7	8	6	10	7	5	0	13	5	3
1	3	7	1	2	1	7	5	1	7
14	1	14	1	13	10	9	10	13	8
3	6	6	2	7	13	15	1	1	0
4	1	3	6	1	0	1	3	12	5
9									
0	0	4	0	0	0	0	6	6	0
1	0	1	8	5	6	5	5	2	5
1	2	0	4	7	1	9	0	7	7
3	1	0	4	3	5	12	4	3	4
11	0	8	10	0	1	0	3	4	2
4									
0	1	8	8	0	0	2	3	7	7
6	4	0	6	3	4	5	2	1	2
1	1	0	0	7	3	3	2	3	10
6	9	3	3	6	9	1	1	2	9
5	4	3	11	2	3	3	0	4	2
5									
5	1	5	2	8	6	2	0	3	6
12	1	6	7	3	4	3	9	7	6
1	1	4	1	5	3	3	3	1	8
7	6	4	0	7	3	1	3	3	8
11	0	13	3	8	12	4	4	0	4
1									

TABLE XXVIII (continued)

4	1	5	0	3	0	10	9	10	1
2	2	2	1	9	0	4	0	6	6
7	0	4	6	2	10	10	7	7	1
1	3	4	5	0	0	12	4	1	4
1	1	7	16	3	5	2	2	4	0
0									
0	7	0	7	9	3	5	9	2	0
0	4	5	1	0	6	1	8	3	12
0	0	7	0	3	13	11	2	0	2
2	6	1	0	2	2	3	8	1	5
15	15	7	7	1	9	4	5	1	0
0									

VITA

Yahia Zare-Mehrjerdi

Candidate for the Degree of

Doctor of Philosophy

Thesis: A GOAL PROGRAMMING MODEL OF THE STOCHASTIC VEHICLE ROUTING PROBLEM

Major Field: Industrial Engineering and Management

Biographical:

Personal Data: Born in Yazd, Iran, March 30, 1953, the son of Mr. Mohammad Hasan Zare-Mehrjerdi and Mrs. Sedigheh Hashemi. Married to Mary Jo Fenske on January 9, 1982.

Education: Graduated from Iranshahr High School, Yazd, Iran, in May, 1971; received Bachelor of Science Degree in Mathematics from Iranian National University in May, 1976; received Bachelor of Science Degree in Civil Engineering from Texas A & I University in December, 1980; received Master of Science Degree in Operations Research from St. Mary's University in December, 1983; completed requirements for the Doctor of Philosophy Degree at Oklahoma State University in December, 1986.

Professional Experience: Teaching Associate, Department of Mathematics, Oklahoma State University, September, 1982 to May, 1983; Teaching Assistant, School of Industrial Engineering and Management, Oklahoma State University, September, 1983 to June, 1986.

Professional Organizations: Institute of Industrial Engineering, Institute of Management Sciences.