A HEURISTIC SOLUTION PROCEDURE FOR THE INVENTORY ROUTING PROBLEM

by

Remzi Yüksel
B.S. in I.E., Boğaziçi University, 1983

Submitted to the Institute for Graduate Studies in Science and Engineering in partial fulfillment of the requirements for the degree of Master of Science in Industrial Engineering

Boğaziçi University

1985

A HEURISTIC SOLUTION PROCEDURE
FOR
THE INVENTORY ROUTING PROBLEM

APPROVED BY

Doç. Dr. İlhan OR
(Thesis Supervisor) .....

Prof. Dr. İbrahim Kavrakoğlu .....

Doç. Dr. Ceyhan Uyar .....

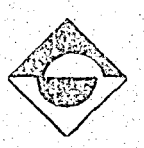
DATE OF APPROVAL

ACKNOWLEDGEMENTS

I wish to express my deepest gratitude to Doç. Dr. İlhan Or for introducing me this topic, and for his invaluable guidance and supervision throughout this thesis.

I also thank Prof. Dr. İbrahim Kavrakoğlu and Doç. Dr. Ceyhan Uyar for their interest and evaluation as members of the thesis commitee.

At last I want to thank Alper Oysal for the painstaking typing.



ABSTRACT

In this study, it is aimed to present a heuristic solution procedure for the Inventory Routing Problem, which is an important extension of the Vehicle Routing Problem. The Inventory Routing Problem has the same basic structure as the Vehicle Routing Problem. However, it leads to a very complex and large mathematical programming formulation since additional consideration is given to vehicle travel time constraints as well as to inventory fluctuations and stockouts.

A computer program is developed in order to obtain computational results for the suggested heuristic solution procedure.

ÖZET

Bu çalışmada, Yol Atama Probleminin önemli bir uzantısı olan Envanterli Yol Atama Problemi için sezgisel bir yöntemin geliştirilmesi amaçlanmıştır. Envanterli Yol Atama Problemi, Yol Atama Problemi ile temelde aynı yapıya sahip olmasına karşın daha karmaşık ve geniş matematiksel programlama formülasyonunu gerektirir. Problemde araç yolculuk süresi, envanter dalgalanmaları ve yokluk durumu gözönüne alınmıştır.

Önerilen sezgisel yöntemin sayısal sonuçlarını elde etmek üzere bir bilgisayar programı geliştirilmiştir.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

I. INTRODUCTION

"The provision of goods and services from a supply point to a demand point" may be defined as logistics. A complete logistics system covers the entire process of moving raw materials and input requirements from suppliers to plants, the conversion of the inputs into products at certain plants, the movement of the products to various warehouses or depots, and the eventual delivery of these products to the final consumers. The distribution activities of this system include all movements and storage of goods "downstream" from the plants. The last step in these movements (from distributic centers to customers), which may be called local delivery, is the most costly link of the distribution chain. These movements need an effective distribution management which presents a variety of distribution problems.

The importance of distribution problems is evident from the magnitude of the associated distribution costs. Surveys by Kearny show that physical distribution costs account for

about 16% of the sales value of a product. Of this, about a fourth is due to downstream distribution of the final product from distribution centers to customers. A detailed study of the distribution problem involves an explicit consideration of vehicle routing and fleet size and mix issues.

The original vehicle routing problem is a distribution management problem. Its application areas range from school bus routing, to blood banking, to delivery of consumer goods. In all cases, the basic components of the problem are a fleet of vehicles with fixed capacities and a set of demands for transporting certain products or objects between specified pick up and delivery points. The aim is to determine which of the demands will be satisfied by each vehicle and which route each vehicle will follow in servicing its assigned demands. Of course, cost minimization is the primary objective. Cost items include essentially fuel, personnel and vehicle depreciation. The routes are designed so that:

(i) each demand location is served by exactly one vehicle (demand constraint);

(ii) the total demand on each route is less than or equal to the capacity of the vehicle assigned to that route (vehicle capacity contraint);

(iii) each route begins and ends at a central depot (routing constraint).

In this study, we will consider an extension of the above described vehicle routing problem. In this extended problem vehicle travel time constraints are incorporated besides the vehicle capacity constraints, then requirements at the delivery points are relaxed into random variables (rather than the

deterministic parameters assumed in the original vehicle routing problem) and furthermore, store of goods (from one delivery period to the next) is allowed. The new problem is called the Inventory Routing Problem.

To summarize the remainder of the study, in section 2 we state the problem. Section 3 introduces a heuristic solution procedure for inventory routing problem. Section 4 solves an example problem using the computer program of the procedure.

II. PROBLEM STATEMENT

In this chapter, we define and explain the inventory routing problem, present application areas and make a literature survey.

2.1 Problem Definition

It is a single depot problem. There are N nodes where the first one is the supply node (depot), and the remaining ones are the demand nodes. The storage capacity SC_i is known for all demand nodes $i = 2,\ldots,N$. The daily demand ξ_q^i of demand node i in day q is independent and identically distributed random variable over the planning horizon. Expected values $R_i = E[\xi_q^i]$ and variances $\sigma_i^2 = VAR[\xi_q^i]$ are given for all $i = 2,\ldots,N$.

The locations of nodes are represented by coordinate points in a plane. Then, the distances between all pairs of nodes, which construct the road network connecting all nodes, can be determined. Products or objects are to be transported from the

supply node to the demand nodes, using at most K vehicles with capacities VC. The term "trip" is used for any sequence of nodes that are serviced by one vehicle in one loading (see Figure 2.1). It starts and ends with the supply node. The term "route" is used for any collection of trips served by the same vehicle in the same delivery period (see Figure 2.1).

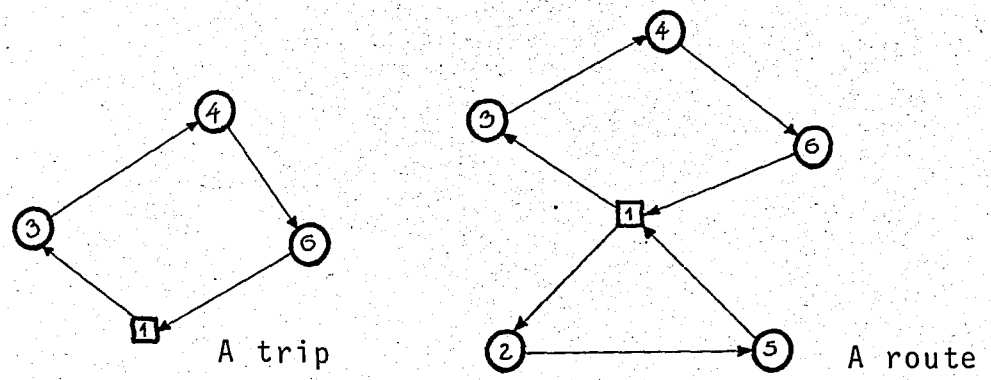


Figure 2.1

Maximum length of each route is T time units which is the vehicle travel time constraint. Also, additional vehicles may be hired, however, that alternative is highly undesirable and costly. Cost of hiring additional vehicles is h TL per unit time, unit capacity for a trip.

Let the planning horizon be Q delivery periods (a day corresponds to a delivery period), and let $I_{i,q}$ be the amount of goods left in inventory from day q to day q+1 at node i, for q = 0,...,Q-1. The amount in inventory just before the first day of the planning horizon is $I_{i,0}$ for the demand node i. Suppose that deliveries to the demand nodes replenish inventories up to their capacities, and that deliveries made to node i in day q may be used to meet the requiriments of day q. Then, $I_{i,q}$ is a random variable dependent not only on daily

demands, but also on the date of last delivery. Let t_i be the date of last replenishment. t_i is less than or equal to zero where it implies last delivery being (-t+1) days before the start of the planning horizon. Then, $I_{i,q}$ can be defined in terms of t_i and distribution function of daily demand as:

$$I_{i,q}(t_i) = \max\left[\left(SC_i - \sum_{r=t_i}^{q} \xi_r^i\right), 0\right]$$

where ξ_r^i with $r \leq 0$ denoting demand in the days before the start of the planning horizon. The amount delivered, $A_{i,q}$, is also a random variable dependent on daily demands and the date of last delivery. A delivery is made to node i on day q is;

$$A_{i,q}(t_i) = \min\left[\sum_{r=t_i}^{q-1} \xi_r^i , SC_i\right]$$

A stockout is said to occur on day q if the amount of goods available at node i on day q is not enough to satisfy the daily demand ξ_q^i. A stockout implies certain penalties and an immediate emergency shipment to replenish the inventory up to its capacity. Assume that emergency shipments do not use the vehicles mentioned before and have relatively high costs.

There are two types of costs in the above system:

1) Operating costs: These are the costs associated with deliveries to the demand nodes. Operating costs are direct delivery costs (cost of fuel, crew, wear and tear, etc.), capital cost and cost of hiring additional vehicles. Letting direct delivery costs and capital costs be the linear functions

of distance traveled, CK is the direct delivery plus capital cost per unit distance traveled for each vehicle.

2) Stockout costs: These are the costs associated with not making deliveries to the demand nodes. A stockout cost SL_i is incurred if there is a stockout at demand node i anytime during the planning horizon. It will also be assumed that the date of last delivery to node i, t_i is known for all nodes $i = 2,\ldots,N$.

Assume that ;

$$P\{\sum_{q=1}^{Q} \xi_q^i > SC_i\} \simeq 0 \qquad i = 2,\ldots,N.$$

This quite realistic assumption means; if any demand node i is replenished (through regular deliveries or emergency shipments) at any time q during the planning horizon, a stockout at node i is not possible in the time interval $|q, Q|$. This important assumption has two implications:

i) with no loss of generality, the problem may be restricted to at most one delivery to any demand node during the planning horizon;

ii) the stockout cost will be incurred at most once during the planning horizon at any demand node.

Now, the expected stockout cost associated with any decision policy can be determined. Suppose the decision policy is to make deliveries to node i on day q of the planning horizor Then, a stockout may occur on days 1,..., q-1 and only if the event

$$\{\sum_{r=t_i}^{q-1} \xi_r^i > SC_i\}$$

occurs. Furthermore, the way parameter t_i is defined implies that event

$$\{ \sum_{r=t_i}^{0} \xi_r^i \leq SC_i \}$$

has occured. (Otherwise, the logic of the model would imply an emergency shipment to node i at some time $t \in (t_i, 0)$, however, this contradicts the assumption that t_i is the date of last replenishment of node i). Then, the expected stockout cost is;

$$SL_i \times P\{ \sum_{r=t_i}^{q-1} \xi_r^i > SC_i \mid \sum_{r=t_i}^{0} \xi_r^i \leq SC_i \}$$

Note that the cumulative distribution of the random variable $\Sigma \xi_r^i$ is necessary for the above cost term. We will assume that it is possible to estimate the cumulative distribution function, $\phi_i^m(b)$, in terms of $E[\xi_r^i]$, $VAR[\xi_r^i]$ and m, which leads to

$$\phi_i^m(b) = P\{ \sum_{r=1}^{m} \xi_r^i \leq b \} = f_i(b, R_i, \sigma_i, m)$$

The objective is to minimize total stockout and delivery costs of all demand nodes over a fixed planning horizon of Q delivery periods.

2.2 Application Areas, Sources of Difficulty, Analysis of Assumptions

The inventory routing problem can be applied mainly to distribution problems, where it is necessary to use inventory and random demands. An example is the delivery of the gasoline to automotive service stations. Also, it can be used in the

industrial gas industry, where the gas producers themselves install tanks at customer locations and determine the replenishment frequency and delivery size. Another potential application is the allocation of a perishable product such as blood, where the supply to various locations in a particular region is coordinated by a regional center.

The importance of the interrelationships between inventory and distribution planning is easily recognized from the above examples. However, the inventory routing problem described in 2.1 leads to a very complex and large mathematical programming formulation. It is almost impossible to generate an exact solution procedure for even relatively very small values of N.K. and Q. Also, the size of the problem grows exponentially as the number of nodes, vehicles and delivery periods increase. Due to the complexity of the inventory routing problem, it can only be solved heuristically.

In the problem definition, some assumptions are used. One of them, which has two important implications, is:

$$P\{\sum_{q=1}^{Q} \xi_q^i > SC_i\} \simeq 0 \qquad i = 2,\ldots,N.$$

Assuming a very small value for the above probability expression, it may be avoided to replenish any demand node more than once during the planning horizon. In other words, if any demand node i is replenished at any time q during the planning horizon, a stockout at node i, which needs a replenishment again, is not possible in the time interval [q, Q]. Without any loss of generality, the delivery to any demand node and the occurrence of stockout at any demand node are limited by this assumption.

Otherwise, our problem becomes evenmore complex; since, all the demand nodes must be checked for each day of the planning horizon whether it will be serviced or not.

2.3 Literature Survey

The literature in this area is quite scarce, however, there have been some pioneer works in the vehicle routing problem literature. A simulation study for a deterministic inventory/routing system was reported in Assad et al. (1982). Random demands have been treated in Stewart and Golden (1982) and Bodin et al. (1982). They described a "Stochastic Vehicle Routing Problem" where inventory considerations are incorporated but planning horizon is limited to a single delivery period and vehicle travel time constraints are put on trips (not on routes). Federgruen and Zipkin (1982) described "A Combined Vehicle Routing and Inventory Allocation Problem", where a more detailed inventory submodel is considered, however, vehicle travel time constraints are ignored and also, the planning horizon is limited to a single delivery period. Also, Cook and Russell (1978) undertook some simulation studies of vehicle routing problems with random delivery sizes.

III. A HEURISTIC SOLUTION PROCEDURE FOR INVENTORY ROUTING PROBLEM

Our problem is to decide which nodes to be visited in delivery period q. As described in 2.1, there are two types of costs in the system; and, our heuristic solution procedure is based on a trade off between these two types of costs. Then, inventory routing problem denotes a trade off between "stockout costs" and "delivery costs". Any given node i will be visited at the "proper" time during the planning horizon if its stockout cost weighs more heavily than its delivery cost, otherwise, it is not visited. Stockout costs of nodes are independent of each other since a delivery or a nondelivery to a demand node i does not affect the stockout cost of node j ($j \neq i$). Therefore, these costs are relatively easy to determine. But delivery costs of nodes are strongly dependent and impossible te determine before the actual routes are generated. The difficulty in the determination of delivery costs of nodes is a complicating factor for our problem.

(These costs must be determined or approximated in some way in order to apply the trade off procedure.) The basic idea of our procedure is to decompose the overall problem into three subproblems in such a way that it will be possible to treat the delivery costs also as independent of each other. Then we will iterate among these subproblems until a satisfactory solution is generated. At every iteration, the previous routing is used as a base for the current delivery costs of nodes.

The first step, in this procedure, is generation of Q sets of seed nodes (one set for each day in the planning horizon), in order to construct an initial routing. These seed nodes are determined with respect to stockout costs only. The second step is the subproblem one, where minimum cost trips are generated satisfying vehicle capacity constraints for each set of nodes. Packing the generated trips into routes satisfying vehicle travel time constraints and minimizing the use of additional hired vehicles is the third step. Therefore, the second subproblem is a packing problem. In subproblem three, new marginal delivery (operating) costs are approximated based on generated routes. Then an assignment is done using cost parameters. This is the last step where Q new sets of nodes are generated.

Now let us explain these steps with the details.

3.1 Generation of Seed Nodes

An initial assignment of demand nodes is made to days and a set of seed nodes is generated for each day of the planning horizon. This assignment is based on stockout costs only. Stockout probabilities must be determined in order to generate stockout costs of nodes given a delivery on day q. As defined in 2.1, the expected stockout cost is

$$SL_i \times P\{ \sum_{r=t_i}^{q-1} \xi_r^i > SC_i \mid \sum_{r=t_i}^{0} \xi_r^i \leq SC_i \}$$

Then, the probability of a stockout at node i, before day q is

$$P_{i,q} = P \{ \sum_{r=t_i}^{q-1} \xi_r^i > SC_i \mid \sum_{r=t_i}^{0} \xi_r^i \leq SC_i \}$$

Unfortunately a closed form or simple and accurate approximations for the above probability expression is not easy to determine. This is however beyond the scope of this study. We will assume that stockout probabilities $P_{i,q}$ will somehow be made available. Then, the assignment is done in the following way:

Define set V_q, $q = 1,\ldots,:$, where V_q is the set of nodes to be visited on the qth day of the planning horizon. Then, starting with $q = 1$ and incrementing up to $q = Q$, execute the following:

For every $i = 2,\ldots,N$, not already assigned to a set, assigned node i to V_q if

$$SL_i \times P\{ \sum_{r=t_i}^{q} \xi_r^i > SC_i \} \simeq SL_i \times P_{iq} > \bar{S}$$

where $\bar{S}$ is a predetermined constant.

The above stockout costs are used for initial step and will be updated later in the procedure, as more information becomes available about delivery costs.

3.2 Routing Subproblem

In this step, a set of minimum cost trips, satisfying vehicle capacity constraints, is generated for each of the sets V_q, $q = 1,\ldots,Q$. Assume that vehicle capacity constraints in our problem are on total expected amounts to be delivered on any trip (rather than the actual amounts delivered). Hopefully this assumption will not cause any major problems in applications. Then, this step is reduced to solving Q original vehicle routing problems.

Any one of the existing routines can be used in order to solve these vehicle routing problems. However, the speed of the routine is important for us, because this subproblem will be executed over and over again (Q times for each iteration). The solutions generated at a particular iteration will not be the final routes but just a basis for the determination of marginal delivery cost (see 3.4 for details). Then, a modified Clark and Wright algorithm (Golden and Magnanti, 1977) is used which is a simple and fast routine.

A Modified Clarke-Wright Algorithm:

This is a heuristic solution procedure based on Clarke and Wright's Savings Method (Clarke and Wright, 1964). Savings

method is an "exchange" procedure in the sense that at each step one set of tours is exchanged for a better set of tours. Initially, we suppose that every two demand nodes i and j are supplied individually from two vehicles (refer to Figure 3.1 below).

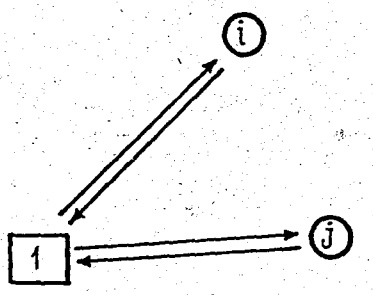


Figure 3.1 Initial Set up

Now if instead of two vehicles, we used only one, then we would experience a savings in travel distance of

$(2d_{1i} + 2d_{1j}) - (d_{1i} + d_{1j} + d_{ij}) = d_{1i} + d_{1j} - d_{ij}$ (see Figure 3.2)

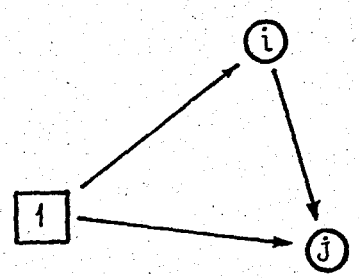


Figure 3.2 Nodes i and j have been linked

For every possible pair of demand nodes i and j there is a corresponding saving s_{ij}. We order these savings from greatest

to least and starting from the top of the list we link nodes i and j where s_{ij} represents the current maximum saving unless the problem constraints are violated. Then, at each step in the Clarke-Wright algorithm we seek the greatest saving s_{ij} subject to the following restrictions:

(i) nodes i and j are not already on the same trip;
(ii) neither i nor j are interior to an existing tour;
(iii) vehicle capacity is not exceeded;
(iv) maximum number of drops is not exceeded (in a sense, maximum trip time).

Nodes i and j are linked together to form a new trip, and the procedure is repeated until no further savings are possible.

The basic Clarke-Wright algorithm is modified in three ways:

1. By using a route shape parameter β to define a modified saving

$$S_{ij} = d_{1i} + d_{1j} - \beta d_{ij}$$

and placing greater emphasis on the distance between nodes i and j rather than their position relative to the supply node as the parameter increases from zero;
2. by considering savings only between nodes that are "close" to each other;
3. by storing savings S_{ij} in a heap structure to reduce comparison operations and ease access.

Let NT_q be the number of trips generated, TD_{jq} be the total duration and TA_{jq} be the expected total amount delivered on trip j, $j = 1,\ldots NT_q$ for each $q=1,\ldots,Q$, at the end of this step.

3.3 Packing Subproblem

NT_q trips generated in the routing subproblem satisfy vehicle capacity constraints only. In this step, these trips are packed into K routes satisfying vehicle travel time constraints for each of the sets V_q, $q = 1,\ldots,Q$. The objective of these packing subproblems is to minimize the use of additional hired vehicles for each day of the planning horizon. The formulation is obtained as follows.

Let NT_q, TD_{jq}, TA_{jq} be the parameters obtained from 3.2. and let h, T be the parameters defined in 2.1. Now, define variables y_{ikq}, y_{joq} for $k = 1,\ldots,K$; $j = 1,\ldots,NT_q$; $q = 1,..,Q$ as follows:

$$y_{ikq} = \begin{cases} 1 & \text{if trip j on day q is assigned to vehicle K} \\ 0 & \text{otherwise} \end{cases}$$

$$y_{joq} = \begin{cases} 1 & \text{if trip j on day q is assigned to a hired vehicle} \\ 0 & \text{otherwise} \end{cases}$$

Then, for each $q = 1,\ldots,Q$ the problem is:

$$\text{minimize} \sum_{j=1}^{NT_q} \bar{c}_{jq} \cdot y_{joq}$$

(P1)

$$\text{subject to :} \quad \sum_{k=0}^{K} y_{jkq} = 1 \qquad j = 1,\ldots, NT_q,$$

$$\sum_{j=1}^{NT_q} TD_{jq}\, y_{jkq} \leq T \qquad k = 1,\ldots,K,$$

$$y_{jkq} \in \{0,1\} \quad j = 1,\ldots, NT_q \; ; \; k = 0,\ldots,K$$

where $\bar{c}_{jq} = h \cdot TA_{jq} \cdot TD_{jq}$.

Problem (P1) is a Generalized Assignment Problem. Notice that a zero minimum value for (P1) implies the ability of serving all nodes in V_q with the available fleet, whereas a nonzero minimum value implies the insufficiency of the fleet and thus a reduction in V_q. The above problem can be formulated in a general way. This formulation is :

(P2)

$$\text{minimize } \bar{Z} = \sum_{i\in I} \sum_{j\in J} c_{ij}\, x_{ij} \qquad (1)$$

$$\text{subject to : } \sum_{j\in J} r_{ij}\, x_{ij} \leq b_i \quad \text{for all } i\in I, \qquad (2)$$

$$\sum_{i\in I} x_{ij} = 1 \quad \text{for all } j\in I \qquad (3)$$

$$x_{ij} \in \{0,1\} \quad \text{for all } i\in I,\ j\in J \qquad (4)$$

where $I \equiv \{0,1,\ldots,K\}$, $J \equiv \{1,\ldots,NT_q\}$. It can be described as the problem of optimally assigning NT_q trips to K+1 vehiches: if trip j is assigned to vehicle i, a cost c_{ij} is involved and amount r_{ij} of travel time is required for vehicle i; each vehicle has a total amount b_i of available travel time. We will assume, without any loss of generality, that c, r and b are positive integers. These positive integers can be defined for our purpose as follows:

$$r_{ij} = TD_{jq} \qquad \text{for } i = 0,\ldots,K \text{ , } j = 1,\ldots, NT_q$$

$$c_{ij} = \begin{cases} \bar{c}_{jq} & \text{for } i = 0 \text{ ; } j = 1,\ldots, NT_q \\ 0 & \text{otherwise} \end{cases}$$

$$b_i = \begin{cases} T & \text{for } i = 1,\ldots,K \\ \text{a large number} & \text{for } i = 0 \text{ (hired vehicle)} \end{cases}$$

Let us first formulate (P2) as a maximization problem as proposed by Martello and Toth (1981): define any vector (t_j) of integer constants such that:

$$t_j > \max_{i \in I} \{c_{ij}\} \qquad \text{for all } j \in J$$

and set

$$p_{ij} = t_j - c_{ij} \qquad \text{for all } i \in I \text{ , } j \in J \text{ ;}$$

from (1) we can write :

$$Z = \sum_{i \in I} \sum_{j \in J} (t_j - p_{ij})\, x_{ij} =$$

$$\sum_{j \in J} t_j \sum_{i \in I} x_{ij} - \sum_{i \in I} \sum_{j \in J} p_{ij}\, x_{ij} \text{ ;}$$

because of (3),

$$Z = t - \sum_{i \in I} \sum_{j \in J} p_{ij}\, x_{ij}$$

where $t = \sum_{j \in J} t_j$. (P2) can thus be formulated as the maximization problem.

$$\text{maximize} \quad W = \sum_{i \in I} \sum_{j \in J} p_{ij} x_{ij} \qquad (5)$$

(P3)

$$\text{subject to} : \quad (2), (3), (4)$$

where p_{ij} can be interpreted as the profit given by assigning trip j to vehicle i.

Martello and Toth propose a branch and bound algorithm where, at each node of the decision-tree, an upper bound is obtained by solving the relaxed problem (5), (2), (4); this relaxation would have no meaning in formulation (P2) since the removal of condition (3) would lead to a trivial problem (solution $x_{i,j} = 0$ for all $i \in I$, $j \in J$) with a useless lower bound of value 0; in (P3) we obtain a nontrivial problem whose solution can be obtained by solving K+1 0-1 single Knapsack problems, one for each vehicle $i \in I$, of the form

$$\text{maximize} \quad u_i = \sum_{j \in J} p_{ij} x_{ij}$$

$$\text{subject to} : \quad \sum_{j \in J} r_{ij} x_{ij} \leq b_i$$

$$x_{ij} \in \{0,1\} \quad \text{for all} \quad j \in J;$$

the corresponding upper bound is

$$u = \sum_{i \in I} u_i .$$

Then, they use a branching strategy where K+1 single Knapsack problems for the computation of the upper bound associated with the first node of the branch-decision tree, but only one single Knapsack problem for each of the other nodes according to the infeasibility type (a trip j is not assigned or assigned to more than one vehicle).

However, it is not necessary to apply all the steps of the above algorithm to our problem. Since taking $t_j = \bar{c}_{jq} + 1$ ($t_j = \max_{i\in I, j\in J} \{c_{ij}\} + 1$), (P3) constitutes a special structure. K single Knapsack problems are generated for vehicles 1 to K which are equal to each other and one single Knapsack problem is generated for vehicle 0 where all cost coefficients are equal to 1. Then, we use a simple procedure for this relaxed subproblem. The procedure is as follows:

(i) solve a single Knapsack problem for vehicle 1;

(ii) solve a single Knapsack problem for vehicle i while not considering the assigned trips to vehicles 1,..., i-1 in order to avoid assigning a trip to more than one vehicle: repeat this step for i = 2,...,K;

(iii) check if trip j, j = 1,...,NT_q, is not assigned to vehicle i, i = 1,...,K, then assign that trip to the hired vehicle (vehicle 0);

where the cost of hired vehicle is nearly minimized, satisfying vehicle travel time constraints. In this procedure, K single Knapsack problems are solved for each day of the planning horizon. Anyone of the existing routines for single Knapsack problem may be used to solve these problems.

In this study, a branch search algorithm is used which was proposed by Greenberg and Hegerich (1970).

A branch search algorithm for the knapsack problem:

Our branch search algorithm first finds an obvious integer solution to the following problem;

$$(\hat{P}1) \qquad \text{maximize} \quad \sum_{j=1}^{NT_q} V_j \bar{x}_j$$

$$\text{subject to :} \quad \sum_{j=1}^{NT_q} w_j \bar{x}_j \leq W$$

$$\bar{x}_j \in \{0,1\} \qquad j = 1,\ldots, NT_q$$

where $V_j = c_{ij}$, $\bar{x}_j = x_{ij}$, $w_j = r_{ij}$ and $W = b_i$, $j = 1,..,NT_q$ for a vehicle i. We also assume that indices have been arranged so that $v_1/w_1 \geq v_2/w_2 \geq \ldots \geq v_{NT_q}/w_{NT_q}$. This solution is a lower bound to the optimal solution. We develop a branch of a tree and explore each part of the branch until the lower bound is reached or until a new feasible solution is found that represents a larger lower bound. We then backtrack and develop new branches of the tree with possibly larger lower bounds. Further branching is excluded when the lower bound is reached. The algorithm stops when all new branches are excluded. The lower bound solution is optimal. The only information that is stored is the current lower bound solution and the branch routin

In this tree search procedure, the optimal fractional solution to

$$\text{maximize} \quad \sum_{j=1}^{NT_q} v_j \bar{x}_j$$

$(\hat{P}2)$

$$\text{subject to} : \sum_{j=1}^{NT_q} w_j \bar{x}_j \leq W$$

$$\bar{x}_j \in \{0,1\} \quad j = 1,\ldots, NT_q$$

with L assigned variables is given by

$$\bar{x}_j = 1 \quad \text{if} \quad j < r \quad , \quad j \neq R(K)(K = 1,\ldots,L)$$

$$\bar{x}_j = 0 \quad \text{if} \quad j > r \quad , \quad j \neq R(K)(K = 1,\ldots,L)$$

$$\bar{x}_{R(K)} = 0 \text{ or } 1 \quad (K = 1,..,L) \text{ depending on the assignment}$$

$$\bar{x}_r = (W - \sum_{j=1}^{L} w_{R(j)} \bar{x}_{R(j)} - \sum_{j\in M(L)} w_j)/w_r$$

$$Z(L+1) = \sum_{j=1}^{L} v_{R(j)} \bar{x}_{R(j)} + \sum_{j\in M(L)} v_j + v_r \bar{x}_r$$

where the M(L) is given by

$$M(L) = \{j \mid j < r \ , \ j \neq R(K) \ , \ (K = 1,\ldots,L)\}$$

and r is the least integer $(0 \leq r \leq NT_q)$ for which

$$\sum_{j\in M(L)} w_j + w_r \geq W - \sum_{j=1}^{L} w_{R(j)} \bar{x}_{R(j)} .$$

If no r exists we have all $\bar{x}_j = 1$ for $j \neq R(K)$ $(K = 1,\ldots,L)$. A lower bound to the solution of $(\hat{P}1)$ is given by

$$x(L) = \sum_{j=1}^{L} v_{R(j)} \bar{x}_{R(j)} + \sum_{j \in M(L)} v_j .$$

The algorithm follows:

1) Set L = 1

2) Solve ($\hat{P}2$). If the solution is all integer we have the optimal solution. Stop. If not the solution is $\bar{x}_1 = \bar{x}_2 , \ldots = \bar{x}_{r-1} = 1$ and $\bar{x}_r$ is fractional. We calculate $x_o = \sum_{j<r} v_j$ and store the solution as a lower bound.
Set $R(1) = r$ and $\bar{x}_r = 0$.

3) a. Solve ($\hat{P}2$) with the L assigned variables added as constraints. If $Z(L+1) \leq x_o$ go to 4. If $Z(L+1) > X_o$ and we have an integer solution, take $x_o = Z(L+1)$, form a new solution and go to 4. If $Z(L+1) > X_o$ and we do not have an integer solution, go to 3.b.

b. If $x(L) > X_o$, take $X_o = X(L)$ and form a new solution set. In any case, set L = L+1, take R(L) = r and $\bar{x}_r = 0$. Go to 3.a.

4)a. If $\bar{x}_{R(L)}$ is equal to zero, change it to one and go to 3.a. If it is equal to one go to 4.b.

b. If L = 1, the stored solution is optimal. Stop. If $L \neq 1$ use variable R(L) as an unassigned one, set L = L-1 and go to 4.a.

3.4 Assignment Subproblem

In this step first marginal delivery cost, $CK \times t_{ikq}$, of serving node i by vehicle K on day q is generated for $i = 2,\ldots, N$; $k = 1,\ldots,K$, $q = 1,\ldots,Q$, based on the existing route structure. This is done in the following way:

Let

R_{kq} : set of trips contained on route K, on day q; $k = 1,..,K$; $q = 1,\ldots,Q$.

N_{jq} : set of nodes served by trip j on day q; $j = 1,\ldots, NT_q$; $q = 1,\ldots,Q$.

$CV(N_{jq})$: set of nodes on the boundary of the convex hull of N_{jq}

Then for	Define	
$i \in N$; $i \notin CV(N_{jq})$ $j = 1,\ldots, NT_q$; $q = 1,\ldots,Q$	t_{ikq} = smallest insertion time between node i and $CV(N_{jq})$ over all $j \in R_{kq}$ $k = 1,..,K$; $q = 1,\ldots,Q$.	(1)
$i \in N$; $i \in CV(N_{j_1,q_1})$ for some $j_1 \in R_{k_1,q_1}$	t_{ikq} = smallest insertion time between node i and $CV(N_{jq})$ over all $j \in R_{kq}$ $q = 1,\ldots,Q$; $k = 1,\ldots,K$ excluding $q = q_1$, $k = k_1$.	(2)
$i \in N$; $i \in CV(N_{j_1,q_1})$ for some $j_1 \in R_{k_1,q_1}$	$t_{ik_1q_1}$ = smallest insertion time between node i and $CV(N_{jq})$ over all $j \in R_{k_1q_1}$, getting a new $CV(N_{j_1, q_1})$ by excluding node i.	(3)

The term "insertion time" between node i and $CV(N_{jq})$ refers to the minimum additional travel time necessary to insert node i into the circuit formed by the nodes in $CV(N_{jq})$. The insertion time between a node which is on the boundary of a convex hull and the route whose includes that node has to be adjusted. Because these values will be used in the generation of new sets of nodes for the next iteration and we do not want a lot of changes on the boundary of that convex hull. This can be done by using a multiplier for condition (3). This multiplier will be analyzed in section 4.

The following example illustrates the determination of t_{ikq} for the nodes and routes given in Figure 3.3 (Q assumed to be unity).

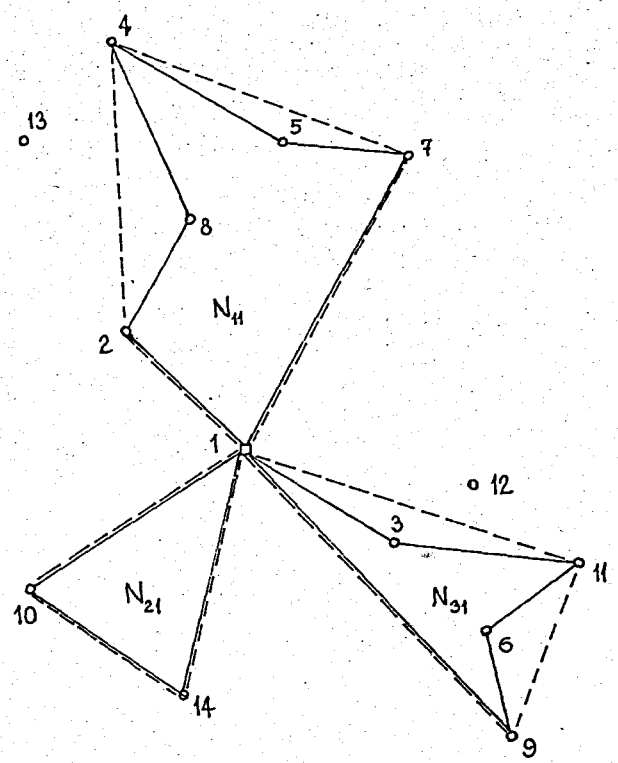


Figure 3.3 Example for the determination of insertion times

In the above example, there are three trips. Let us pack these trips into two routes. Assume trips 1 and 2 are assigned to vehicle 1 and trip 3 is assigned to vehicle 2. Then, the sets can be determined as follows:

$R_{11} = \{1,2\}$ and $R_{21} = \{3\}$

$N_{11} = \{2,8,4,5,7\}$, $N_{21} = \{10,14\}$ and $N_{31} = \{3,11,6,9\}$

$CV(N_{11}) = \{1,2,4,7\}$, $CV(N_{21}) = \{1,10,14\}$ and

$CV(N_{31}) = \{1,11,9\}$.

Using these sets, the determination of t_{ikq}, which will be used for generating marginal delivery costs, is done in the following way :

For condition (1), take demand nodes 5, 13, 12 :

$t_{511} = d_{45} + d_{57} - d_{47}$

$t_{521} = d_{15} + d_{5,11} - d_{1,11}$

$t_{13\ 1\ 1} = d_{2,13} + d_{13,4} - d_{24}$

$t_{13,2,1} = d_{1,13} + d_{13,11} - d_{1,11}$

$t_{12,1,1} = d_{1,12} + d_{12,7} - d_{1,7}$

$t_{12,2,1} = d_{1,12} + d_{12,11} - d_{1,11}$

For condition (2), take demand nodes 4, 11;

$4 \in CV(N_{\underline{1}1})$ where $\underline{1} \in R_{11}$, then exclude $k = 1$

$t_{421} = d_{14} + d_{411} - d_{1,11}$

$11 \in CV(N_{31})$ where $3 \in R_{21}$, then exclude $K = 2$

$t_{11,1,1} = d_{1,11} + d_{11,7} - d_{1,7}$

For condition (3), take demand nodes 4,11;

new $CV(N_{11}) = \{1,2,8,5,7\}$ after excluding node 4,

new $CV(N_{31}) = \{1,3,6,9\}$ after excluding node 11.

Then,

$$t_{411} = (d_{84} + d_{45} - d_{85}) \times m$$

$$t_{11,2,1} = (d_{3,11} + d_{11,6} - d_{36}) \times m$$

where m is a multiplier.

Now it is possible to treat the delivery costs as independent of each other. Then, variables x_{ixq}, x_{ioq} $i = 2,\ldots,N$, $k = 1,\ldots,K$, $q = 1,\ldots,Q$ are defined for the last subproblem in the following way:

$$x_{ikq} = \begin{cases} 1 & \text{if node } i \text{ is serviced by vehicle } k \text{ on day } q \\ 0 & \text{otherwise} \end{cases}$$

$$x_{ioQ} = \begin{cases} 1 & \text{if node } i \text{ is not serviced on or before day } Q \\ 0 & \text{otherwise} \end{cases}$$

and all costs involved in the problem are expressed in terms of these variables as :

$$\text{Total Delivery Costs} : \sum_{k=1}^{K} \sum_{i=2}^{N} \sum_{q=1}^{Q} CK\, t_{ikq}\, x_{ikq}$$

where CK is the operating cost per unit time of each vehicle

t_{ikq} is the incremental time necessary to visit node i on route k in day q, i = 2,...,N ; k = 1,...,K ; q = 1,..,

Total Stockout Costs:

$$\sum_{q=1}^{Q} \sum_{i=2}^{N} [SL_i\, p\{ \sum_{r=t_i}^{q-1} \xi_r^i > SC_i \mid \sum_{r=t_i}^{0} \xi_r^i \leq SC_i \}] [\sum_{k=1}^{K} x_{ik}]$$

$$+ \sum_{i=2}^{N} [SL_i\, p \{ \sum_{r=t_i}^{0} \xi_r^i > SC_i \mid \sum_{r=t_i}^{0} \xi_r^i \leq SC_i \}] [x_{io}\, Q]$$

where SL_i is the cost of a stockout at node i on any given day.

SC_i is the storage capacity of node i;

$p \{ \sum_{r=t_i}^{q-1} \xi_r^i \; SC_i \mid \sum_{r=t_i}^{0} \xi_r^i \leq SC_i \}$ is the probability of a stockout during the planning horizon given a delivery on day q, which is defined as P_{iq} in 3.1.

At this stage, a penalty cost for exceeding the vehicle travel time constraint is taken into consideration. Because vehicle travel time constraints may be violated, if the assignment problem is based on stockout and delivery costs only. Also, an increase in the cost of additional hired vehicle

can be incurred. We went to discourage or encourage the assignment of demand nodes to a vehicle k in day q according to its total travel time. In our problem, vehicle travel time constraints are same for all vehicles. Then, the total travel time for a vehicle k in day q, which is generated in the routing subproblem, may be used to reflect the penalty cost associated with (timewise) overloading vehicle k in day q. Also, a multiplier, which will be analyzed in Section 4, is necessary for these penalty cost.

Total Penalty Costs :

$$\sum_{k=1}^{K} \sum_{q=1}^{Q} n\, v_{kq} \left(\sum_{i=2}^{N} t_{ikq}\, x_{ikq} - T \right)$$

where v_{kq} is the penalty costs of exceeding the time constraint on vehicle k in day q;

n is a multiplier;

T is maximum length of any route on any day.

Finally, based on the above costs, the assignment subproblem, problem P4 below, is constructed. This is the subproblem that determines whether a node will be serviced during the planning horizon and if so, on what day and by which vehicle:

$$\min \sum_{k=1}^{K} \sum_{i=2}^{N} \sum_{q=1}^{Q} c_k \cdot t_{ikq} \cdot x_{ikq} + \sum_{k=1}^{K} \sum_{q=1}^{Q} n \cdot v_{kq} \left(\sum_{i=2}^{N} t_{ikq}\, x_{ikq} - T \right)$$

$$(P4) \qquad + \sum_{i=2}^{N} SL_i \left| \sum_{q=1}^{Q} p_{iq} \sum_{k=1}^{K} x_{ikq} + P_{i,Q+1}\, x_{ioq} \right|$$

$$\text{subject to :} \quad \sum_{k=1}^{K} \sum_{q=1}^{Q} x_{ikq} + x_{ioq} = 1 \qquad i = 2, \ldots, N$$

$$x_{ikq} = 0,1 \qquad i = 2,\ldots,N \;;\quad k = 1,\ldots,K \;;\; q = 1,..,Q$$

$$x_{ioq} = 0,1 \qquad i = 2,\ldots,N.$$

The above problem is equivalent to :

$$\min \sum_{i=2}^{N} \Big[\sum_{k=1}^{K} \sum_{q=1}^{Q} \big| (CK + nv_{kq})\, t_{ikq} + SL_i\, p_{iq} \big] x_{ikq} + SL_i\, p_{i,Q+1}\, x_{ioQ} \big|$$

$(\bar{P}4)$

$$\text{subject to :} \quad \sum_{k=1}^{K} \sum_{q=1}^{Q} x_{ikq} + k_{ioQ} = 1 \qquad i = 2,..,N$$

$$x_{ikq} = 0,1 \qquad i = 2,\ldots,N \;;\quad k = 1,\ldots,K \;;\; q = 1,..,Q$$

$$x_{ioQ} = 0,1 \qquad i = 2,\ldots,N$$

Defining

$$\bar{c}_{ikq} \equiv (CK + n\, v_{kq})\, t_{ikq} + SL_i\, p_{iq}$$

$$\bar{c}_{ioQ} \equiv SL_i\, p_{i,Q+1}$$

Problem $(\bar{P}4)$ becomes :

$$\min \sum_{i=2}^{N} \Big[\sum_{k=1}^{K} \sum_{q=1}^{Q} \bar{c}_{ikq}\, q_{ikq} + \bar{c}_{ioQ}\, x_{ioQ} \Big]$$

$(\bar{\bar{P}}4)$

$$\text{subject to :} \quad \sum_{k=1}^{K} \sum_{q=1}^{Q} x_{ikq} + x_{ioQ} = 1 \qquad i = 2,\ldots,N$$

$$x_{ikq} = 0,1 \qquad i = 2,\ldots,N \;;\; k = 1,\ldots,K \;;\; q = 1,..,Q$$

$$x_{ioQ} = 0,1 \qquad i = 2,\ldots,N$$

Furthermore, it is possible to decompose the above problem into N-1 trivial problems as:

For $i = 2,\ldots,N$ solve

$$\min \sum_{k=1}^{K} \sum_{q=1}^{Q} \bar{c}_{ikq} x_{ikq} + \bar{c}_{ioQ} x_{ioQ}$$

$$(\hat{P}4) \qquad \text{subject to} : \sum_{k=1}^{K} \sum_{q=1}^{Q} x_{ikq} + x_{ioQ} = 1$$

$$x_{ikq} = 0,1 \qquad k = 1,\ldots,K \; ; \; q = 1,\ldots,Q$$
$$x_{ioQ} = 0,1$$

In which case determination of an optimal solution becomes very simple:

(i) For each $i=2,\ldots,N$, determine k^*, q^* such that

$$\bar{c}_{ik^*q^*} = \min (\bar{c}_{ikq}) \quad \text{over} \quad k=0,\ldots,K \quad q = 1,\ldots,Q$$

(ii) set $x_{ik^*q^*} = 1$

$$k_{ikq} = 0 \quad k = 0,\ldots,K \; ; \; q = 1,\ldots,Q \; (k,q) \neq (k^*,q^*)$$

Note that parameters t_{ikq} used in the above problem are only approximations to the actual marginal delivery costs. Real marginal delivery costs are dependent on the current set of trips and not the previous set. However, basing the approximations on the convex hull of the previous trips, rather than the trips themselves is hoped to be a compensating factor; while the actual trips may change considerably from one iteration to the next, their general structures and orientations should not change as much, and convex hull is usually a good proxy for general structure and orientation.

3.5 The Algorithm

Initial step : For every $i = 2,\ldots,N$, $q = 1,\ldots,Q$ 1 generate estimations for e_i , P_{iq}. Initialize the sets V_q , $q = 1,\ldots,Q$ in the following manner: Starting with $q = 1$ and incrementing up to $q = Q$, execute the following

for every $i = 2,\ldots,N$, not already assigned to a set, assign i to V_q if

$$SL_i \; P_{iq} > \bar{s}$$

where $\bar{s}$ is a predetermined constant.

Step 1 : Using a modified Clarke-Wright algorithm, generate a set of minimum cost trips on the set V_q, while satisfying the vehicle capacity constraints, for every $q = 1,\ldots,Q$.

Step 2 : Using the generalized assignment formulation, pack the generate trips into K vehicle routes while satisfying travel time constraints and minimizing the use of hired vehicles, for every $q = 1,\ldots,Q$.
If the solution obtained is satisfactory, stop; otherwise go to step 3.

Step 3 : Update marginal operating costs using the routes generated in Step 2 and penalty costs found by vehicle travel time. Update sets V_q , $q = 1,..,Q$ by using the decision rules developed for problem ($\hat{P}4$).
If there is no change in V_q , $q = 1,..,Q$, stop; otherwise go to step 1.

IV. A NUMERICAL EXAMPLE AND COMPUTATIONAL RESULTS

The heuristic algorithm of Section 3 was implemented in FORTRAN IV. The program consists of a fortran deck of about 1000 cards, including 7 subroutines. The size of the computer needed depends on the number of nodes being considered. Present dimensions allow for as many as 100 nodes and 10 days. (Appendix I gives the list of computer program, Appendix II gives the input data file.)

A numerical example, which is generated randomly, is solved through the algorithm presented in this study by using the above computer program. In this computational experiment, the multipliers m and n, which are defined in section 3.4, are analyzed with this numerical example. It is as follows:

Let N = 50, K = 2, VC = 265, CK = 10, HK = 0,25, $R = E[\xi_q] = 20$ and $\sigma^2 = VAR[\xi_q] = 3$ (suppose the daily requirements of each demand node in day q have the same probability distribution). Let the planning horizon be 3

3 delivery periods (3 days). Table 4.1 gives the x and y coordinates, storage capacity, stockout cost and the date of last delivery for each node.

The computer program first generates the sets of seed noeds ($\bar{s}$ = 500, one set for each day). We use normal distribution table, Appendix III gives the available stockout probabilities.

$$V_1 = \{1,2,3,4,5,8,11,12,13,15,16,20,22,24,25,27,28,32,33,39,40,41,42,43,45,48\}$$

$$V_2 = \{1,6,7,9,10,14,18,19,26,37,44,47,49,50\}$$

$$V_3 = \{1,17,23,30,31,35,38,46\}$$

Notice that demand nodes 21,29,34 and 36 are not assigned to any day.

Then, routing subproblem generates minimum cost trips for each set of nodes while satisfying vehicle capacity constraints. Table 4.2 gives the number of trips generated for each day, the total distance (instead of total duration) and the expected total amount delivered for each trip. Also Figure 4.1, 4.2 and 4.3 show the positions of nodes and the network , which are formed by the trips generated in this subproblem, for the first, second, and the third day respectively.

Now, the NT_q trips generated in the routing subproblem are packed into two routes satisfying vehicle travel time constraints. We express the vehicle travel time constraints

Table 4.1

		X-COORD.	Y-COORD.	STORAGE CAPACITY	LAST DELIVERY	ST-OUT COST
SUPPLY POINT :	1 *	85.	90.	9999.9	0.	0.
DEMAND POINTS :	2 *	2.	0.	18.	1.	1000.
	3 *	1.	8.	38.	2.	1050.
	4 *	7.	4.	56.	2.	900.
	5 *	0.	15.	45.	1.	1000.
	6 *	15.	15.	34.	3.	700.
	7 *	20.	4.	29.	3.	870.
	8 *	28.	34.	45.	2.	950.
	9 *	17.	40.	50.	1.	560.
	10 *	57.	0.	39.	3.	780.
	11 *	34.	68.	45.	1.	900.
	12 *	89.	34.	45.	3.	1100.
	13 *	90.	100.	67.	2.	1200.
	14 *	85.	86.	78.	2.	590.
	15 *	75.	39.	67.	3.	830.
	16 *	23.	89.	69.	2.	1000.
	17 *	101.	57.	89.	1.	1250.
	18 *	155.	120.	89.	2.	1160.
	19 *	123.	153.	86.	2.	950.
	20 *	167.	187.	39.	2.	1090.
	21 *	182.	198.	89.	1.	880.
	22 *	104.	187.	45.	1.	990.
	23 *	108.	119.	65.	1.	560.
	24 *	114.	134.	44.	2.	700.
	25 *	128.	138.	47.	2.	900.
	26 *	138.	183.	63.	1.	980.
	27 *	105.	172.	41.	2.	1000.
	28 *	60.	140.	28.	1.	850.
	29 *	173.	172.	71.	2.	500.
	30 *	104.	187.	84.	1.	1500.
	31 *	147.	65.	97.	2.	1300.
	32 *	56.	196.	38.	2.	600.
	33 *	76.	187.	87.	3.	900.
	34 *	195.	95.	96.	1.	1080.
	35 *	160.	20.	67.	1.	770.
	36 *	189.	1.	86.	1.	890.
	37 *	176.	43.	87.	2.	1300.
	38 *	18.	145.	67.	1.	900.
	39 *	29.	178.	40.	2.	780.
	40 *	65.	156.	45.	3.	560.
	41 *	89.	132.	31.	1.	1000.
	42 *	100.	99.	33.	1.	600.
	43 *	98.	109.	63.	2.	750.
	44 *	111.	11.	56.	1.	1200.
	45 *	122.	67.	63.	2.	970.
	46 *	47.	198.	78.	1.	900.
	47 *	109.	187.	87.	2.	1300.
	48 *	130.	50.	53.	2.	540.
	49 *	38.	109.	80.	2.	2000.
	50 *	89.	65.	67.	1.	1800.

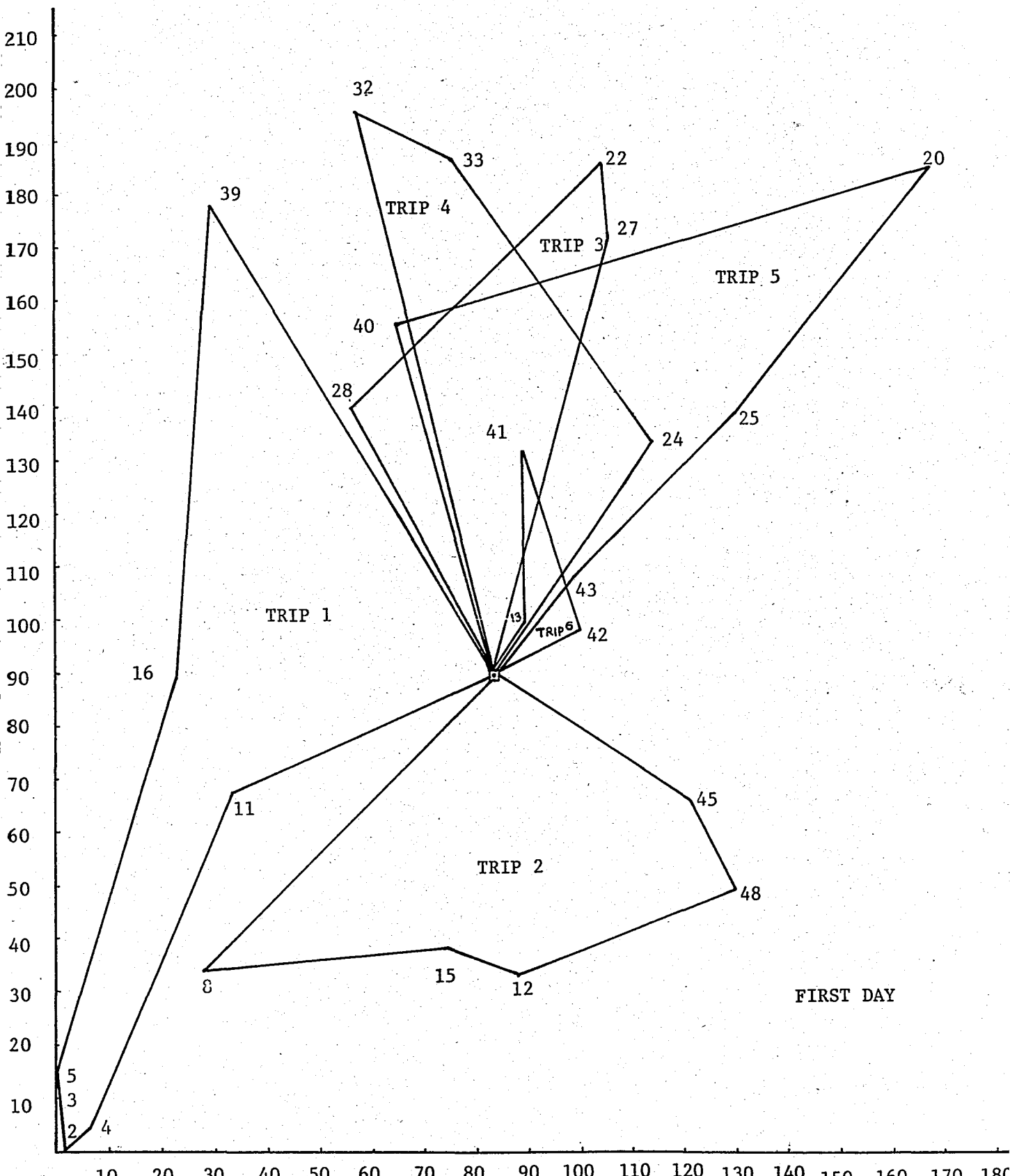

TRIP 1
TRIP 2
TRIP 3
TRIP 4
TRIP 5
TRIP 6
FIRST DAY
32
33
22
20
39
27
40
28
41
24
25
43
42
16
11
45
48
8
15
12
5
3
2
4
10 20 30 40 50 60 70 80 90 100 110 120 130 140 150 160 170 180
10 20 30 40 50 60 70 80 90 100 110 120 130 140 150 160 170 180 190 200 210

Figure 4.1

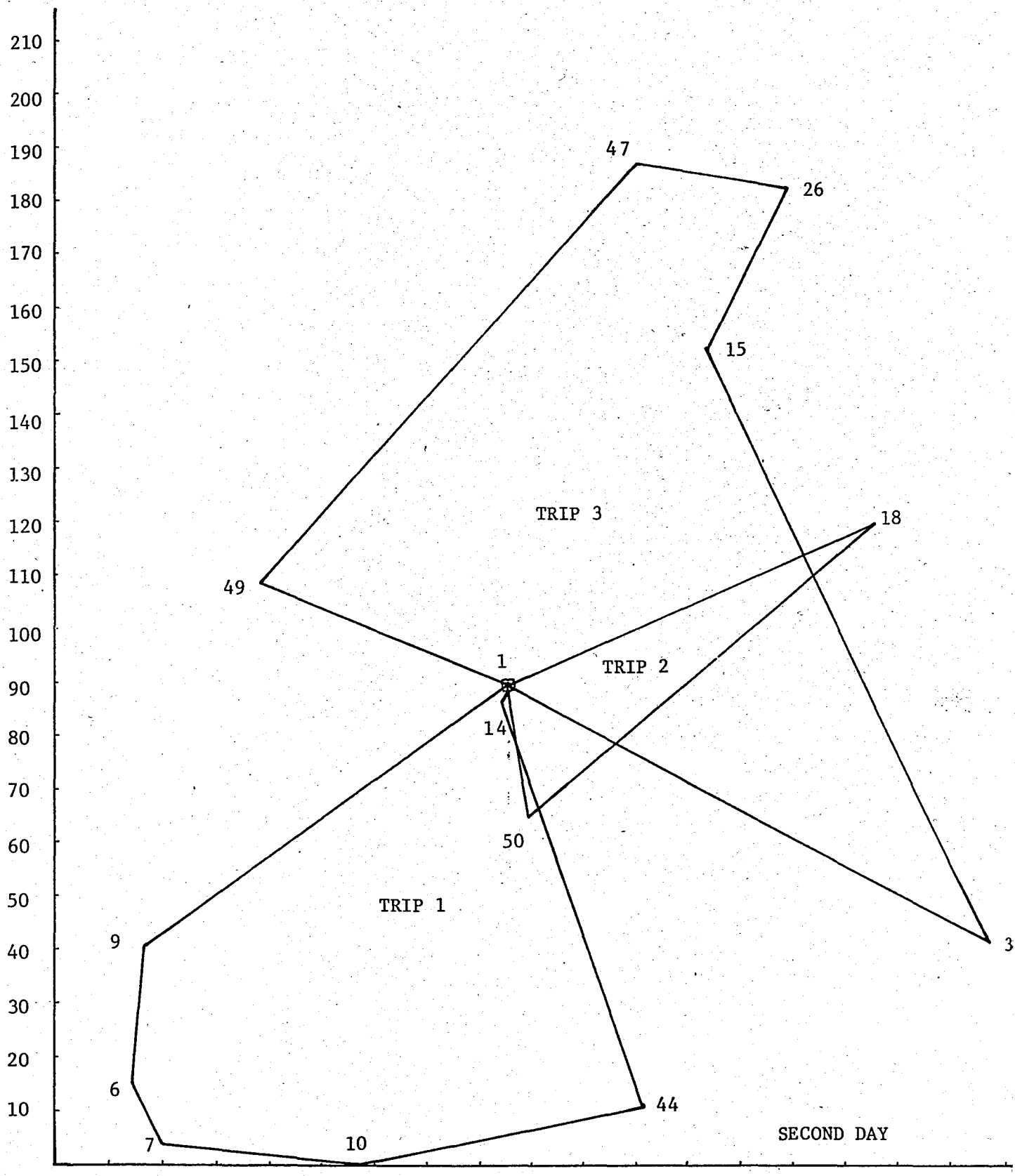


Figure 4.2

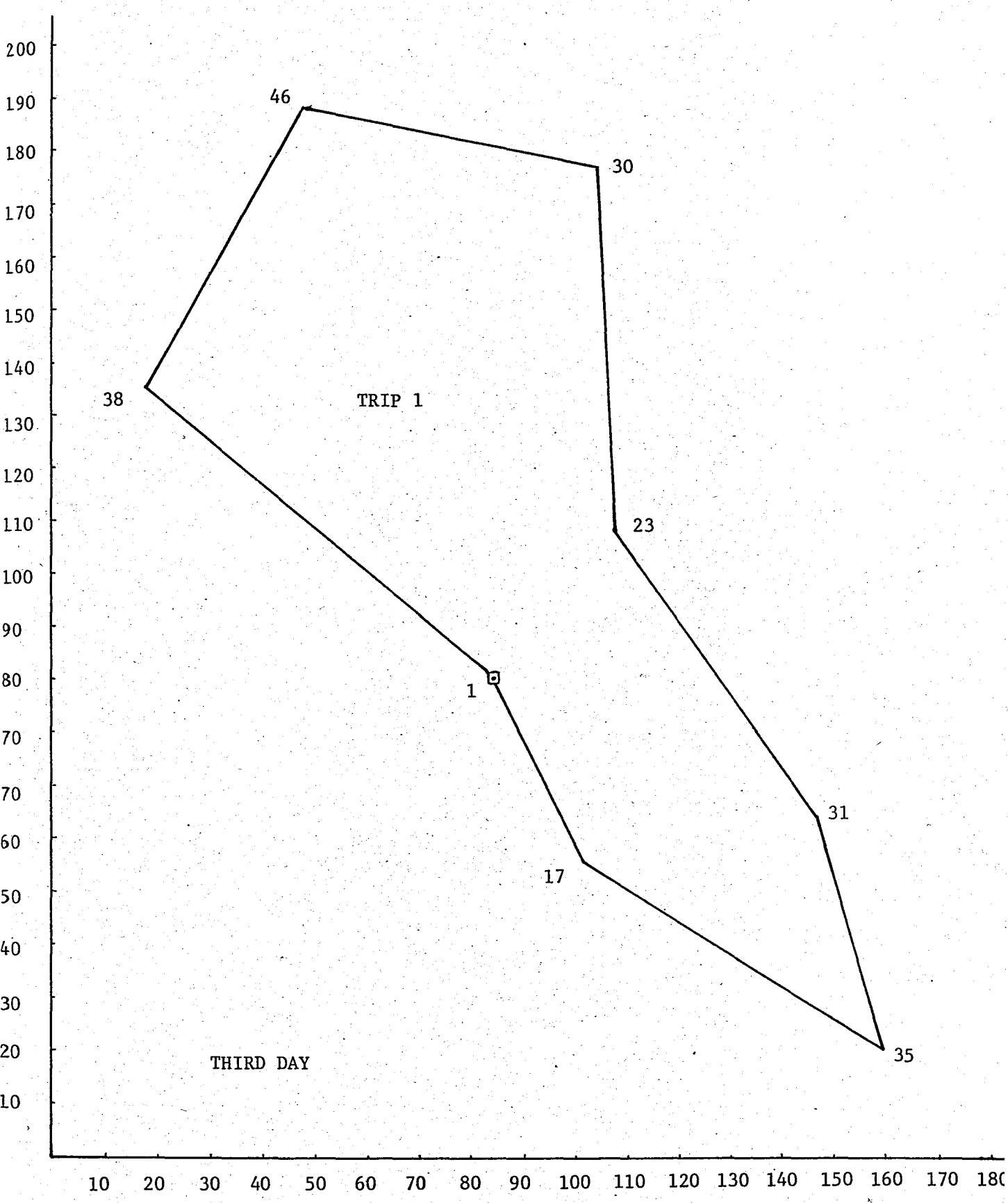

200
190
180
170
160
150
140
130
120
110
100
90
80
70
70
60
50
40
30
20
10
46
30
38
TRIP 1
23
1
31
17
35
THIRD DAY
10 20 30 40 50 60 70 80 90 100 110 120 130 140 150 160 170 180

Figure 4.3

in terms of the unit of length, which is also used in the total distance values.

Table 4.2

$q = 1,2,3 \qquad j = 1,\ldots,NT_q$

q	NT_q	j	TD_{jq}	TA_{jq}
1	6	1	417.54	263
		2	248.40	228
		3	219.72	225
		4	249.74	253
		5	302.94	245
		6	95.47	160
2	3	1	297.27	236
		2	187.39	107
		3	443.51	246
3	1	1	493.03	265

Let T = 700. Then, Table 4.3 gives the results of the packing subproblem.

Table 4.3

	Trip number	
q	Vehicle 1	Vehicle 2
1	1,4	2,5,6
2	2,3	1
3	1	-

Trip 3 uses additional hired vehicle in day 1.

At this stage, the costs which are defined in section 2.1 can be calculated. These costs are as follows:

Total Delivery Cost	:	29550.3
Total Stockout Cost	:	40301.7
Cost of Hired Vehicle:		12359.3
Total Cost	:	82211.3

Note that, 45 nodes are served in the first iteration. These costs do not change for different values of m,n. Because m and n do not have an effect until this stage.

Then, assignment subproblem generates new sets of nodes which will be used in the second iteration. The multipliers m, and n are used in this subproblem. This means, the related costs change according to the values of m and n. Table 4.4 gives the changes in the cost values for some values of n taking m = 1. For small values of n, the use of hired vehicle is encouraged. For example, when n is equal to 0.005, the cost of additional hired vehicles is present at each iteration. As we increase the value of n, hired vehicle usage ends. Therefore, the cost of additional hired vehicle can be nearly minimized by adjusting the value of n.

Table 4.5 gives the changes in the cost values for some values of m using two different values of n. It is seen that increasing the value of m does not change the results very much. Also, some total cost values are same for different values of m. Then, taking m = 1 is satisfactory in most cases for our purpose. But, it it is desired, total cost may be

Iteration	m	n	No of served nodes				Delivery Cost	Stockout Cost	Cost of Hired Vehicle	Total Cost
			1	2	3	Σ				
2	1	.005	25	6	-	31	20452.0	24074.0	16990.9	61516.9
3			25	6	-	31	19677.2	23236.3	22120.3	65033.8
4			24	6	-	30	22348.1	23498.4	25504.2	71350.7
5			25	5	-	30	18259.8	24116.0	8522.2	50898.0
2	1	.1	16	5	1	22	14511.9	30198.7	-	44710.6
3			12	3	-	15	7712.3	33656.7	-	41369.0
4			18	-	-	18	13631.0	30658.3	-	44289.3
5			13	-	-	13	7155.7	34210.2	-	41365.9
2	1	.25	10	7	1	18	12076.9	33822.9	-	45899.8
3			12	4	-	16	8619.4	32414.4	-	41033.8
4			13	2	-	15	11148.4	32114.2	-	43262.6
5			9	3	-	12	8116.7	35611.9	-	43728.6
2	1	.5	10	7	1	18	12076.9	33822.9	-	45899.8
3			11	3	-	14	10205.0	32154.5	-	42359.5
4			8	3	-	11	7631.1	36143.6	-	43774.7
5			14	1	-	15	11410.9	32114.2	-	43525.1
2	1	1	7	7	1	15	10682	36870.2	-	47552.2
3			12	1	1	14	11809.6	33109.7	-	44919.3
4			4	4	1	9	7400.2	38043.4	-	45443.6
5			13	1	1	15	13125.3	33069.4	-	46194.7

Table 4.4

Iteration	m	n	No of served Nodes				Delivery Cost	Stockout Cost	Cost of Hired Vehicle	Total Cost
			1	2	3	Σ				
2	1.5	.1	16	5	1	22	14511.9	30198.7	-	44710.6
3			12	3	-	15	7712.3	33656.7	-	41369.0
4			18	-	-	18	13631.0	30658.3	-	44289.3
5			12	-	-	12	7669.0	35160.2	-	42829.2
2	2	.1	15	5	1	21	13472.4	30460.8	-	43933.2
3			10	5	-	15	9948.1	35353.8	-	45301.9
4			15	2	-	17	11209.4	31550.4	-	42759.8
5			10	3	-	13	7464.6	34495.6	-	41960.2
2	3	.1	15	5	1	21	13472.4	30460.8	-	43933.2
3			10	5	-	15	9948.1	35353.8	-	45301.9
4			16	1	-	17	13212.2	31550.4	-	44762.6
5			12	1	-	13	7719.5	33461.8	-	41181.3
2	1.5	.25	10	7	1	18	12076.9	33822.9	-	45899.8
3			13	3	-	16	10449.9	31646.1	-	42096.0
4			8	3	-	11	7697.3	36561.9	-	44259.2
5			13	2	-	15	9464.5	32726.3	-	42190.8
2	2	.25	10	7	1	18	12076.9	33822.9	-	45899.8
3			13	3	-	16	10449.9	31646.1	-	42096.0
4			8	3	-	11	7697.3	36561.9	-	44259.2
5			13	2	-	15	9464.5	32726.3	-	42190.8

Table 4.5

minimized by adjusting the value of m after deciding a value for n.

Additional runs are done in order to see the effects of the multipliers when some cost coefficients are changed. We take m = 1 and multiply all the stockout cost coefficients by the same constant. Table 4.6 gives the results of this experiment. If we multiply the coefficients by two, n = 0.1 is satisfactory for our purpose. But, as we continue to increase these coefficients, we need to increase the value of n in order to discourage the use of additional hired vehicle. Therefore, by adjusting the values of m and n, a near optimal solution for the inventory routing problems can be obtained using the proposed heuristic solution procedure.

m = 1

		No of served nodes							
Iteration	SO	1	2	3	Σ	Delivery Cost	Stockout Cost	Cost of Hired Vehicle	Total Cost
1		25	13	7	45	29550.3	80603.3	12359.3	122512.9
2		19	6	-	25	16057.9	55024.7	-	71082.6
3	n=.1	14	8	-	22	14891.0	57955.1	-	72846.1
4		17	5	-	22	16115.8	56846.3	6380.0	79342.1
5	2	17	8	-	25	18024.3	53312.1	-	71336.4
1		25	13	7	45	80603.3	80603.3	12359.3	122512.9
2	n=.05	23	6	-	29	50879.5	50879.5	22072.6	94405.1
3		20	5	-	25	52037.6	52037.6	1495.5	72525.0
4	2	20	8	-	28	49741.6	49741.6	10766.7	81274.9
5		18	6	-	24	56828.3	56828.3	-	73309.1
1		25	13	7	45	201508.3	201508.3	12359.3	243417.9
2	n=.1	24	5	1	36	122101.6	122101.6	16619.5	162768.0
3		17	8	-	25	130222.7	130222.7	-	149323.4
4	5	17	6	-	23	132343.9	132343.9	-	148728.0
5		17	8	-	25	130222.7	130222.7	-	149323.4
1		25	13	7	45	201508.3	201508.3	12359.3	243417.9
2	n=.25	19	6	2	27	133281.8	133281.8	-	154131.9
3		14	11	-	25	139869.0	139869.0	-	157188.5
4	5	18	5	2	25	133335.9	133335.9	-	153578.2
5		17	9	-	26	131416.9	131416.9	-	149768.8
1		25	13	7	45	322413.3	322413.3	12359.3	364322.9
2		19	6	2	27	213250.9	213250.9	-	234100.5
3	n=.35	14	11	-	25	223790.4	223790.4	-	240929.9
4		18	5	2	25	213337.4	213337.4	-	233579.7
5	8	17	9	-	26	210267.0	210267.0	-	228618.9

Table 4.6

V. CONCLUSIONS

The procedure as a whole is a simple iterative procedure which should produce a near optimal solution. In step 1, the sets of seed nodes are being generated. This generation is based on stockout considerations only. These seeds are used only to obtain some initial estimates for marginal delivery costs and do not lock the final solution. In step 2, the sets of minimum time trips, satisfying vehicle capacity constraints, are being generated through the savings algorithm. These trips should make a reasonably food basis ready for the computation of marginal delivery costs. In step 3, trips are packed into vehicle routes, satisfying vehicle travel time constraints. For this step, the procedure, which is defined in 3.3, is used. The main problem is to decide which nodes to visit and which nodes to skip. This problem is handled in step 4 and a decision rule, based on a tradeoff between "stockout costs" and "operating costs", is obtained. At this stage, good approximations for delivery costs, $c_k t_{ikq}$ are available. In addition, stockout probabilities, P_{iq}, which

are used to estimate expected stockout costs, are known. As a result, the suggested tradeoff between "stockout costs" and "operating costs" is significant.

From the computational results, we see that an improvement at each iteration is not guaranteed. Also, the convergence of the procedure can not be promised. However, it may be possible to handle these handicaps. In order to guarantee improvement at each iteration, the best solution so far may be stored and compared with the new solution. To guarantee convergence the number of iterations may be limited. Another modification for the convergence may be terminating the procedure whenever there is no improvement from one iteration to the next. Additional consideration should also be given to the multipliers discussed in section 3.4.

REFERENCES

1. Assad,A., B.Golden, R.Dahl, and M.Dror, "Design of An Inventory-Routing System for A Large Propane Distribution Firm", working Paper, #82-016, University of Maryland at College Park, (1982).

2. Bodin,L., B.Golden, A.Assad, and M.Ball, "The State of the Art in the Routing and Scheduling of Vehicles and Crews", Working Paper, #81-035, University of Maryland at College Park, (1982).

3. Clarke,G., and J.Wright, "Scheduling of Vehicle from a Central Depot to a Number of Delivery Points", Operations Research, 12, 568-581, (1964).

4. Cook,T., and R.Russell, "A Simulation and Statistical Analysis of Stochastic Vehicle Routing with Timing Constraints", Decision Sciences, 9, 673-687, (1978).

5. Federgruen,A., and P.Zipkin, "A Combined Vehicle Routing and Inventory Allocation Problem", Working Paper #345A, Columbia University, (1982).

6. Golden,B., T.Magnanti and H.Nguyen, "Implementing Vehicle Routing Algorithms", Networks, 7, 113-148, (1977).

7. Greenberg,H., and R.L.Hegerich, "A Branch Search Algorithm for the Knapsack Problem", Management Science, 16, 327-332, (1970).

8. Martello,S., and P.Toth, "An Algorithm for the Generalized Assignment Problem", Computing, 21, 590-603, (1978).

9. Or,I., "A Heuristic Solution Procedure for the Inventory Routing Problem", Research Papers FBE-EM 83/15, Boğaziçi University, (1983).

10. Steward,W., and B.Golden, "Stochastic Vehicle Routing: A Comprehensive Approach", Working Paper #81-010, University of Maryland at College Park, (1982).

APPENDIX I

```
      PROGRAM TAPE(DATA,RESULT,RESULT1,TABLE,TAPE5=DATA,TAPE6=RESULT,
     $TAPE7=RESULT1,TAPE9=TABLE)
C*************************************************************
C*                  A HEURISTIC SOLUTION PROCEDURE           *
C*                            FOR                            *
C*                  THE INVENTORY ROUTING PROBLEM            *
C*************************************************************
      REAL X(150),Y(100),TLONG(100),X1(100),Y1(100)
      INTEGER TRA(5),TRCAP(5),Q(100),TRU(5),ILSE(2000),V(10,100)
      INTEGER PTOUR(100,4),ID(8000,3),ITRIP(100,10),INODE(100)
      INTEGER ITRIP1(10,100,10),NND(10),NNOD(10,100),NKK(10),ITRU(100
      DIMENSION TD1(10,100),TA1(10,100),D(100,100),T(100,6,10)
      DIMENSION NZ(350),IASS(100),IHIRE(10,20),IYCK(50),IRN(10,5)
      DIMENSION B1(6),XS1(50),IROT(10,6,20),IHIRE1(10),IHLLL1(10,100,
      INTEGER P(10,50),K(10,50),IITRIP(100,10),NCON(10),NCON1(10,100
      DIMENSION SC(100),PR(100,10),VD(5,10),TLD(100),SL(100),IHIRE2(10
      DIMENSION IHOLL(100,10),NKS(100),NQS(100),NND1(10),V1(10,100)
      NVT=1
      PAK=1.
      READ(5,1)NPT1,NDAY,TRA(1),TRCAP(1),MXCROP,WIDTH,HEIGHT,CK,HK,ORT
     $SOV
    1 FORMAT(//////,40X,13,//,40X,12,//,40X,13,//,40X,13,//,40X,13,//
     $40X,F5.0,//,40X,F5.0,//,40X,F5.0,//,40X,F7.2,//,40X,F5.0,//,40X
     $F5.0,//////)
      DO 10 I=1,NPT1
      READ(5,2)X1(I),Y1(I),Q(I),TLD(I),SL(I)
   10 CONTINUE
    2 FORMAT(28X,F6.1,5X,F6.1,5X,I4,8X,F4.0,5X,F5.0)
      READ(5,*)DEN1,DEN2,ITER1,SBAR,TKIS,DEN3
      WRITE(6,*)DEN1,DEN2,SBAR,TKIS,DEN3
      DO 7777 IU=1,NPT1
 7777 SL(IU)=DEN3*SL(IU)
      K=1
      DO 1020 JJ=1,35
      READ(9,1021)(NZ(JJ),JJ=K,K+9)
      K=K+10
 1021 FORMAT(10I4)
 1020 CONTINUE
C*************************************************************
C*                  GENERATION OF SEED NODES                 *
C*************************************************************
      DO 1024 I=1,NDAY
      NND(I)=1
 1024 V(I,NND(1))=1
      SIGMA=SOV/ORT
      DO 1023 J=2,NPT1
      SC(J)=Q(J)
      ASSC=SC(J)/ORT
      PER1=TLD(J)
      DO 1022 I=1,NDAY+1
      CALL BULERV(NZ,ASSC,PER1,SIGMA,ERV)
      PER2=I
      SC1=ASSC
      CALL BULPR2(NZ,SC1,ERV,PER2,SIGMA,PR2)
      PR(J,I)=PR2
      IF(IASS(J).EQ.1)GO TO 1022
      IF(I.EQ.NDAY+1.AND.IASS(J).EQ.0)NQS(J)=NDAY+1
      IF((PR(J,I)*SL(J)).LE.SBAR)GO TO 1022
      NQS(J)=I
      NND(I)=NND(I)+1
      V(I,NND(I))=J
      IASS(J)=1
 1022 CONTINUE
 1023 CONTINUE
      DO 1052 I=1,NDAY
      WRITE(6,3)I,NND(I)
```

```
    3 FORMAT(5X,'***',I2,'% DAY***',//,33X,'# OF NODES =',I3)
 1002 WRITE(6,*)(V(I,J),J=1,NND(I))
      NV=TRA(1)
      DO 4000 ITER=1,ITER1
      IF(ITER.EQ.1)GO TO 4001
      DO 4002 J1B=1,NDAY
      NND(J1B)=NND1(J1B)
      DO 4002 J2B=1,NND(J1B)
 4002 V(J1B,J2B)=V1(J1B,J2B)
 4001 SCOST=0.
      DCOST=0.
      HCOST=0.
      DO 4020 N1=2,NPT1
 4020 SCOST=SCOST+SL(N1)*PR(N),NOS(N1))
C*************************************************************
C*                    VEHICLE ROUTING PROBLEM                *
C*************************************************************
      DO 1000 I1A=1,NDAY
      IF(NND(I1A).EQ.1)GO TO 1000
      DO 1001 J=1,NND(I1A)
      X(J)=X1(V(I1A,J))
 1001 Y(J)=Y1(V(I1A,J))
      NPT=NND(I1A)
      ANPT=NPT
      DIV=1
      CALL CONVEP(X,Y,NPT,NVT,TRA,TRCAP,PAR,PXCHGP,C,DIV,HEIGHT,
     $WIDTH,ID,PTOUR,TDIST,ISUM,NARC,TRU)
C     WRITE(6,200)TDIST
C     WRITE(6,300)TRU
C     DO 30 J=1,NARC
C     IF(ID(J,3).EQ.0)GO TO 30
C     WRITE(6,400)(ID(J,K),K=1,3),PTOUR(ID(J,2),2),PTOUR(ID(J,2)
C    $,1),PTOUR(ID(J,2),3),PTOUR(ID(J,2),4)
C  30 CONTINUE
C 200 FORMAT(' THE TOTAL DISTANCE TRAVELED IS ',F10.2//)
C 300 FORMAT('  NUMBER OF VEHICLES OF EACH TYPE',/5I6//)
C 400 FORMAT('   ',3I5,3X,4I5)
C     WRITE(6,*)NARC
C     DO 5000 IH=1,NARC
C5000 WRITE(6,*)(ID(IH,IF),IF=1,3)
      K=1
      DO 334 I=1,NARC
      IF(ID(I,3).EQ.0)GO TO 334
      IF(IUSE(I).EQ.1)GO TO 334
      NN=PTOUR(ID(I,2),4)+1
      ITRIP(K,1)=ID(I,1)
C     WRITE(6,*)ITRIP(K,1)
      ITRIP(K,2)=ID(I,2)
C     WRITE(6,*)ITRIP(K,2)
      IUSE(I)=1
      IF(NN.LT.3)GO TO 339
      DO 335 N=3,NN
      DO 336 J=1,NARC
      IF(ID(J,3).EQ.0)GO TO 336
      IF(IUSE(J).EQ.1)GO TO 336
      IF(ID(J,1).EQ.ITRIP(K,N-1))GO TO 331
      IF(ID(J,2).NE.ITRIP(K,N-1))GO TO 336
      ID(J,2)=ID(J,1)
  331 IF(ITRIP(K,N).NE.0)GO TO 335
      ITRIP(K,N)=ID(J,2)
C     WRITE(6,*)ITRIP(K,N)
      IUSE(J)=1
      IF(ID(J,3).EQ.1)GO TO 335
      GO TO 338
  336 CONTINUE
```

```
  338 NN=NN+1
      ITRIP(K,N+1)=ITRIP(K,N-1)
C     TETA=333333
C     WRITE(6,*)ITRIP(K,N+1),TETA
  335 CONTINUE
  339 ITRIP(K,NN+1)=ITRIP(K,1)
      DO 337 NNN=1,NARC
      IF(ID(NNN,3).EQ.0)GO TO 337
      IF(IUSE(NNN).EQ.1)GO TO 337
      IF(ID(NNN,2).EQ.ITRIP(K,NN))IUSE(NNN)=1
  337 CONTINUE
      INODE(K)=NN+1
      K=K+1
  334 CONTINUE
      KK=K-1
      NT=KK
C     WRITE(6,*)((ITRIP(L1,L2),L2=1,INODE(L1)),L1=1,NT)
C*****************************************************************
C*                        CONVEX HULL                            *
C*****************************************************************
      CALL CHULL(NT,X,Y,ITRIP,INODE,NCON,IHULL)
      DO 79 I=1,NT
      NCON1(III,I)=NCON(I)
      IF(IHULL(I,1).EQ.1)GO TO 83
      JS=0
      DO 84 JV=2,NCON(I)
      IF(IHULL(I,JV).NE.1)GO TO 84
      JS=1
      JV1=JV
   84 CONTINUE
      IF(JS.NE.0)GO TO 85
      NCON(I)=NCON(I)+1
      IITRIP(I,1)=1
      DO 70 JE=2,NCON(I)
   70 IITRIP(I,JE)=IHULL(I,JE-1)
      GO TO 79
   83 DO 71 JP=1,NCON(I)
   71 IITRIP(I,JP)=IHULL(I,JP)
      GO TO 79
   85 I6=1
      DO 72 JR=JV1,NCON(I)
      IITRIP(I,I6)=IHULL(I,JR)
   72 I6=I6+1
      DO 73 JR1=1,JV1-1
      IITRIP(I,I6)=IHULL(I,JR1)
   73 I6=I6+1
   79 CONTINUE
      DO 81 I=1,NT
      DO 81 J=1,INODE(I)-1
      II=ITRIP(I,J)
      IK=0
      DO 82 K=1,NCON(I)
   82 IF(II.EQ.IITRIP(I,K))IK=1
      IF(IK.EQ.1)GO TO 81
      IC=0
      CINS=999999.
      DO 90 NN=1,NCON(I)
      IC1=IITRIP(I,NN)
      IS1=IITRIP(I,NN+1)
      IF(NN.EQ.NCON(I))IS1=IITRIP(I,1)
      CINS1=((X(IC1)-X(II))**2+(Y(IC1)-Y(II))**2)**.5
      CINS2=((X(II)-X(IS1))**2+(Y(II)-Y(IS1))**2)**.5
      CINS3=((X(IC1)-X(IS1))**2+(Y(IC1)-Y(IS1))**2)**.5
      CINS4=CINS1+CINS2-CINS3
      IF(CINS4.GE.CINS)GO TO 90
```

```
      IO=IO1
   90 CONTINUE
      DO 91 L=1,NCON(I)
      IF(IO.NE.IITRIP(I,L))GO TO 91
      LL=L
      GO TO 92
   91 CONTINUE
   92 NCON(I)=NCON(I)+1
      DO 80 K=NCON(I),LL+2,-1
      IITRIP(I,K)=IITRIP(I,K-1)
   80 CONTINUE
      IITRIP(I,LL+1)=II
C     WRITE(6,*)IO,LL
   81 CONTINUE
      DO 2000 I=1,NT
      DO 2001 J=1,NCON(I)
      ITRIP(I,J)=IITRIP(I,J)
 2001 CONTINUE
      ITRIP(I,NCON(I)+1)=1
 2000 CONTINUE
      DO 342 KK=1,KK
      NT=INODE(KK)
      NNUD(I1A,KK)=NT
      DO 443 JJ=1,NT-1
      TLONG(KK)=TLONG(KK)+SQRT((X(ITRIP(KK,JJ))-X(ITRIP(KK,JJ+
     $1)))**2+(Y(ITRIP(KK,JJ))-Y(ITRIP(KK,JJ+1)))**2)
  443 CONTINUE
      TDI(I1A,KK)=TLONG(KK)
      TA1(I1A,KK)=PTOUR(ITRIP(KK,2),3)
  342 CONTINUE
      DO 1004 I1=1,KK
      DO 3001 J1=1,INODE(I1)
 3001 ITRIP1(I1A,I1,J1)=V(I1A,ITRIP(I1,J1))
      DO 3000 J22=1,NCON1(I1A,I1)
 3000 IHULL1(I1A,I1,J22)=V(I1A,IHULL(I1,J22))
 1004 CONTINUE
      WRITE(6,1006)I1A
 1006 FORMAT(///'******',I3,'. DAY')
      NKK(I1A)=KK
      DO 1005 I2=1,NKK(I1A)
      IG1=NNUD(I1A,I2)
      DCOST1=CK*TD1(I1A,I2)
      WRITE(6,4)I2,TD1(I1A,I2),TA1(I1A,I2),DCOST1
      DCOST=DCOST+DCOST1
      WRITE(6,*)(ITRIP1(I1A,I2,K),K=1,IG1)
      WRITE(6,5)
      IG2=NCON1(I1A,I2)
      WRITE(6,*)(IHULL1(I1A,I2,KG),KG=1,IG2)
 1005 CONTINUE
    4 FORMAT(//,'    TRIP',I3,10X,'LENGHT OF TRIP=',F8.2,
     $10X,'LOAD OF TRIP=',F5.0,10X,'DELIVERY COST=',F10.2)
    5 FORMAT('   CONVEX HULL')
      DO 1010 I=1,NARC
      ID(I,3)=0
 1010 IUSE(I)=0
      DO 1011 I=1,KK
      DO 1012 J=1,10
 1012 ITRIP(I,J)=0
      TLONG(I)=0.
 1011 INODE(I)=0
 1000 CONTINUE
C*************************************************************
C*                    PACKING SUBPROBLEM
C*************************************************************
      DO 1020 I=1,NV+1
```

```
      IF(NND(I).EQ.1)GO TO 1039
      NVR1=NKK(I)
      NYOK=0
      I2=0
      DO 1036 I3=1,NVR1
      DO 1037 J3=1,NV
 1037 P(J3,I3)=HK*TA1(I,I3)+TD1(I,I3)
      DO 1038 J4=1,NV+1
      R(J4,I3)=TD1(I,I3)
      IF(R(J4,I3).LE.B1(J4))GO TO 1038
      NYOK=NYOK+1
      I2=I2+1
      IYOK(I3)=1
      IHIRE1(I)=1
      IHIRE(I,I2)=I3
      HCOST=HCOST+P(I,I3)
 1038 CONTINUE
 1036 CONTINUE
C     DO 7000 I7=1,NV+1
C     WRITE(6,*)(P(I7,I8),I8=1,NVR1)
C7000 WRITE(6,*)(R(I7,I9),I9=1,NVR1)
      DO 1031 J=1,NV
      IKIS=J
      CALL KNAP(IKIS,P,R,B1,NYOK,IYOK,NVR1,XS1)
      IRN(I,J)=0
      TOP10=0.
      DO 1032 I1=1,NVR1
      IF(XS1(I1).EQ.0.)GO TO 1032
      TOP10=TOP10+R(NV+1,I1)
      IRN(I,J)=IRN(I,J)+1
      IROT(I,J,IRN(I,J))=I1
      NYOK=NYOK+1
      IYOK(I1)=1
 1032 CONTINUE
      VD(J,I)=B1(J)-TOP10
      IF(NYOK.EQ.NVR1)GO TO 1039
 1031 CONTINUE
      DO 1033 J1=1,NVR1
      IF(IYOK(J1).EQ.1)GO TO 1033
      IHIRE1(I)=1
      I2=I2+1
      IHIRE2(I)=I2
      IHIRE(I,I2)=J1
      HCOST=HCOST+HK*TA1(I,J1)+TD1(I,J1)
 1033 CONTINUE
 1039 DO 1080 IO=1,NV
      IF(VD(IO,I).EQ.0.)VD(IO,I)=B1(IO)
 1080 CONTINUE
      DO 1034 J2=1,NVR1
 1034 IYOK(J2)=0
 1030 CONTINUE
      DO 1035 I=1,NDAY
      DO 2035 J=1,NV
      WRITE(6,8)I,J,IRN(I,J),VD(J,I)
      WRITE(6,*)(IROT(I,J,K),K=1,IRN(I,J))
 2035 CONTINUE
      IF(IHIRE1(I).NE.1)GO TO 1035
      WRITE(6,2036)
    8 FORMAT(/,'***',I2,'. DAY',5X,'ROUTE ',I2,5X,'# OF TRIPS',I2,
     $5X,'SHADOW PRICE=',F5.0)
 2036 FORMAT(/'THESE TRIPS USE HIRED VEHICLE')
      KR=IHIRE2(I)
      WRITE(6,*)(IHIRE(I,KF),KF=1,KR)
 1035 CONTINUE
      DO 1040 I=1,NPT1
```

```
      DO 1040 J=1,NPT1
 1040 D(I,J)=((X1(I)-X1(J))**2+(Y1(I)-Y1(J))**2)**.5
C*****************************************************************
C*                    ASSIGNMENT SUBPROBLEM                      *
C*****************************************************************
      DO 1041 I=1,NDAY
      DO 1041 J=1,NKK(I)
      DO 1041 K=1,NCON1(I,J)
      IVAR=IHULL1(I,J,K)
 1041 ITRU(IVAR)=1
C     WRITE(6,*)(ITRU(I),I=1,NPT1)
      DO 1042 I=2,NPT1
      IF(ITRU(1).NE.1)GO TO 1043
      IAA=0
      DO 1048 IG=1,NDAY
      DO 1049 J=1,NV
      KI1=IRN(IG,J)
      DO 1050 JJ=1,KI1
      KJ1=IROT(IG,J,JJ)
      NCN=NCON1(IG,KJ1)
      DO 1051 IA=1,NCN
      IF(I.EQ.IHULL1(IG,KJ1,IA))IAA=1
 1051 CONTINUE
      IF(IAA.NE.1)GO TO 1050
      IG1=IG
      NV1=J
      ITRJ=KJ1
      GO TO 1043
 1050 CONTINUE
 1049 CONTINUE
 1048 CONTINUE
 1043 CONTINUE
      DO 1044 IG=1,NDAY
      DO 1045 J=1,NV
      IF(J.EQ.NV1.AND.IG.EQ.IG1)GO TO 1045
      KI=IRN(IG,J)
      IF(KI.EQ.0)GO TO 3050
      DO 1046 K=1,KI
      KJ=IROT(IG,J,K)
      NCN=NCON1(IG,KJ)
      IHULL1(IG,KJ,NCN+1)=IHULL1(IG,KJ,1)
      IF(K.NE.1)GO TO 1047
      NC=IHULL1(IG,KJ,1)
      NS=IHULL1(IG,KJ,2)
      TINS1=D(NC,I)+D(I,NS)-D(NC,NS)
 1047 CONTINUE
      DO 1046 K1=1,NCN
      NC=IHULL1(IG,KJ,K1)
      NS=IHULL1(IG,KJ,K1+1)
      TINS=D(NC,I)+D(I,NS)-D(NC,NS)
      IF(TINS.GE.TINS1)GO TO 1046
      TINS1=TINS
 1046 CONTINUE
      T(I,J,IG)=TINS1
      GO TO 1045
 3050 T(I,J,IG)=D(I,I)
 1045 CONTINUE
 1044 CONTINUE
      IF(ITRU(1).NE.1.OR.IAA.EQ.0)GO TO 1042
      KBK=1
      DO 1052 II1=1,NNOD(IG1,ITR1)
      II2=ITRIP1(IG1,ITR1,II1)
      IF(I.EQ.II2)GO TO 1052
      ITRIP(1,KBK)=ITRIP1(IG1,ITR1,II1)
      KBK=KBK+1
 1052 CONTINUE
```

```
      I13=NNOD(IG1,ITR1)-1
      CALL CHULL(1,X1,Y1,ITRIP,I13,NCCN,IHULL)
      DO 1057 JJ1=1,NCON(1)
 1057 IHULL1(I12,ITR1,JJ1)=IHULL(1,JJ1)
      I8=IRN(IG1,NV1)
      DO 1053 K8=1,I8
      L8=IROT(IG1,NV1,K8)
      NCN=NCUN1(IG1,L8)
      IF(L8.NE.ITR1)GO TO 1054
      NCN=NCON(1)
      IG2=NDAY+1
      IHULL1(IG2,L8,NCN+1)=IHULL1(IG2,L8,1)
      IF(K8.NE.1)GO TO 1055
      NC=IHULL1(IG2,L8,1)
      NS=IHULL1(IG2,L8,2)
      TINS1=D(NO,I)+D(I,NS)-D(NC,NS)
      GO TO 1055
 1054 IHULL1(IG1,L8,NCN+1)=IHULL1(IG1,L8,1)
      IF(K8.NE.1)GO TO 1055
      NC=IHULL1(IG1,L8,1)
      NS=IHULL1(IG1,L8,2)
      TINS1=D(NO,I)+D(I,NS)-D(NC,NS)
 1055 DO 1056 K9=1,NCN
      IF(L8.EQ.ITR1)IGG=IG2
      NC=IHULL1(IGG,L8,K9)
      NS=IHULL1(IGG,L8,K9+1)
      TINS=D(NO,I)+D(I,NS)-D(NO,NS)
      IF(TINS.GE.TINS1)GO TO 1056
      TINS1=TINS
 1056 IGG=IG1
 1053 CONTINUE
      T(I,NV1,IG1)=DEN1*TINS1
      IG1=0
      NV1=0
      ITR1=1
      DO 1070 KT=1,10
 1070 ITRIP(1,KT)=0
      I13=0
      DO 1058 KP=1,10
 1058 IHULL(1,KP)=0
      NCON(1)=0
 1042 CONTINUE
      WRITE(6,4021)SCOST,DCOST,HCOST
 4021 FORMAT(5X,'TOTAL STOCK-OUT COST=',F8.1,/,5X,'TOTAL DELIVERY
     $ COST=',F12.1,/,5X,'COST OF HIRING VEHICLE=',F8.1,///)
C     WRITE(6,*)111111
      DO 3002 I=2,NPT1
      COST1=99999999.
      DO 3003 K=1,NV
      DO 3004 IQ=1,NDAY+1
      IF(IQ.EQ.NDAY+1)GO TO 3005
      COST=(CK+DEN2*(700-VD(K,IC)))*T(1,K,IQ)+SL(I)*PR(I,IQ)
C     WRITE(6,*)88,COST
      GO TO 3006
 3005 COST=SL(I)*PR(I,NDAY+1)
C     WRITE(6,*)99,COST
 3006 IF(COST.GE.COST1)GO TO 3004
      COST1=COST
      NKS(I)=K
      NQS(I)=IQ
 3004 CONTINUE
 3003 CONTINUE
 3002 CONTINUE
      IF(ITER.NE.1)GO TO 5555
      DO 2050 I=1,NPT1
```

```
      WRITE(7,*)NQS(I),NKS(I)
      WRITE(7,*)(PR(I,JU),JU=1,4)
      DO 2050 J=1,NV
 2050 WRITE(7,*)(T(I,J,I15),I15=1,NDAY)
 5555 CONTINUE
C     WRITE(6,*)222222
      DO 3008 I=1,NDAY+1
      NND1(I)=1
 3008 V1(I,1)=1
C     WRITE(6,*)333333
      DO 3007 I=2,NPT1
      IDAY=NQS(I)
C     WRITE(6,*)444,IDAY
      NND1(IDAY)=NND1(IDAY)+1
C     WRITE(6,*)44,NND1(IDAY)
      V1(IDAY,NND1(IDAY))=I
C     WRITE(6,*)4,V1(IDAY,NND1(IDAY))
 3007 CONTINUE
C     WRITE(6,*)555555
      DO 3009 I=1,NDAY
      WRITE(6,3)I,NND1(I)
 3009 WRITE(6,*)(V1(I,J),J=1,NND1(I))
C     WRITE(6,*)666666
      DO 3012 IW=1,NDAY
      IHIRE1(IW)=0
      DO 3013 LC=1,IHIRE2(IW)
 3013 IHIRE(IW,LC)=0
      IHIRE2(IW)=0
      DO 3014 K=1,NV+1
      VC(K,IW)=0.
      DO 3015 LD=1,IRN(IW,K)
 3015 IRUT(IW,K,LD)=0
      IRN(IW,K)=0
      DO 3016 I=1,NPT1
      ITRU(I)=0
 3016 T(I,K,IW)=0.
 3014 CONTINUE
      DO 3017 IY=1,NKK(IW)
      DO 3018 IZ=1,NNOD(IW,IY)
 3018 ITRIP1(IW,IY,IZ)=0
      DO 3019 IX=1,NCON1(IW,IY)
 3019 IHULL1(IW,IY,IX)=0
      NNOD(IW,IY)=0
      NCON1(IW,IY)=0
 3017 CONTINUE
      NKK(IW)=0
 3012 CONTINUE
 4000 CONTINUE
      STOP
      END
      SUBROUTINE CNVRP(X,Y,NPT,NVT,TRA,TRCAP,PAR,*XDROP,C,DIV,HEIGHT,
     $WIDHT,ID,PTOUR,TDIST,ISUM,NARC,TRU)
      REAL D(100),S(2500),X(100),Y(100)
      INTEGER ID(8000,3),ADJA(100,50),TRA(5),TRCAP(5),TRU(5),PTOUR(10
     $0,4),POINT(100),IX(2500),JX(2500),C(100)
      INTEGER BX(100),BY(100),P
      COMMON /AREA2/S,IX,JX
      TRA(1)=9999999
      WIDHT=WIDHT/DIV
      HEIGHT=HEIGHT/DIV
      DO 61 I=2,NPT
      POINT(I)=0
      BX(I)=X(I)/WIDHT+1
   61 BY(I)=Y(I)/HEIGHT+1
      NARC=0
      II=0
```

```
      NT=NPT-1
      I=1
      DO 107 J=2,NPT
      NARC=NARC+1
      ID(NARC,1)=1
      ID(NARC,2)=J
      DISTAN=(X(I)-X(J))**2+(Y(I)-Y(J))**2
  107 D(J)=SQRT(DISTAN)
      DO 101 I=2,NT
      IP1=I+1
      DO 102 J=IP1,NPT
      IA=IABS(BX(I)-BX(J))
      IB=IABS(BY(I)-BY(J))
      IF(IA.GT.1.OR.IB.GT.1)GO TO 102
      NARC=NARC+1
      II=II+1
      POINT(I)=POINT(I)+1
      POINT(J)=POINT(J)+1
      IF(POINT(I).GT.100.OR.POINT(J).GT.100)GO TO 998
      ADJA(I,POINT(I))=II
      ADJA(J,POINT(J))=II
      ID(NARC,1)=I
      ID(NARC,2)=J
      DISTAN=(X(I)-X(J))**2+(Y(I)-Y(J))**2
      S(II)=D(I)+D(J)-PAR*SQRT(DISTAN)
  102 CONTINUE
  101 CONTINUE
      SUM=0.
      DO 25 J=2,NPT
   25 SUM=SUM+D(J)
      TOTST=2.*SUM
      NN=NARC-NPT+1
      TRU(1)=NPT-1
      IF(NVT.EQ.1)GO TO 17
      DO 15 I=2,NVT
   15 TRU(I)=0
   17 CONTINUE
      DO 2 I=2,NPT
      PTOUR(I,1)=I
      PTOUR(I,2)=1
      PTOUR(I,3)=Q(I)
      PTOUR(I,4)=1
    2 CONTINUE
      DO 21 I=1,NT
   21 ID(I,3)=2
      DO 22 I=NPT,NARC
   22 ID(I,3)=0
      DO 111 I=1,NN
      JX(I)=I
  111 IX(I)=I
      CALL STHEAP(NN)
   90 CONTINUE
      JG=0
      K=1
      IF(S(K).LE.0)GO TO 16
      P=IX(K)+NPT-1
      IN=ID(P,1)
      JN=ID(P,2)
      IPT1=PTOUR(IN,1)
      IPT2=PTOUR(JN,1)
      IF(ID(IN-1,3).EQ.0.OR.ID(JN-1,3).EQ.0)GO TO 12
      IF(IPT1.EQ.IPT2)GO TO 12
      ICAP=PTOUR(IN,3)+PTOUR(JN,3)
      DO 91 LP=1,NVT
      IF(TRCAP(LP).GE.ICAP.AND.TRU(LP).LT.TRA(LP))GO TO 10
   91 CONTINUE
```

```
      GO TO 12
   10 MCAP=PTOUR(IN,4)+PTOUR(JN,4)
      IF(MCAP.GT.MXDROP)GO TO 12
      TRU(LH)=TRU(LH)+1
      TRU(PTOUR(IN,2))=TRU(PTOUR(IN,2))-1
      TRU(PTOUR(JN,2))=TRU(PTOUR(JN,2))-1
      DISTAN=(D(IN)+D(JN)-S(K))/PAR
      TDIST=TDIST-(D(IN)+D(JN)-DISTAN)
      DO 103 L=2,NPT
      IF(PTOUR(L,1).EQ.IPT1.OR.PTOUR(L,1).EQ.IPT2)GO TO 64
  103 CONTINUE
   64 DO 105 M=2,NPT
      IF(PTOUR(M,1).NE.IPT1.AND.PTOUR(M,1).NE.IPT2)GO TO 105
      PTOUR(M,1)=PTOUR(L,1)
      PTOUR(M,2)=LH
      PTOUR(M,3)=ICAP
      PTOUR(M,4)=MCAP
  105 CONTINUE
      IC(P,3)=1
      IC(IN-1,3)=IC(IN-1,3)-1
      IC(JN-1,3)=IC(JN-1,3)-1
      IF(IC(JN-1,3).NE.0)GO TO 11
      IB=POINT(JN)
      JC=1
      DO 97 I=1,IB
      IC=JX(ADJA(JN,I))
      IF(S(IC).LE.0.)GO TO 97
      S(IC)=0.
      CALL OTHEAP(NN,IC)
   97 CONTINUE
   11 IF(IC(IN-1,3).NE.0)GO TO 12
      IB=POINT(IN)
      DO 98 I=1,IB
      IC=JX(ADJA(IN,I))
      IF(S(IC).LE.0.)GO TO 98
      S(IC)=0.
      CALL OTHEAP(NN,IC)
   98 CONTINUE
      GO TO 90
   12 IF(JC.EQ.1)GO TO 90
      S(K)=0.
      CALL OTHEAP(NN,K)
      GO TO 90
   16 CONTINUE
      ISUM=0
      DO 228 I=1,NVT
  228 ISUM=ISUM+TRU(I)
      GO TO 301
  998 WRITE(6,999)
  999 FORMAT('-',3X,'POINT(I)0 FOR SOME I EXCEEDS 100'//)
  301 CONTINUE
      DO 200 I=1,100
      D(I)=0.
      POINT(I)=0
      BX(I)=0
      BY(I)=0
      DO 200 J=1,50
  200 ADJA(I,J)=0
      P=0
      DO 201 I=1,NARC
      S(I)=0.
      IX(I)=0
  201 JX(I)=0
      RETURN
      END
      SUBROUTINE STHEAP(NN)
```

```
      REAL S(2500)
      INTEGER IX(2500),JX(2500)
      COMMON /AREA2/S,IX,JX
      IF(-2)50,60,1
    1 CONTINUE
      NH=NN/2
      DO 10 I=1,NH
      II=NH+1-I
      COPY=S(II)
      IXC=IX(II)
      JXC=IXC
    2 J=II+II
      IF(J-NN)5,7,9
    5 IF(S(J+1).GT.S(J))J=J+1
    7 IF(S(J).LE.COPY)GO TO 9
      S(II)=S(J)
      IX(II)=IX(J)
      JX(IX(J))=II
      II=J
      GO TO 2
    9 S(II)=COPY
      IX(II)=IXC
      JX(JXC)=II
   10 CONTINUE
   50 RETURN
   60 IF(S(1).GT.S(2))GO TO 50
      DT=S(1)
      S(1)=S(NN)
      S(2)=DT
      IX(1)=2
      JX(1)=2
      IX(2)=1
      JX(2)=1
      GO TO 50
      END
      SUBROUTINE OTHEAP(NN,IC)
      REAL S(2500)
      INTEGER IX(2500),JX(2500)
      COMMON /AREA2/S,IX,JX
      II=IC
      COPY=S(II)
      IXC=IX(II)
      JXC=IXC
   12 J=II+II
      IF(J-NN)15,17,19
   15 IF(S(J+1).GT.S(J))J=J+1
   17 IF(S(J).LE.COPY)GO TO 19
      S(II)=S(J)
      IX(II)=IX(J)
      JX(IX(J))=II
      II=J
      GO TO 12
   19 S(II)=COPY
      IX(II)=IXC
      JX(JXC)=II
   20 RETURN
      END
      SUBROUTINE BULERV(NZ,SSC,T,SIGMA,FRV)
      DIMENSION NZ(350)
      III=0
      SSC1=SSC
      FLUX1=1.0*10000001
    4 IF(SSC.LT.0.5)GO TO 2
    6 ZZ=(SSC-T)/(SIGMA*SQRT(T))
      IF(ZZ.LE.0.0)THEN
        IF(ZZ.GE.-3.49)THEN
```

```
          LL=-100.*ZZ+1.
          FLUX=1.-NZ(LL)/10000.0
        ELSE
          FLUX=0.000001

        ENDIF
      ELSE
        IF(ZZ.LE.3.49)THEN
          LL=100.0*ZZ+1.
          FLUX=NZ(LL)/10000.0
        ELSE
          FLUX=0.99999999
        ENDIF
      ENDIF
      III=III+1
      IF(III.EQ.1)FLUX1=FLUX
      FLUXT=FLUXT+FLUX
      SSC=SSC-1.0
      GO TO 4
    2 ET=(FLUXT-FLUX1)/FLUX1
      ERV=SSC1-FI
      SSC=SSC1
      RETURN
      END
      SUBROUTINE BULPR2(NZ,SC1,ERV,C,SIGMA,PR2)
      DIMENSION NZ(350),Z(350)
      PR2=0
      JJJ=0
      SC1=SC1-ERV
      ZZ=(SC1-0)/(SIGMA*SQRT(C))
      IF(ZZ.LE.0.0)THEN
        IF(ZZ.GE.-3.49)THEN
          LL=-100*ZZ+1.
          FLUX=1.-NZ(LL)/10000.0
        ELSE
          FLUX=0.000001
        ENDIF
      ELSE
        IF(ZZ.LE.3.49)THEN
          LL=100.0*ZZ+1.
          FLUX=NZ(LL)/10000.0
        ELSE
          FLUX=1.9999999
        ENDIF
      ENDIF
      PR2=1-FLUX
      RETURN
      END
      SUBROUTINE KNAP(IKIS,P,R,B1,NYOK,IYOK,NVR1,XS1)
      DIMENSION B1(6),BB(5),PCR(50),NIN(50)
      DIMENSION IA(50),XS(50),Z(100),X1(50),RR(50),XX(100)
      DIMENSION XS1(50),IYOK(50),B(5)
      INTEGER P(10,50),R(10,50)
      II=IKIS
      B(II)=B1(II)
      DO 30 I=1,NVR1
      PCR(I)=P(II,I)/R(II,I)
      IF(IYOK(I).LE.1)PCR(I)=0.
   30 CONTINUE
      DO 40 I=1,NVR1
   40 NIN(I)=I
      DO 31 J=2,NVR1
      I=J-1
      DUM=PCR(J)
      NDUM=NIN(J)
      PDUM=P(II,J)
```

```
      RDUR=R(II,J)
   32 IF(DUR.LE.PDR(I))GO TO 33
      PDR(I+1)=PDR(I)
      NTN(I+1)=NTN(I)
      P(IT,I+1)=P(II,I)
      R(IT,I+1)=R(II,I)
      I=I-1
      IF(I.GT.0)GO TO 32
   33 PDR(I+1)=DUR
      NTN(I+1)=NDUR
      P(IT,I+1)=PDUR
      R(II,I+1)=RDUR
   31 CONTINUE
      NVR=NVR1-NYGK
      SUM1=0.
      DO 100 I=1,NVR
  100 SUM1=SUM1+R(II,I)
      IF(SUM1.GT.B(II))GO TO 101
      DO 102 I=1,NVR
  102 XS(I)=1.
      GO TO 70
  101 L=1
      DO 1 I=1,NVR
    1 X1(I)=2.
      SUM=0.
      OBJ=0.
      DIF=B(IT)
      DO 2 I=1,NVR
      IR=I
      IF(DIF.EQ.0.)III=1
      IF(DIF.LT.R(II,I))GO TO 3
      OBJ=OBJ+P(II,I)
      SUM=SUM+R(II,I)
      DIF=B(II)-SUM
    2 CONTINUE
    3 XO=OBJ
      RR(1)=IR
      X1(IR)=0.
      IA(IR)=1.
      IF(IR.NE.1)GO TO 29
      XS(IR)=0.
      GO TO 42
   29 DO 4 I=1,IR-1
    4 XS(I)=1
   42 DO 5 I=IR+1,NVR
    5 XS(I)=0.
      XS(IR)=0.
      IF(III.EQ.1)GO TO 70
   20 XR=0.
      OBJ=0.
      BB(II)=B(II)
      DO 7 LL=1,L
      OBJ=OBJ+(P(II,RR(LL))*X1(RR(LL)))
    7 BB(II)=BB(II)-(R(II,RR(LL))*X1(RR(LL)))
      IF(BB(II).LT.0.)GO TO 9
      DO 6 I=1,NVR
      IR=I
      IF(IA(I).EQ.1)GO TO 6
      IF(BB(II).LT.R(II,I))GO TO 8
      BB(II)=BB(II)-R(II,I)
      OBJ=OBJ+P(II,I)
    6 CONTINUE
    8 XR=BB(II)/R(II,IR)
      Z(L+1)=OBJ+XR*P(II,IR)
      IF(Z(L+1).LE.XO)GO TO 9
      IF(XR.NE.0.)GO TO 17
```

```
      XO=Z(L+1)
      IF(IR.EQ.1)GO TO 74
      DO 11 I=1,IR-1
      IF(IA(I).EQ.1)GO TO 12
      XS(I)=1.
      GO TO 11
   12 XS(I)=X1(I)
   11 CONTINUE
      IF(IR.EQ.NVR)GO TO 71
   74 DO 13 I=IR+1,NVR
      IF(IA(I).EQ.1)GO TO 14
      XS(I)=0.
      GO TO 13
   14 XS(I)=X1(I)
   13 CONTINUE
      XS(IR)=0.
      GO TO 73
   71 TOPL=0.
      DO 44 I=1,NVR-1
   44 TOPL=TOPL+(R(II,I)*XS(I))
      IF(TOPL.EQ.B(II))GO TO 43
      XS(IR)=1.
      GO TO 73
   43 XS(IR)=0.
   73 CONTINUE
      GO TO 9
   10 XX(L)=0BJ
      IF(XX(L).LE.XO)GO TO 19
      XO=XX(L)
      IF(IR.EQ.1)GO TO 72
      DO 15 I=1,IR-1
      IF(IA(I).EQ.1)GO TO 16
      XS(I)=1.
      GO TO 15
   16 XS(I)=X1(I)
   15 CONTINUE
      IF(IR.EQ.NVR)GO TO 201
   72 DO 17 I=IR+1,NVR
      IF(IA(I).EQ.1)GO TO 18
      XS(I)=0.
      GO TO 17
   18 XS(I)=X1(I)
   17 CONTINUE
      XS(IR)=1.
      GO TO 204
  201 TOP1=0.
      DO 202 KD=1,NVR-1
  202 TOP1=TOP1+(R(II,KD)*XS(KD))
      IF((B(II)-TOP1).LT.R(II,NVR))GO TO 203
      XS(IR)=1.
      GO TO 204
  203 XS(IR)=0.
  204 CONTINUE
   19 L=L+1
      RR(L)=IR
      X1(IR)=0.
      IA(IR)=1
      IF(IR.EQ.NVR)GO TO 9
      GO TO 20
    9 IF(X1(RR(L)).NE.0)GO TO 21
      X1(RR(L))=1.
      GO TO 20
   21 IF(L.EQ.1)GO TO 70
      X1(RR(L))=2.
      IA(RR(L))=1.
```

```
      GO TO 9
   73 DO 103 I=1,NVR1
  103 XSL(NIN(I))=XS(I)
      DO 104 I=1,NVR1
      IA(I)=0
  104 XS(I)=0
      RETURN
      END
      SUBROUTINE CHULL(NT,X,Y,ITRIP,INODE,NCON,IHULL)
      REAL X(100),Y(100)
      INTEGER ITRIP(100,10),INODE(100),NCON(100),IHULL(100,10)
      DO 1 IT=1,NT
      IJ=1
      II=ITRIP(IT,1)
      BEST=X(II)
      IFND=II
      XLAST=X(II)
      NSTAR=II
      JJ=INODE(IT)-1
      DO 2 JK=2,JJ
      J=ITRIP(IT,JK)
      IF(X(J).GE.BEST)GO TO 3
      NSTAR=J
      BEST=X(J)
    3 IF(X(J).LE.XLAST)GO TO 2
      NFND=J
      XLAST=X(J)
    2 CONTINUE
      N=NSTAR
      IHULL(IT,IJ)=NSTAR
      NCON(IT)=IJ
    4 BEST=-10000.
      IP=INODE(IT)-1
      DO 5 IC=1,IP
      I=ITRIP(IT,IC)
      IF(X(I)-X(N))5,60,70
   60 IF(Y(I).EQ.Y(N))GO TO 5
      IF(Y(I).GT.Y(N))SLOPE=9999.
      IF(Y(I).LT.Y(N))SLOPE=-9999.
      GO TO 8
   70 SLOPE=(Y(I)-Y(N))/(X(I)-X(N))
    8 IF(SLOPE.LE.BEST)GO TO 5
      BEST=SLOPE
      NNEXT=I
    5 CONTINUE
      IJ=IJ+1
      IHULL(IT,IJ)=NNEXT
      NCON(IT)=IJ
      N=NNEXT
      IF(N.NE.NFND)GO TO 4
   11 BEST=-10000.
      DO 12 IS=1,IP
      I=ITRIP(IT,IS)
      IF(X(I)-X(N))13,14,12
   14 IF(Y(I).EQ.Y(N))GO TO 12
      IF(Y(I).LT.Y(N))SLOPE=9999.
      IF(Y(I).GT.Y(N))SLOPE=-9999.
      GO TO 15
   13 SLOPE=(Y(I)-Y(N))/(X(I)-X(N))
   15 IF(SLOPE.LE.BEST)GO TO 12
      BEST=SLOPE
      NNEXT=I
   12 CONTINUE
      IF(NNEXT.EQ.NSTAR)GO TO 1
      IJ=IJ+1
      IHULL(IT,IJ)=NNEXT
```

```
      NCONF(I)=IJ
      N=NNEXT
      GO TO 11
    1 CONTINUE
      RETURN
      END
```

APPENDIX II

INVENTORY ROUTING PROBLEM
DATA FILE

NUMBER OF POINTS => NPT: ___

OF VEHICLES AVAILABLE => TRA: ___

CAPACITY OF VEHICLE => TRCAP: ___

MAX. # OF DROPS ON A TRIP => MXDROP: ___

RANGE OF X-COORDINATE => WIDTH: ____.

RANGE OF Y-COORDINATE => HEIGHT: ____.

		X-COORDINATE ____.__	Y-COORDINATE ____.__	DEMAND ____.__
SUPPLY POINT :	1 *	.	.	9999.9
DEMAND POINTS :	2 *	.	.	.
	3 *	.	.	.
	4 *	.	.	.
	5 *	.	.	.
	6 *	.	.	.
	7 *	.	.	.
	8 *	.	.	.
	9 *	.	.	.
	10*	.	.	.
	11*	.	.	.
	12*	.	.	.
	13*	.	.	.
	14*	.	.	.
	15*	.	.	.
	16*	.	.	.
	17*	.	.	.
	18*	.	.	.
	19*	.	.	.
	20*	.	.	.
	21*	.	.	.
	22*	.	.	.
	23*	.	.	.
	24*	.	.	.
	25*	.	.	.
	26*	.	.	.
	27*	.	.	.
	28*	.	.	.
	29*	.	.	.
	30*	.	.	.
	31*	.	.	.
	32*	.	.	.
	33*	.	.	.
	34*	.	.	.
	35*	.	.	.
	36*	.	.	.
	37*	.	.	.
	38*	.	.	.
	39*	.	.	.
	40*	.	.	.
	41*	.	.	.
	42*	.	.	.
	43*	.	.	.
	44*	.	.	.
	45*	.	.	.

APPENDIX III

i	P_{i1}	P_{i2}	P_{i3}	P_{i4}
2	.99	.99	.99	.99
3	.99	.99	.99	.99
4	.99	.99	.99	.99
5	.63	.99	.99	.99
6	.5	.99	.99	.99
7	.5	.99	.99	.99
8	.99	.99	.99	.99
9	.5	.99	.99	.99
10	.5	.99	.99	.99
11	.63	.99	.99	.99
12	.99	.99	.99	.99
13	.63	.99	.99	.99
14	.02	.99	.99	.99
15	.99	.99	.99	.99
16	.53	.99	.99	.99
17	$\simeq 0$	0	.5	.99
18	$\simeq 0$	.52	.99	.99
19	$\simeq 0$	.64	.99	.99
20	.99	.99	.99	.99
21	$\simeq 0$	0	.5	.99
22	.63	.99	.99	.99
23	$\simeq 0$	.59	.99	.99
24	.99	.99	.99	.99
25	.99	.99	.99	.99
26	$\simeq 0$	.77	.99	.99
27	.99	.99	.99	.99
28	.99	.99	.99	.99
29	.47	.99	.99	.99
30	$\simeq 0$	0	.64	.99
31	$\simeq 0$	.13	.99	.99
32	.99	.99	.99	.99
33	.71	.99	.99	.99
34	$\simeq 0$	0	.36	.99
35	$\simeq 0$	.52	.99	.99
36	$\simeq 0$	0	.53	.99
37	$\simeq 0$	.59	.99	.99
38	$\simeq 0$	.52	.99	.99
39	.99	.99	.99	.99
40	.99	.99	.99	.99
41	.99	.99	.99	.99
42	.99	.99	.99	.99
43	.95	.99	.99	.99
44	.27	.99	.99	.99
45	.95	.99	.99	.99
46	$\simeq 0$	.11	.99	.99
47	$\simeq 0$	.59	.99	.99
48	.99	.99	.99	.99
49	4×10^{-4}	.99	.99	.99
50	$\simeq 0$	.51	.99	.99