A MINIMUM 1-ARBORESCENCE ALGORITHM FOR NON-SYMMETRIC CARRIER-DISPATCHING PROBLEMS

by

PETER JIN-HER WAN

B.E. (I.E.), Chung Yuan Christian College
of Science and Engineering, Chung Li, Taiwan,
Republic of China, 1968

9984

A MASTER'S REPORT

submitted to partial fulfillment of the
requirement for the degree

MASTER OF SCIENCE

Department of Industrial Engineering
Kansas State University
Manhattan, Kansas

1972

Approved by:

Said Ashour

Major Professor

ACKNOWLEDGEMENT

Sincere gratitude goes to Dr. Said Ashour for the excellent guidance and patience he extended during this research. Sincere thanks also go to Professor Stephen Konz, Industrial Engineering; Professor J. J. Smaltz, Department of Industrial Engineering; and Professor A. H. Duncan, Department of Mechanical Engineering for their kind patronage.

I greatly thank Ms. Marie Jirak for her assistance in typing.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

Carrier-dispatching problem is one of the most frequently occuring real-world problems in the transportation field. The problem is of interest because it can be applied not only to the trucking industry but also to railways, airlines and waterway shipping. It is worth mentioning that the average annual expenditure of trucking industry alone is over $42 billion annually. Hence, the development of an economical transportation system becomes more necessary.

Several approaches have been applied to carrier-dispatching problems in recent year, but neither one is powerful enough to handle one of the realistic size problems. It is not that these problems are mathematically very complicated to state, but they can usually be formulated into a simple mathematical programming form, for which methods of solution are readily available. Obviously, the difficulty lies in the combinatorial nature of the problem.

The carrier-dispatching problem can be viewed as a special case of a classical traveling salesman problem under resource constraints. It is a problem of determining the optimal delivery or pick-up routes in which a number of destinations or demand points to be covered by various carriers such that a certain objective is optimized. Among the objectives usually considered are: (1) the minimization of total time traveled, (2) the minimization of total traveling cost, (3) the minimization of total traveled distance, and (4) the minimization of total number of dispatched carriers. The carrier-dispatching problem has been approached by branch-and-bound, simulation, integer programming, dynamic programming, and heuristics.

In this research, the carrier-dispatching problem has been solved by a heuristic algorithm. The objective is to minimize the total traveling cost. This indicates that the number of dispatched carriers in the routes is to be minimized at same time.

1.1 Problem Definition:

The simplest form of carrier-dispatching problems may be the classical traveling salesman problem in which a salesman may visit each of n destinations or demand points D_1, D_2, ..., D_n, once, and only once, starting from and terminating at the same depot or office. In general, the salesman may be a vehicle, school bus, truck, or an airplane. The operation may be one of "delivery" (e.g. merchandise delivery), one of "pick-up (e.g. trash service), one combining both "delivery" and "pick-up" (e.g. distilled water service), or one involving neither (e.g. house calls of physician).

In real-world situations, the classical traveling salesman problem may be modified by some added characteristics (resource constraints). For example, there may be a limited cost (distance for each trip). Thus, the salesman must determine his route within a minimized cost as well as satisfying the limitation. Moreover, as in problems involving delivery or pick-up of commodities d_1, d_2, ..., d_n, at destinations D_1, D_2, ..., D_n, and usually there is a capacity q of each carrier and when $d_i > q$, it will be necessary to make one or more return to the depot or office to get some more commodities.

When d_1, d_2, ..., d_n represent the quantities of commodity to be delivered or picked-up at destinations D_1, D_2, ..., D_n, various situations arise depending on whether the entire quantity must be served in a single visit or whether the splitting of the total requirement into two or more partial serving is permissible. Furthermore, in some dispatching problems there is a whole set of entities to be dispatched, as for instance, a number of carriers or a set of crews. Another important variation among carrier-dispatching problems concerns the objective function. A solution with respect to the minimum distance traveled, least time of travel, or minimum number of dispatching carriers, may be desired.

According to the nature of the carrier-dispatching problem, it can be briefly categorized as follows:

1. Single-depot problems. There are several destinations or demand points and only one depot in which commodities are stored. The salesman determines optimal routes which cover every destination or demand point, starting from and ending at the depot, delivering or picking-up the commodity.

2. Multi-depot problems. It is a combination of single-depot problems. There are more than one depot, a carrier can start from any one of them but must return back to the same depot, several routes are formed which satisfy the demand of every destination or demand point under certain constraints.

3. Line-haul problems. Such problems are modification of multi-depot problems. Every destination or demand point can be used for

shipping as well as receiving commodity to and from each other, that is, every destination can be a depot as in single-and multi-depot problems. The problem perhaps can be stated as follows: it is required to deliver or pick-up commodity to their respective destinations or demand points such that the total traveled cost (distance) is minimized.

1.2 Literature Review

The carrier-dispatching problem is a generalized traveling salesman problem as stated above. The major methods proposed for solving the carrier-dispatching problem are branch-and-bound [9], simulation [2,9,14,17], integer programming [1,7], dynamic programming [11], and heuristic programming [3,4,5,6,9,10,15,16].

Hayes [9] has employed the branch-and-bound approach proposed by Little et. al. [13] for solving the traveling salesman problem. The method is also applicable for non-symmetrical cost or distance matrix. In this algorithm, Hayes incorporated two of the possible modifications which were suggested by Little, et al. The first was a "go to the right" modification which enabled more nodes to be examined with decreased computation time. Since the size of the tree grew so fast with an increase in the number of demand points, the second modification was "throw away the tree" modification of the "go to the right" method. This modification reduced the number of nodes which had to be stored in memory during the solution process. This method is not successfully applied to large size carrier-dispatching problems.

Braun [2] has formulated a simulation approach to the carrier-dispatching problem. It begins with a random generation of the sequence of stops and assigns available carrier with sufficient capacity to them and then using a traveling salesman problem technique to find optimal tours. It was noted that poor solution was obtained as the problem size increases.

Integer programming seems to be an appropriate method to solve the carrier-dispatching problem. Balinski and Quandt [1] have developed an integer programming model based on Gomory's algorithm [7]. They define m activities A_i, $i = 1, 2, \ldots, m$, and an n-dimensional column vector a_{ij}, with 0 and 1 variable. When the j-th demand point is in i-th activity, $a_{ij} = 1$; otherwise, $a_{ij} = 0$. If C_i, where $i = 1, 2, \ldots, n$, is the cost of an activity, the carrier-dispatching problem can be formulated as:

$$\text{Minimize} \quad \sum_{i=1}^{m} W_i C_i$$

$$\text{Subject to} \quad \sum_{i=1}^{n} W_i a_{ij} = 1 \qquad \text{where } j = 1, 2, \ldots, n,$$

where n is the number of demand points and m is the number of routes. W_i is zero-one variable having a value 1 if the activity A_i is used; zero otherwise, the limitation of this method are : (1) the model consists of a large number of constraints even for a small size problem, (2) it makes the solution undesirable when the problem has other constraints because it just considers the activity constraint. It is possible to improve this method by using linear programming to obtain a lower bound on the total cost and then using 0-1 integer programming to solve it.

Held and Karp [11] have formulated a dynamic programming model to the traveling salesman problem and mentioned that it may be extended to a multi-loop traveling salesman problem (a traveling salesman problem with more than one salesman). Hayes [9] has examined such an idea and developed a "straight forward" method to solve the carrier-dispatching problems. It is also just good for small size problems.

Dantzig and Ramser [5] have first applied heuristic programming to the carrier-dispatching problem. In this method, each demand point is assigned to a fixed route and combined into pairs to detect likely links in the distance (cost) matrix based on the proximity of any two demand points. And then the constraints of the links are checked. After all demand points are in respective route, the total distance (cost) of each route is minimized by a traveling salesman problem technique. Clarke and Wright [3] have felt that Dantzig and Ramser ignored the idea of the minimizing total traveled distance (cost), so they have shifted the emphasis to minimizing total traveled distance (cost) by using the savings technique. Their method is feasible for solving practical size problems. Cochran [4] has modified Clarke and Wrights' method. He has added a route restriction on the traveled distance of each trip as a constraint and developed a truck reassignment routine to improve the previous assignment. This method has been successfully extended by Hering [10] by using a "look ahead feature" for forming routes.

The literature mentioned above are concerned with the single-depot carrier-dispatching problems. A multi-terminal delivery problem algorithm, developed by Tillman [15], is based on the savings concept as in Clarke and Wrights' [3]. At present, there is little research on the multi-terminal delivery problem.

1.3 Proposed Research

As shown above, the carrier-dispatching problem can be solved by many methods. By examining these approaches,it appears that the heuristic programming approach is the most practical procedure at present because of their computational feasibility. With this in mind, we propose a heuristic algorithm based on the graph-theoretical concept.

In this report, two algorithms using the minimum arboresence to form an optimal tour covering all demand points and then partitioning the tour into several subtours to satisfy the capacity constraints, are proposed. Both single-and multi-depot carrier-dispatching problems have been considered. The algorithm is illustrated by sample problems. In addition, a geometrical search to the savings approach will be examined. It is conjectured that this search will help reduce the computational effort involved.

CHAPTER II

NON-SYMMETRIC SINGLE-DEPOT CARRIER-DISPATCHING PROBLEMS

The purpose of this chapter is to develop a heuristic method for solving the non-symmetric single-depot carrier-dispatching problem. This approach is based on minimum 1-arborescence and trial-and-error method.

The concept, definitions, and notation are provided in the balance of this chapter. A sample problem will demonstrate clearly the approach and the associated algorithm stated in formal steps.

2.1 Basic Concepts

The non-symmetric single depot carrier-dispatching problem is basically one of determining routes for a number of carriers, delivering or picking-up commodities from a depot (terminal) to a set of given destinations (demand points) such that the total traveling cost is minimized and the following conditions are satisfied: (1) the number of available carriers and the carrying capacities are known, (2) the demand of each demand point is known, (3) a given demand point can only appear on one route, and (4) the routes to be determined must all be either "delivery" or "pick-up" routes and not both.

The proposed approach consists of two main phases. Phase I consists of developing a minimum 1-arborescence graph to find the optimal tour for all destinations or demand points. In phase II, the resulted tour is decomposed by trial-and-error to a number of subroutes such that all constraints are satisfied. In order to follow the discussion easily, it is necessary to introduce some definitions as follows:

Tree. A tree is defined as a connected graph of n points with n-1 edges.

Arborescence. An arborescence is a directed graph on the vertex set 1, 2, ..., n such that exactly one edge is directed into each vertex, there is no cycle pass through any vertex.

Spanning arborescence. A spanning arborescence for a graph is one which covers the graph (that is, it consists of every point of the graph).

1-arborescence. A 1-arborescence is an arborescence with an additional node connected to it by two edges. The one with a minimum cost is the minimum 1-arborescence.

Degree of node. The number of arcs which connect to a node is referred to as the degree of a node.

Out-of-kilter node. A node has more than two degrees is considered as out-of-kilter high. A node has one degree is considered as out-of-kilter low.

In-kilter node. A node of degree two with both edges having the same direction.

Front-end-of-edge rule. In an arborescence, each edge has to be directed toward only one of the nodes.

The objective in this approach is to find out a tour or a route from the minimum 1-arborescence graph and then decompose the tour to some subroutes. Each tour is a 1-arborescence and a 1-arborescence is a tour if, and only if, each of the nodes is in-kilter. It is, therefore, true that a minimum 1-arborescence which is a tour, is a solution to the carrier-dispatching problem given that all demand points are in one tour. Consider a vector π and a gap function $f(\pi)$ are associated with the minimum 1-arborescence. Let π be a real column vector having all zero elements in the initial state of the minimum 1-arborescence. Each element represents the cost of the difference between two different states. A gap function $f(\pi)$ is a function which is equal to the difference between the cost of a minimum tour with respect to a weight vector π and minimum 1-arborescence with respect to the same weight π. Hence,

$$f(\pi) = w + 2 \sum_{i=1}^{n} \pi_i - \min [D_k + \sum_{i=1}^{n} \pi_i \, d_{ik}], \tag{1}$$

where w is the cost of a minimum tour with respect to the cost elements $d(i,j)$, D_k is the cost of the k-th 1-arborescence with respect to the cost elements $d(i,j)$ and d_{ik} is the degree of node in the 1-arborescence. The appropriate choice of the vector π which minimizes the gap function $f(\pi)$ will eventually lead a minimum 1-arborescence to an optimal tour. From (1), we get

$$f(\pi) = w - \min [D_k + \sum_{i=1}^{n} \pi_i \, (d_{ik} - 2)]. \tag{2}$$

Let

$$\min [D_k + \sum_{i=1}^{n} \pi_i \, (d_{ik} - 2)] = w(\pi) \quad \text{and} \quad d_{ik} - 2 = v_{ik}.$$

This problem is equivalent to the following linear programming problem:

$$\text{Maximize} \quad w(\pi)$$

$$\text{Subject to} \quad w \leqq D_k + \sum_{i=1}^{n} \pi_i \, v_{ik} \tag{3}$$

$$\text{where} \quad k = 1, 2, \ldots, q.$$

Dualizing, we obtain

Minimize $\sum_k D_k y_k$

Subject to $y_k \geq 0$

$$\sum_k y_k = 1 \tag{4}$$

and $\sum (-v_{ik}) y_k = 0$

where $i = 2, 3, \ldots, n-1.$

2.2 Sample Problem

The proposed algorithm for the non-symmetric single-depot carrier-dispatching problem is illustrated by a sample problem. Consider a problem of 6 demand points. The corresponding cost matrix and resource constraints are given in Tables 2.1 through 2.3. The solution is formed step by step as follows:

Step 1. Construct the initial minimum 1-arborescence. First, a minimum spanning arborescence of the cost matrix is constructed and then two links are drawn from the arcs associated with the demand point D_1 connected to the minimum spanning arborescence. The minimum spanning arborescence is constructed by choosing the minimum cost elements from the cost matrix. Also, it is important that each of nodes 2,3, ..., 6 is in the spanning arborescence and there is no loop formed by the node and each link has to be directed toward only one of the nodes. In constructing the minimum 1-arborescence, the last two links are chosen from node 1 satisfying the condition that the sum of two links has a minimum

Table 2.1 Cost Matrix of the Sample Problem

From To	T	1	2	3	4	5	6
T	-	2	7	4	8	11	10
1	3	-	27	43	16	30	26
2	6	7	-	16	1	30	25
3	3	20	13	-	15	5	0
4	6	21	16	25	-	18	18
5	10	12	46	27	48	-	5
6	15	23	5	5	9	5	-

Table 2.2 Demand of Each Point

Demand point	1	2	3	4	5	6
Quantity required	12	12	10	8	10	8

Table 2.3 Truck Allocation

Capacity	25
Available	3

value of $\sum d(1,J) + d(I,1)$ $I,J = 2,3, \ldots, 6$ and there is no conflict to the front-end-of-edge rule. If there are more than one spanning 1-arborescence with the same cost, the corresponding 1-arborescence is formed for each one and the one with the minimum cost is the minimum 1-arborescence. The minimum spanning arborescence and the minimum 1-arborescence are shown on Figure 2.1_a and 2.1_b. We set level $L = 0$ and compute the lower bound at this level such that

$$C^0 = \sum_{p,q} d(p, q), \qquad (p,q) \in A^0,$$

where A^0 is the initial 1-arborescence, or

$$C^0 = d(1,4) + d(2,1) + d(6,2) + d(3,6) + d(3,5) + d(4,3)$$

$$= 16 + 7 + 5 + 0 + 5 + 25 = 58.$$

Step 2. Identify the out-of-kilter nodes of the minimum 1-arborescence. In this sample problem, node (5) is out-of-kilter low and node (3) is out-of-kilter high.

Step 3. Compute the cost increments of the out-of-kilter nodes. If node (k) is out-of-kilter low using the following formula is used:

$$\delta^L(k) = \min_{(k,\ell)\in A^L} [d(k,\ell) - \max\,(d(r,s))],$$

where $(k) \neq (r), (s)$ and (r,s) is a set of links which can be replaced by link (k,ℓ) without forming a loop among nodes $2,3, \ldots, 6$ but a loop must be formed at node 1 and the link (k,ℓ) must satisfy the front-end-of-edge rule. In our example,

$$\delta^0(5) = d(5,6) - d(3,6) = 5.$$

However, if node (k) is out-of-kilter high, the increment of the cost of the minimum 1-arborescence is computed such that

$$\delta^L(k) = \min_{(k,\ell)\varepsilon A^L} [\min (d(r,s) - d(k,\ell)],$$

where (r,s) is a set of links which can possibly replace arc (k,ℓ) without forming a loop over nodes 2,3, ..., 6 but form a loop with node (1) and which does not create any conflict with the front-end-of-edge rule. Node (3) is out-of-kilter high and hence, the increment is

$$\delta^0(3) = d(5,6) - d(3,6) = 5.$$

Step 4. Select a node having the minimum cost increment

$$\delta^*(k) = \min [\delta^L(k)],$$

or

$$\delta^*(3) = \delta^*(5) = 5.$$

As a tie exists between nodes (3) and (5), we select node (5) by random, and then set L = 1. The corresponding lower bound is updated such that

$$C^{L+1} = C^L + \delta^*(k) = 58 + 5 = 63.$$

Step 5. Find the optimal tour. Since at this level all nodes are in-kilter, the tour is then obtained such that $1 \rightarrow 4 \rightarrow 3 \rightarrow 5 \rightarrow 6 \rightarrow 2 \rightarrow 1$. The cost of the optimal tour is $C^* = C^1 = 63$ (see Figure 2.1_c at L = 1). If the search is failed, repeat step 2 till a minimum 1-arborescence represents a tour.

Step 6. Calculate all the cost $d(S_i)$ of the possible subroutes

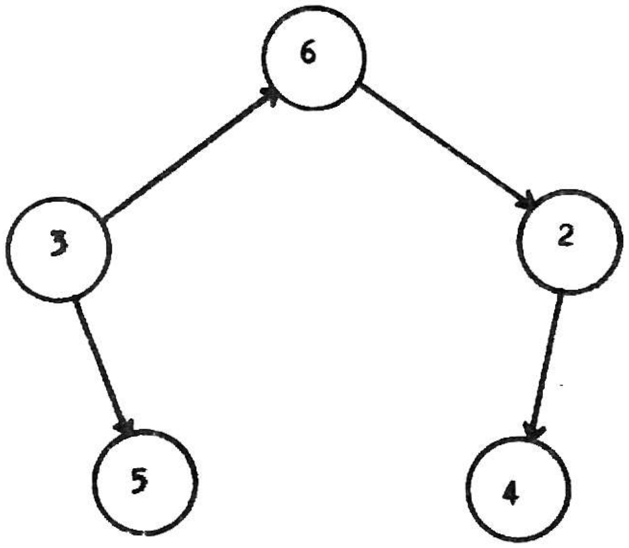


Figure 2.1a Minimum Spanding Arborescence

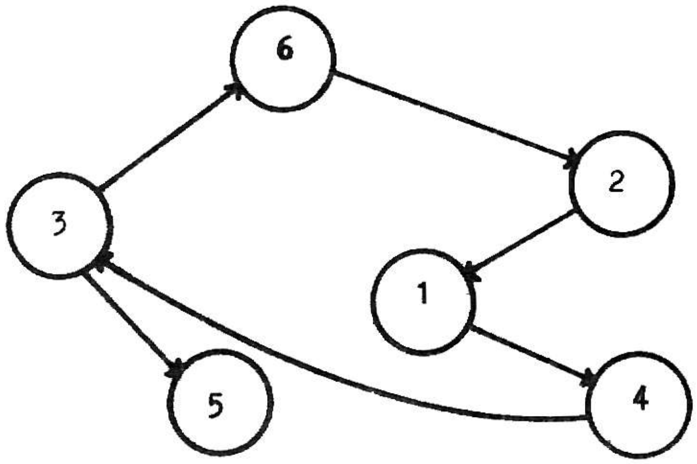


Figure 2.1$_b$ Minimum 1-Arborescence

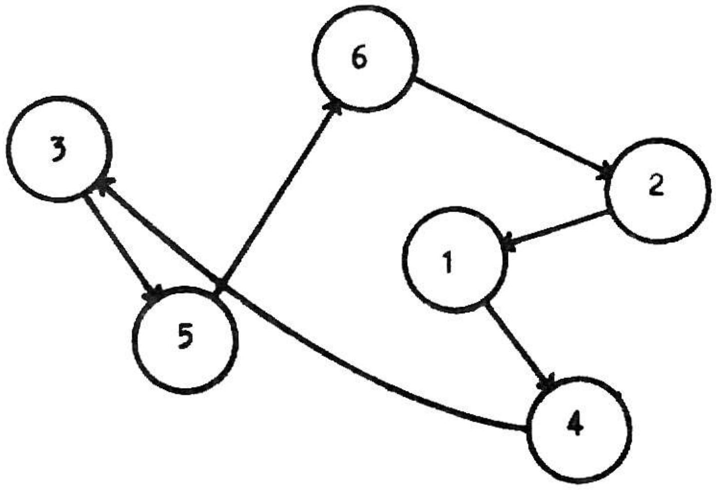


Figure 2.1$_c$ Minimum Tour

about the depot including one demand point only and arrange it in descending order

$S_i = (i_T, i_k, i_T)$ where $k = 1, 2, \ldots, n,$
$j = 1, 2, \ldots, n,$

or

$S_1 = (i_T, 6, i_T)$ $d(S_1) = 25$

$S_2 = (i_T, 5, i_T)$ $d(S_2) = 21$

$S_3 = (i_T, 4, i_T)$ $d(S_3) = 14$

$S_4 = (i_T, 2, i_T)$ $d(S_4) = 13$

$S_5 = (i_T, 3, i_T)$ $d(S_5) = 7$

$S_6 = (i_T, 1, i_T)$ $d(S_6) = 5.$

The available carrier with the largest capacity to the first subroute S_1 is then assigned.

Step 7. Expand the subroute S_1 by adding some demand points one by one according to the related position in the resulted tour about node (6). Each added node should satisfy the capacity of the carrier; otherwise, another new subroute has to be set up by S_2. The capacity of the carrier is checked with the total demand in one route till all demand points have been covered in the routes. In our sample problem, it can be stated as follows:

S_1	S_3	S_2	Total Cost
$T \to 6 \to 2 \to T$	$T \to 1 \to 4 \to T$	$T \to 3 \to 5 \to T$	64
$T \to 5 \to 6 \to T$	$T \to 2 \to 1 \to T$	$T \to 4 \to 3 \to T$	82

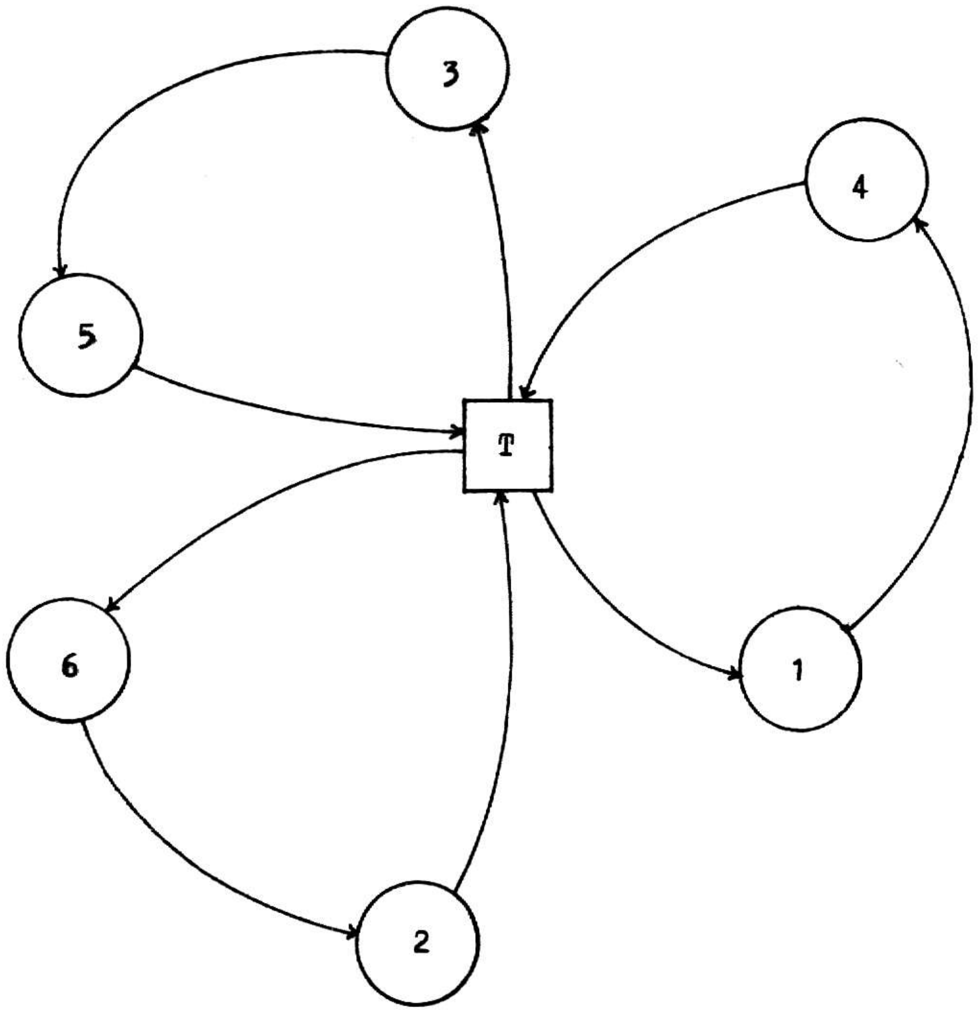


Figure 2.1$_d$ Minimum Delivery Route

Step 8. Find the optimal solution. The best solution can be found by choosing the one with a minimum total cost. As in Figure 2.1_d,

	Cost	Demand	Carrier	Capacity
$T \rightarrow 6 \rightarrow 2 \rightarrow T$	24	20	1	25
$T \rightarrow 1 \rightarrow 4 \rightarrow T$	19	20	1	25
$T \rightarrow 3 \rightarrow 5 \rightarrow T$	21	20	1	25
Total	64	60	3	

2.3 Computational Algorithm

In summary, the computational algorithm is stated step by step as follows:

Step 1 Construct the initial minimum 1-arborescence.

1.1 Set level $L = 0$.

1.2 construct a minimum spanning arborescence over demand points $D_2, D_3, \ldots, D_n$ by choosing n-2 links

1.3 Find the last two links associated with node (1) in the minimum 1-arborescence by choosing two edges with a minimum sum of $d(1, J) + d(I, 1)$. $I, J = 2, \ldots, n$ and the front-end-of-edge rule is satisfied.

1.4 Compute the corresponding lower bound on the tour cost such that

$$C^L = \sum d(k,\ell), \qquad (k,\ell) \in A^L$$

Step 2 Identify the out-of-kilter nodes

2.1 If there is only one minimum 1-arborescence identify all out-of-kilter nodes and go to step 3.

2.2 If there are more than one minimum 1-arborescence identify out-of-kilter nodes on each minimum 1-arborescence and go to step 3.

Step 3 Compute the cost increment.

3.1 If node (k) is out-of-kilter low, compute the increment such that

$$\delta^L(k) = \min_{(k,\ell)\varepsilon A^L} [d(k,\ell) - \max d(r,s)],$$

where $(k) \neq (r)$, (s) and $\{(r,s)\}$ is a set of candidate links.

3.2 If node (k) is out-of-kilter high, compute the increment such that

$$\delta^L(k) = \min_{(k,\ell)\varepsilon A^L} [\min (d(r,s)) - d(k,\ell)],$$

where $\{(r,s)\}$ is a set of candidate links.

Step 4 Select a weight substitute links.

4.1 Find the minimum increment substitute such that

$$\delta^*(k) = \min_k [\delta^L(k)].$$

4.2 Make a substitute by a link associated with $\delta^*(k)$. If there is a tie, select any by random.

4.3 Set level $L = L + 1$ and update the lower bound on the cost of the minimum 1-arborescence such that

$$C^L = C^{L+1} + \delta^*(k).$$

Step 5 Search for the minimum tour.

5.1 If all nodes in any minimum 1-arborescence are in-kilter, identify the tour and the corresponding cost such that $C^* = C^L$. Then go to step 6.

5.2 If no one minimum 1-arborescence has all of its nodes in-kilter, go to step 2.

Step 6 Find the initial subroute about the depot.

6.1 compute the cost of all possible subroutes including one demand point only and arrange it in ascending order.

6.2 Assign a carrier with maximum capacity to the first subroute S_1.

Step 7 Find the feasible solution.

7.1 Insert a set {M} which is a set of sequenced demand points associated with the optimal tour and check the resource constraints.

7.2 If total demand in a route is larger than the carrier's capacity, assign another carrier to the second initial route and go to step 7.1 until all demand points have been assigned.

Step 8 Find the optimal solution.

8.1 Compute the total cost of each set of the feasible solutions.

8.2 Find the minimum travelled cost.

CHAPTER III

NON-SYMMETRIC MULTI-DEPOT CARRIER-DISPATCHING PROBLEMS

The non-symmetric multi-depot carrier-dispatching problem can be considered as a generalization of the single-depot problem. The main difference between these two problems is that there are more than one depot in the multi-depot case. Hence, it can be stated as follows: determine the best delivery or pick-up routes about the depots covering a set of destinations (demand points) such that the total traveling distance, cost, or time is minimized. There are some required conditions: (1) the cost matrix is non-symmetric, (2) the number of available carriers and the carrying capacity are known, (3) the demand of each demand point is known and must be fulfilled, (4) every demand point can only appear once on the routes, and (5) the routes to be determined out will be either "delivery" or "pick-up" and not both or a mix of these two. The proposed research is to develop a practical algorithm for the non-symmetric multi-depot problems. A sample problem and a formal algorithm are given in this chapter.

3.1 Basic Concepts

Since the multi-depot carrier-dispatching problem is a generalization of the single-depot case, the proposed algorithm is based on the one shown in chapter II.

There are three major procedures concerned with this proposed algorithm. In phase 1, an optimal tour covering all the demand points but the depots is formed by the minimum 1-arborescence algorithm. In phase 2, several routes including the depots are constructed by the resulted tour. In phase 3, the feasible solution is improved by a "trial-and-error" method.

3.2 Sample Problem

The proposed algorithm for the non-symmetric multi-depot carrier-dispatching problem is demonstrated by a sample problem consisting of 10 demand points and 2 depots. The cost matrix and resource constraint are given in Tables 3.1, through 3.3. The objective is to find a set of routes such that the total traveling cost is minimized. The solution is formed step by step as follows.

Note that steps 1 through 5 in the proposed algorithm of Chapter II are used to construct an optimal tour covering the 10 demand points from which a minimum spanning 1-arborescence is constructed. The summary of these steps is displayed in Table 3.4. The optimal tour is

$$1 \rightarrow 9 \rightarrow 5 \rightarrow 6 \rightarrow 4 \rightarrow 7 \rightarrow 10 \rightarrow 2 \rightarrow 3 \rightarrow 6 \rightarrow 1,$$

and the total traveling cost is 33.

Step 6. Find the initial subtour for every depot. Calculate the cost of all possible routes for each depot and arrange it in ascending order. In our sample problem all initial routes are given such that

$$S_i = (i_{T_t}, i_k, i_{T_t}) \qquad \text{where } k = 1,n$$
$$i = 1,n$$
$$t = 1,m$$

or

Table 3.1 Cost Matrix of Sample Problem

From To	1	2	3	4	5	6	7	8	9	10
1	-	51	55	90	41	63	77	69	0	23
2	50	-	0	69	8	53	0	46	73	72
3	30	77	-	21	25	51	47	16	0	60
4	65	0	6	-	2	9	17	5	26	42
5	0	94	0	5	-	0	41	31	59	48
6	79	65	0	0	15	-	17	47	32	48
7	76	96	48	27	34	0	-	0	25	0
8	0	17	9	27	46	15	84	-	0	24
9	56	7	45	39	0	93	67	79	-	38
10	30	0	42	56	49	77	76	49	23	-

Table 3.2 Cost from the Demand Points to the Depots, and the Demand of each Point.

	T_1		T_2		Demand
1*	52	51	30	40	20
2	53	52	80	60	15
3	50	48	60	70	8
4	70	68	50	65	16
5	5	10	40	45	7
6	75	70	51	35	9
7	57	60	41	55	17
8	60	50	45	31	10
9	40	51	35	40	14
10	30	55	25	51	5

*An illustration, cost from demand point 1 to depot T_1 is 52 and from depot T_1 to demand point 1 is 51.

Table 3.3 Truck Allocation

Available Carrier	2	2	2	1
Capacity	50	40	30	20

Table 3.4 Summary of Solution to the Sample Problem

Level L	Candidate Arcs	Arcs Comprising the Min. 1-Arborescence	Out-of-Kilter Node High		Out-of-Kilter Node Low		Minimum Cost Increment	Selected Node	Lower Bound
			(k)	$\delta^L(k)$	(k)	$\delta^L(k)$	$\delta^*(k)$	(k^*)	
0	(3,8)	(1,7),(8,1),(9,5) (5,6),(6,4),(10,2) (2,3),(7,10),(7,8) (2,7)			3	18	18	3	0
1	(4,7)	(1,9),(8,1),(9,5) (6,4),(10,2),(2,3) (3,8),(2,7),(7,10) (5,6)			10	17	10	17	16
2		(1,4),(9,5),(5,6) (6,4),(4,7),(7,10) (10,2),(2,3),(3,8) (8,1)							33*

*A tour is obtained at this stage.

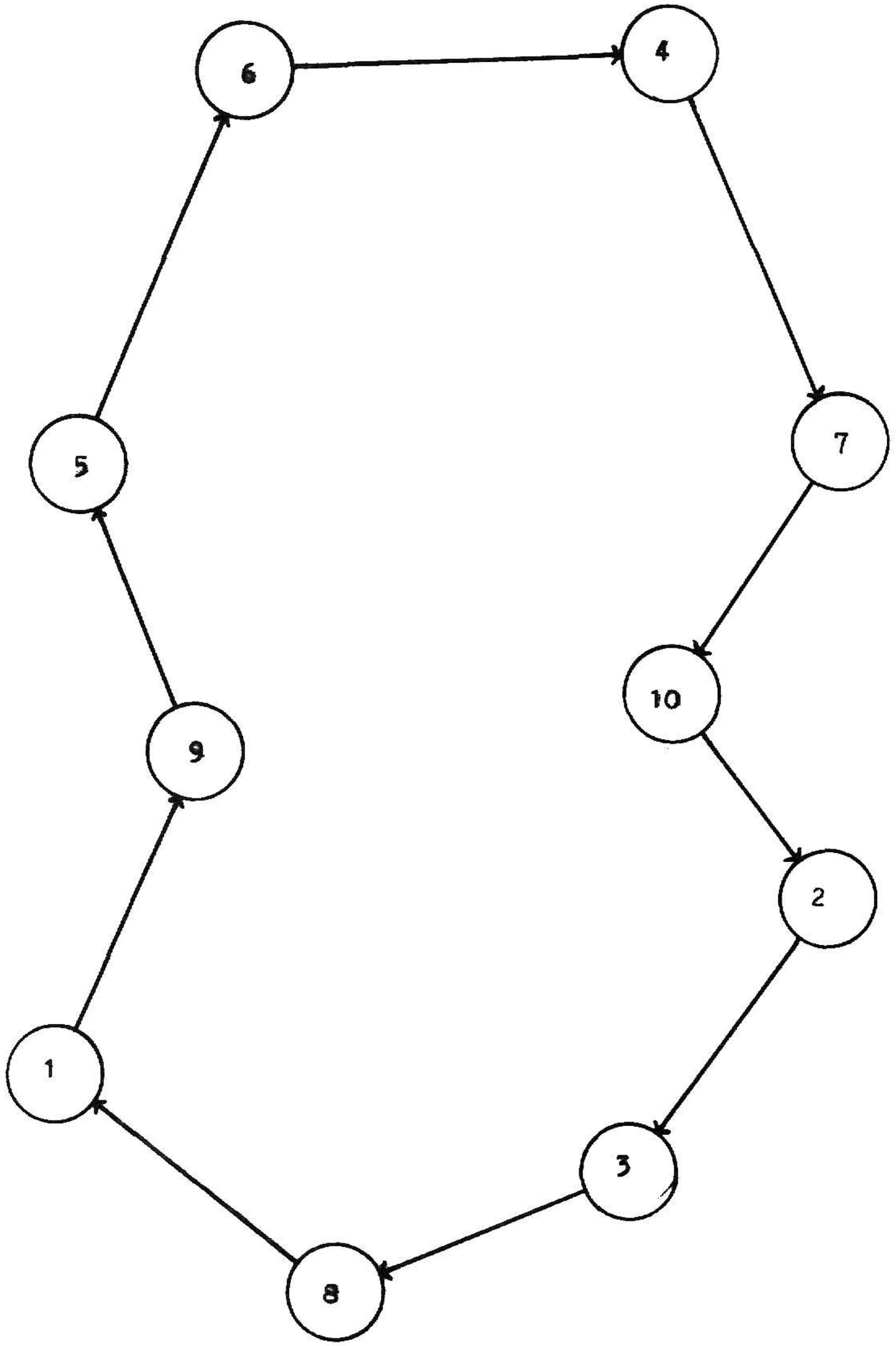


Figure 3.1 A Minimum Tour of the Sample Problem 2

$S_1 = (i_{T_1}, 5, i_{T_1})$	$d(S_1) = 15$
$S_2 = (i_{T_1}, 10, i_{T_1})$	$d(S_2) = 65$
$S_3 = (i_{T_2}, 1, i_{T_2})$	$d(S_3) = 70$
$S_4 = (i_{T_2}, 10, i_{T_2})$	$d(S_4) = 73$
$S_5 = (i_{T_2}, 9, i_{T_2})$	$d(S_5) = 75$
$S_6 = (i_{T_2}, 8, i_{T_2})$	$d(S_6) = 76$
$S_7 = (i_{T_2}, 5, i_{T_2})$	$d(S_7) = 85$
$S_8 = (i_{T_2}, 6, i_{T_2})$	$d(S_8) = 86$
$S_9 = (i_{T_2}, 9, i_{T_2})$	$d(S_9) = 91$
$S_{10} = (i_{T_1}, 3, i_{T_1})$	$d(S_{10}) = 98$
$S_{11} = (i_{T_1}, 1, i_{T_1})$	$d(S_{11}) = 103$
$S_{12} = (i_{T_1}, 2, i_{T_1})$	$d(S_{12}) = 105$
$S_{13} = (i_{T_1}, 8, i_{T_1})$	$d(S_{13}) = 110$
$S_{14} = (i_{T_2}, 4, i_{T_2})$	$d(S_{14}) = 115$
$S_{15} = (i_{T_2}, 7, i_{T_2})$	$d(S_{15}) = 126$

$S_{16} = (i_{T_2}, 3, i_{T_2})$ $\quad d(S_{16}) = 130$

$S_{17} = (i_{T_1}, 4, i_{T_1})$ $\quad d(S_{17}) = 138$

$S_{18} = (i_{T_2}, 2, i_{T_2})$ $\quad d(S_{18}) = 140$

$S_{19} = (i_{T_1}, 6, i_{T_1})$ $\quad d(S_{19}) = 145$

$S_{20} = (i_{T_1}, 7, i_{T_1})$ $\quad d(S_{20}) = 147$.

Then a carrier which has the largest capacity is assigned to the first route S_1.

Step 7. Find the feasible solution for all depots by the position of demand points in the resulted tour.

Expand the subroute S_1 by adding some demand points one by one according to the related position in the resulted tour about node (5). Each added node should satisfy the capacity of the carrier; otherwise, another new subroute has to be set up by S_2. The capacity of the carrier is checked with the total demand in one route till all demand points have been covered in the routes. In our sample problem it can be stated as follows:

$$\begin{cases} T_1 - 5 - 6 - 4 - 7 - T_1 \\ T_1 - 10 - 2 - 3 - 8 - 1 - T_1 \\ T_2 - 9 - T_3 \end{cases}$$

$$\left\{\begin{array}{l} T_1 - 9 - 5 - 6 - 4 - T_1 \\ T_1 - 10 - 2 - 3 - 8 - 1 - T_1 \\ T_2 - 7 - T_2 \end{array}\right.$$

$$\left\{\begin{array}{l} T_1 - 1 - 9 - 5 - 6 - T_1 \\ T - 10 - 2 - 3 - 8 - T_1 \\ T_2 - 4 - 7 - T_2 \end{array}\right.$$

$$\left\{\begin{array}{l} T_1 - 7 - 10 - 2 - 3 - T_1 \\ T_2 - 8 - 1 - 9 - T_2 \\ T_1 - 5 - 6 - 4 - T_1 \end{array}\right.$$

Step 8. Improve the feasible solutions. Reassign the demand points in the routes to different depots if the new assignment can have better solutions. In our sample problem they can be improve as:

$$\left\{\begin{array}{l} T_1 - 5 - 6 - 4 - 7 - T_1 \\ T_2 - 10 - 2 - 3 - 8 - 1 - T_2 \\ T_2 - 9 - T_2 \end{array}\right.$$

$$\left\{\begin{array}{l} T_2 - 9 - 5 - 6 - 4 - T_2 \\ T_2 - 10 - 2 - 3 - 8 - 1 - T_2 \\ T_2 - 7 - T_2 \end{array}\right.$$

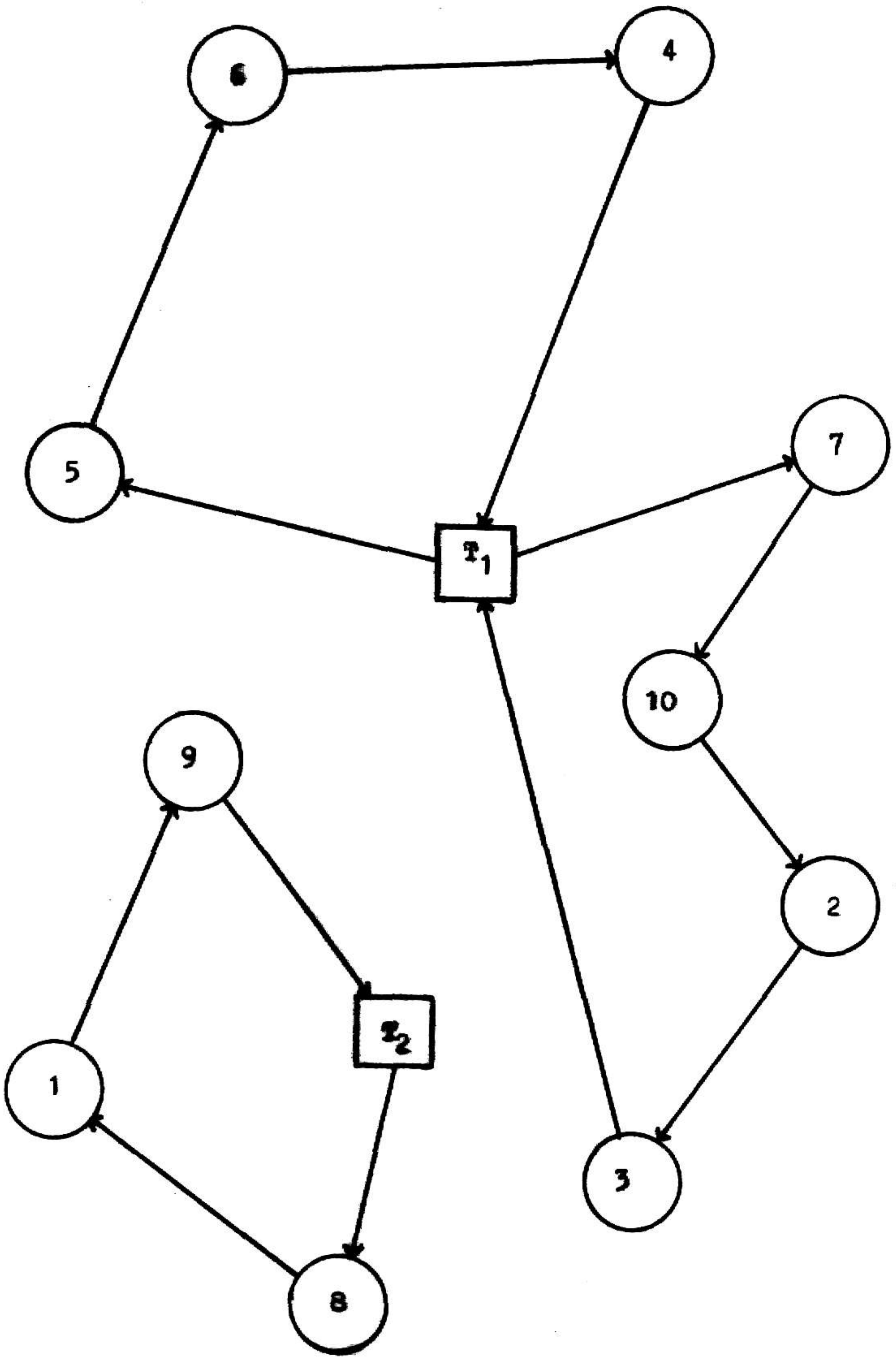


Figure 3.2 Minimal Routes of the Sample Problem 2

$$\left\{\begin{array}{l} T_2 - 1 - 9 - 5 - 6 - T_2 \\ T_2 - 10 - 2 - 3 - 8 - T_2 \\ T_2 - 4 - 7 - T_2 \end{array}\right.$$

$$\left\{\begin{array}{l} T_1 - 7 - 10 - 7 - 3 - T_1 \\ T_2 - 8 - 1 - 9 - T_2 \\ T_1 - 5 - 6 - 4 - T_1 \end{array}\right.$$

Compute the total cost of each set of solutions, the minimum one is

	Cost	Demand	Carrier	Capacity
$T_1 - 7 - 10 - 2 - 3 - T_1$	110	45	1	50
$T_2 - 8 - 1 - 9 - T_2$	66	44	1	50
$T_1 - 5 - 6 - 4 - T_1$	80	32	1	40
Total	256	121	3	

Figure 3.2 depicts the best routes which we found.

3.3 Computational Algorithm

Step 1 Construct the initial minimum 1-arborescence.

1.1 Set level L = 0

1.2 construct a minimum spanning arborescence over demand points D_2, D_3, ..., D_n by choosing n-2 links.

1.3 Find the last two links associated with node (1) in the minimum 1-arborescence by chosing two edges with a minimum sum of $d(1, J) + d(I,1)$, $I, J = 2, \ldots, n$, and the front-end-of-edge rule is satisfied.

1.4 Compute the corresponding lower bound on the tour cost such that

$$C^L = \sum_{(k,\ell)} d(k,\ell), \qquad (k,\ell) \in A^L$$

Step 2 Identify the out-of-kilter nodes.

2.1 If there is only one minimum 1-arborescence identify all out-of-kilter nodes and go to step 3.

2.2 If there are more than one minimum 1-arborescence identify out-of-kilter nodes on each minimum 1-arborescence and go to step 3.

Step 3 Compute the cost increment.

3.1 If node (k) is out-of-kilter low, compute the increment such that

$$\delta^L(k) = \min_{(k,\ell)\in A^L} [d(k,\ell) - \max (r,s)],$$

where $(k) \neq (r)$, (s) and $\{(r,s)\}$ is a set of candidate links.

3.2 If node (k) is out-of-kilter high, compute the increment such that

$$\delta^L(k) = \min_{(k,\ell)\in A^L} [\min (d(r,s)) - d(k,\ell)],$$

where $\{(r,s)\}$ is a set of candidate links.

Step 4 Select a weight substitute links.

4.1 Find the minimum increment substitute

$$\delta^{*}(k) = \min_{k} [\delta^{L}(k)].$$

4.2 Make a substitute by a link associated with $\delta^{*}(k)$. If there is a tie, select any by random.

4.3 Set level L = L + 1 and update the lower bound on the cost of the *minimum* 1-arborescence such that

$$C^{L} = C^{L+1} + \delta^{*}(k).$$

Step 5 Search for the minimum tour.

5.1 If all nodes in any minimum 1-arborescence are in kilter, identify the tour and the corresponding cost such that $C^{*} = C$. Then go to step 6.

5.2 If no one minimum 1-arborescence has all of its nodes in-kilter, go to step 2.

Step 6 Find the initial subroutes for every depot.

6.1 Compute the cost of all possible routes for all depots which including only one demand point and arrange them as an ascent order.

6.2 Assign a carrier with largest capacity to the first route S_1.

Step 7 Find the feasible solution for all depots.

7.1 Construct a route by inserting a number of demand points which associated to the first subroute S_1 one by one. Check the total demand $\sum d$ in the route with the carrier capacity q, if $\sum d<q$ repeat this step.

7.2 Assign the second carrier with second largest capacity to the second subroute S_2, go back to Step 7.1.

7.3 Check if all demand points are in the subroutes, if not, go to Step 7.2.

Step 8 Improve the feasible solutions to the best one.

8.1 According to the feasible solutions, make the change of depots of the routes if new depot will give a better solution.

CHAPTER IV

A GEOMETRICAL SEARCH TO THE SAVINGS APPROACH

Large size carrier-dispatching problems can be solved by Clarke and Wrights' savings approach. But it must be preceded by the computation and searching of a large matrix of the saving values. All potentially significant links between destinations or demand points are considered as candidates. Consequently, the computational time involved is considerably high. In order to reduce the computer time, however, the search from a large savings file can be eliminated by using a geometrical search technique on an order list of the polar co-ordinates of the destinations or demand points.

A proposed algorithm based on Yellow's ideas [18] is developed for solving a symmetric and non-symmetric single-depot carrier-dispatching problems. The multi-depot problem can be solved with suitable modifications.

4.1 Basic Concepts:

Consider a general saving value s_{ij} given by linking demand points i and j such that

$$s_{ij} = d_{0i} + d_{0j} - d_{ij}; \qquad (1)$$

where

s_{ij} = saving value,

d_{0i} = distance from demand point i to depot,

d_{ij} = distance between demand points i and j.

This formula can be transferred to polar co-ordinates related to the depot such that

$$s_{ij} = r_i + r_j - \sqrt{r_i^2 + r_j^2 - 2r_ir_j\cos(\theta_i-\theta_j)} \tag{2}$$

where (r_i,θ_i) is the polar co-ordinate of a point i. From equation (2)

$$(r_i + r_j - s_{ij})^2 = (r_i^2 + r_j^2 - 2r_ir_j\cos(\theta_i-\theta_j))$$

$$2r_ir_j-2S_{ij}(r_i+r_j) + s_{ij}^2 - -2r_ir_j\cos(\theta_i-\theta_j)$$

$$2r_kr_j(\cos(\theta_i-\theta_j) + 1) - 2s_{ij}(r_i+r_j) + s_{ij}^2 = 0 \tag{3}$$

let (R,ϕ) be the polar co-ordinate of a point on the locue. From (3) it can be rewritten as:

$$2Rr_i[\cos\ (\theta_i - \phi) + 1] - 2s\ (R+r_i) + s^2 = 0 \tag{4}$$

Equation (4) shows that the locus of points give equal saving values when demand point i is fixed, and the loci are hyperbolas. If

$$r_j > \frac{s^2-2sr_i}{2s-2r_i[1+\cos(\theta_j-\theta_i)]}\ . \tag{5}$$

where

$(r_i,\ \theta_i)$ = a fixed demand point i,

(r_j,θ_j) = a demand point j linked with demand point i.

Then the link (i,j) has a higher savings value. From inequality (5), several candidate links can be found and with those links can form some routes to solve the carrier-dispatching problem.

4.2 Symmetric Sample Problem

The proposed algorithm for the symmetrical single-depot carrier-dispatching problems is illustrated by a sample problem with 6 demand points, the distance matrix and truck allocation are shown in Tables 4.1 and 4.2. Note that any trip by a truck should not exceed 160 miles.

Step 1. Transfer the rectangular co-ordinates of the demand points to polar co-ordinates using the formula:

$$r_{ij} = d_{ij}$$

and

$$\theta_{ij} = \cos^{-1} \frac{d_{0i}^2 + d_{0j}^2 - d_{ij}^2}{2d_{0i}d_{0j}}$$

where $\theta_{ij} = \theta_i - \theta_j$

or

$\theta_{12} = 0.90$

$\theta_{13} = 0.27$ $\theta_{23} = 0.62$

$\theta_{14} = 1.02$ $\theta_{24} = 0.12$ $\theta_{34} = 0.76$

$\theta_{15} = 0.56$ $\theta_{25} = 0.30$ $\theta_{35} = 0.40$ $\theta_{45} = 0.46$

$\theta_{16} = 0.85$ $\theta_{26} = 0.00$ $\theta_{36} = 0.93$ $\theta_{46} = 0.19$ $\theta_{56} = 0.59$

Step 2. Compute the savings by

Table 4.1 Distance Matrix

Q	p_0						
7	38	p_1					
9	42	35	p_2				
8	56	22	33	p_3			
7	63	54	22	45	p_4		
6	64	38	27	20	29	p_5	
5	80	62	38	65	22	45	p_6

Table 4.2 Truck Allocation

Capacity	12	15	18
Available	2	1	1

$$s_{ij} = r_i + r_j - \sqrt{r_i^2 + r_j^2 - 2r_i r_j \cos(\theta_i - \theta_j)}$$

or

$s_{12} = 45$

$s_{13} = 72$ $\quad s_{23} = 65$

$s_{14} = 47$ $\quad s_{24} = 83$ $\quad s_{34} = 74$

$s_{15} = 64$ $\quad s_{25} = 79$ $\quad s_{35} = 100$ $\quad s_{45} = 98$

$s_{16} = 56$ $\quad s_{26} = 84$ $\quad s_{36} = 71$ $\quad s_{46} = 121$ $\quad s_{56} = 99$

Step 3. Find candidate links with higher savings. Compute the value

$$V_{ij} = r_i - \frac{s^2 - 2sr_i}{2s - 2r_j[1 + \cos(\theta_i - \theta_i)]},$$

and consider only the value V_{ij} which is greater than zero. In our sample problem V_{52}, V_{62}, V_{64} are greater than 0, so links (5,2), (6,2), (6,4) are the candidates.

Step 4. Find the feasible solution with candidate links. Make the right combination of the candidate links which found from step 3, form all possible routes, and assign a carrier to each route. Check the capacity of the carrier with the total demand in the routes and the traveled distance of any route should not be over 160 miles. The feasible solutions are as follows:

Solution 1	Distance	Solution 2	Distance
T - 6 - 2 - T	160	T - 5 - 2 - T	133
T - 1 - T	76	T - 1 - T	76
T - 3 - T	112	T - 3 - T	112
T - 4 - T	126	T - 4 - T	126
T - 5 - T	128	T - 6 - T	160
Total	602	Total	607

Step 5. Find the solution with minimum total traveled distance. Improve the feasible solutions by combining some routes into one and which will not conflict the constraints. Calculate the total traveled distance for each set of solutions. The minimum one is then the best solution. In our sample problem, the following solutions are:

Solution 1(a)	Distance	Truck Capacity	Truck No.
T - 6 - 2 - T	160	15	1
T - 1 - 3 - T	116	18	1
T - 4 - T	126	12	1
T - 5 - T	128	12	1
Total	830		4

Solution 1(b)	Distance	Truck Capacity	Truck No.
T - 6 - 2 - T	160	18	1
T - 4 - 5 - T	156	15	1
T - 1 - T	76	12	1
T - 3 - T	112	12	1
Total	504		4

*Solution 2	Distance	Truck Capacity	Truck No.
T - 5 - 2 - T	133	18	1
T - 1 - 3 - T	116	15	1
T - 4 - T	126	12	1
T - 6 - T	128	12	1
Total	503		4

The best solution is then solution 2.

4.3 Non-symmetric Sample Problem

The proposed algorithm is also applicable for non-symmetric case. For example, consider 4 demand points. The associated distance and truck allocation are shown in Tables 4.3 through 4.5.

Step 1. Transfer the rectangular co-ordinates of the demand points to polar co-ordinates such that

$$r_{ij} = d_{ij},$$

and

$$\theta_{ij} = \cos^{-1} \frac{d_{i0}^2 + d_{0j}^2 - d_{ij}^2}{2d_{i0}d_{0j}}$$

or

$$\theta_{12} = 0.403 \qquad \theta_{21} = 0.505$$

$$\theta_{13} = 0.505 \qquad \theta_{31} = 0.504$$

$$\theta_{14} = 0.128 \qquad \theta_{41} = 0.141$$

$$\theta_{23} = 0.200 \qquad \theta_{32} = 0.100$$

$$\theta_{24} = 0.715 \qquad \theta_{42} = 0.609$$

$$\theta_{34} = 0.609 \qquad \theta_{43} = 0.505$$

Step 2. Compute the savings such that

$$s_{ij} = r_{i0} + r_{0j} - \sqrt{r_{i0}^2 + r_{0j}^2 - 2r_{i0}\, r_{0j}\, (\theta_i - \theta_j)} \qquad (6)$$

where

r_{i0}	distance from point i to depot 0,
r_{0j}	distance from depot 0 to point j,

or

$s_{12} = 160$	$s_{21} = 150$
$s_{13} = 150$	$s_{31} = 140$
$s_{14} = 80$	$s_{41} = 70$
$s_{23} = 180$	$s_{32} = 190$
$s_{24} = 130$	$s_{42} = 140$
$s_{34} = 140$	$s_{43} = 150.$

Step 3. Find candidate links with higher saving. Compute the value

$$V_{ij} = t_{i0} - \frac{s^2 - 2s\, r_{i0}}{2s - 2r_{i0}\,[1 + \cos\,(\theta_j - \theta_i)]}\,,$$

and consider only the value(s) which is greater than zero. In this sample problem, V_{12}, V_{21}, V_{13}, V_{24}, V_{34}, V_{42}, and V_{43} are greater than zero. Thus, links (1,2), (2,1), (2,4), (3,4), and (4,2) are candidates

Step 4. Determine the feasible solution.

<u>Alternative 1:</u> $T \rightarrow 2 \rightarrow 1 \rightarrow T$ and $T \rightarrow 3 \rightarrow 4 \rightarrow T$.

Table 4.3 Distance Matrix

From To	T	1	2	3	4
T	-	100	100	100	100
1	100	-	40	50	120
2	100	50	-	20	70
3	100	60	10	-	60
4	100	180	60	50	-

Table 4.4 Demand of Demand Point

Demand point	1	2	3	4
Demand	6	6	4	4

Table 4.5 Truck Allocation

Capacity	15	10
Available	1	1

<u>Alternative 2</u>: T → 2 → 1 → T and T → 1 → 3 → T

<u>Alternative 3</u>: T → 1 → 2 → T and T → 3 → 4 → T

Step 5. Find a best solution. In our sample problem the best solution of the feasible solution is:

	Cost	Demand	Capacity	Carrier
T → 1 → 2 → T	240	12	15	1
T → 3 → 4 → T	260	8	10	1
Total	500	20		2

4.4 Computational Algorithm

The algorithm discussed above is stated below in formal steps:

Step 1. Construct polar co-ordinates of demand points

1.1 Transform the rectangular co-ordinates of the demand points to the polar co-ordinates by

$$r_{ij} = d_{ij},$$

and

$$\theta_i - \theta_j = \cos^{-1}\left(\frac{d_{0i}^2 + d_{0j}^2 - d_{ij}^2}{2d_{0i}\, d_{0j}}\right)$$

Step 2. Compute the savings such that

$$s_{ij} = r_i + r_j - \sqrt{r_i^2 + r_j^2 - 2r_i r_j \cos(\theta_i - \theta_j)}$$

Step 3. Find candidate links with higher savings.

3.1 Compute

$$V_{ij} = r_i - \frac{s_{ij}^2 - 2s_{ij}r_i}{2s_{ij} - 2r_i\,[1 + \cos(\theta_i - \theta_i)]}$$

3.2 If $V_{ij} > 0$ then the link (i,j) indicates the candidate link i-j.

Step 4. Determine feasible solutions.

4.1 Combine the candidate links into several routes and assign a truck to each route.

4.2 Check for conflict. If there is any conflict, another combination of the routes is selected.

4.3 Check the capacity of the carrier with the total demand in the routes. If $\sum d_i > q$, then assign another carrier

Step 5. Find a best solution.

5.1 Improve the feasible solution by combining some routes into one.

5.2 Check the constraints and the capacity of carrier with total demand in the new routes. If there is a conflict, go to step 4.

5.3 Compute the total traveled distance for each set of feasible solutions and find the one with the least distance.

Note that in non-symmetric case, d_{0i} is replaced by d_{i0}, r_i by r_{i0}, and r_j by r_{0j}.

CHAPTER V

SUMMARY AND CONCLUSIONS

The carrier-dispatching problem is one of the most frequently occurring real-world problems in the transportation area. This problem may be defined as: finding a set of delivery or pick-up routes, in which consists of a number of destinations or demand points to be covered by a number of carriers. It is required to determine a set of delivery or pick-up routes which satisfy the imposed constraints and such that the total traveling cost is minimized. The problem under consideration has deterministic demands and non-symmetric cost matrix.

The carrier-dispatching problem may be regarded as a generalization of the classical traveling salesman problem. The major methods proposed for solving the single-depot problems are branch-and-bound (Hayes, 1967), simulation (Braun, 1967), integer programming (Balinski and Quandt, 1964), dynamic programming (Held and Karp, 1962) and heuristic programming (Dantzig and Ramser, 1959; Clarke and Wright, 1964; Hering, 1970). The methods available for multi-depot problems are heuristics (Tillman, 1969 and Hering, 1970). By examining these approaches, it appears that the heuristic programming approach is the most practical procedure at present, because of their computational feasibility. With this in mind, a minimum 1-arborescence algorithm for solving both single- and multi-depot carrier-dispatching problems has been developed.

The proposed algorithm for solving multi-depot carrier-dispatching problems consists of three main phases. In phase I, an optimal tour covering all the demand points but the depots is formed by the minimum 1-arborescence algorithm. In phase II, several routes including the depots are constructed by the resulted tour. In phase III, the feasible solution is improved by a "trial-and-error" method.

A geometrical search of savings approach has been developed as a criteria to choose the links for both symmetric and non-symmetric cases.

In this research, four test problems have been solved by the proposed methods: (1) single-depot carrier-dispatching method, (2) multi-depot carrier-dispatching method, and (3) geometrical search of savings approach. And the results are shown in Table 5.1. No comparison with other existing procedures is made because no non-symmetric sample problems appear in the literature.

The proposed approach looks promising because of the relative success obtained, specially, for small and medium size problems.

Table 5.1 Summary of Results

Problem No.	No. of Demand Points	No. of Depots	Symmetric or Non-sym.	Total Cost
1	4	1	non-sym.	510
2	5	1	non-sym.	145
3	6	1	non-sym.	163
4	10	1	non-sym.	270
5	4	2	non-sym.	36
6	5	2	non-sym.	51
7	6	2	non-sym.	148
8	10	2	non-sym.	256
9*	4	1	non-sym.	500
10*	6	1	non-sym.	151
11*	5	1	sym.	281
12*	10	1	sym.	104

*Problem solved by geometrical search of savings approach

REFERENCES

[1] Balinski, M.L. and Quandt, R.E., "On an Integer Program for a Delivery Problem, "Operations Research, vol. 12, 1964, pp. 300-304.

[2] Braun, W., "A Computerized Simulation Approach to the Solution of the Carrier-dispatching Problem," Master's Thesis, Kansas State University, 1967.

[3] Clarke, G. and Wright, J. W., "Scheduling of Vehicles from a Central Depot to a Number of Delivery points, "Operations Research, vol. 12, no. 4, 1964, pp. 568-581.

[4] Cochran, H., "Optimization of a Carrier Routing Problem," Master's Thesis, Kansas State University, 1966.

[5] Dantzig, G. B. and Ramser, J. H., "The Truck Dispatching Problem," Management Science, vol. 6, no. 1, 1959, pp. 80-91.

[6] Gaskell, T. J., "Bases for Vehicle Fleet Scheduling," Operations Research Quarterly, vol. 18, no. 3, 1967, pp. 281-295.

[7] Gomory, R.E., "All-Integer Programming Algorithm" RC-189, IBM Research Center, January 29, 1960.

[8] Hausman, W. H. and Gilmour, P., "A Multi-period Truck Delivery Problem," Transportation Research, vol. 1, 1967, pp. 349-357.

[9] Hayes, R. L., "The Delivery Problem, "Doctoral Dissertation, Carnegie Institute of Technology, 1967.

[10] Hering, Robert "Evaluation of Some Heuristic Look Ahead Rules for Multiple Terminal Delivery Problems", M.S. Thesis, Kansas State University, Manhattan, Kansas (1970).

[11] Held, M. and R. M. Karp, "A Dynamic Programming Approach to Sequencing Problems, "Journal of the Society for Industrial and Applied Mathematics 10, 1962, pp. 196-210.

[12] Held, M. and R. M. Karp, "The Traveling-Salesman Problem and Minimum Spanning Trees," Operations Research, vol. 18, 1970, pp. 1138-1162.

[13] Little, J. D. C., Murty, K. C., Sweeney, D. W., and Karel, C., "An Algorithm for the Traveling Salesman Problem," Operations Research, vol. 11, 1963, pp. 972-989.

[14] Newton, R. M. and Thomas, W. H. "Design of School Bus Routes by Digital Computer," Paper presented at 13th National Meeting of ORSA, October, 1966.

[15] Tillman, F. A. and Cochran, H., "A Heuristic Approach for Solving the Delivery Problem, "Journal of Industrial Engineering, vol. 19, no. 7, 1968, pp. 354-358.

[16] Tyagi, M. S., "A Practical Method for Truck Dispatching Problems," Journal Operations Research Society of Japan, vol. 10, nos. 3 & 4, 1968, pp. 76-92.

[17] Quen, J., Charnes, A., and Werson, S., "Simulation and Analysis of a Refuse Collection System, "Journal of the Society Engineering Division ASCE, vol. 91, no. 3A5, Proc. Papper 4491, (1965).

[18] Yellow, P. C., "A Computational Modification to the Savings Method of Vehicle Scheduling, "Operational Research Quarterly, vol. 21, no. 2, 1970, pp. 281-283.

APPENDIX I

FLOW CHART OF ALGORITHM

Figure B The Flow Chart of the Single-Depot Algorithm

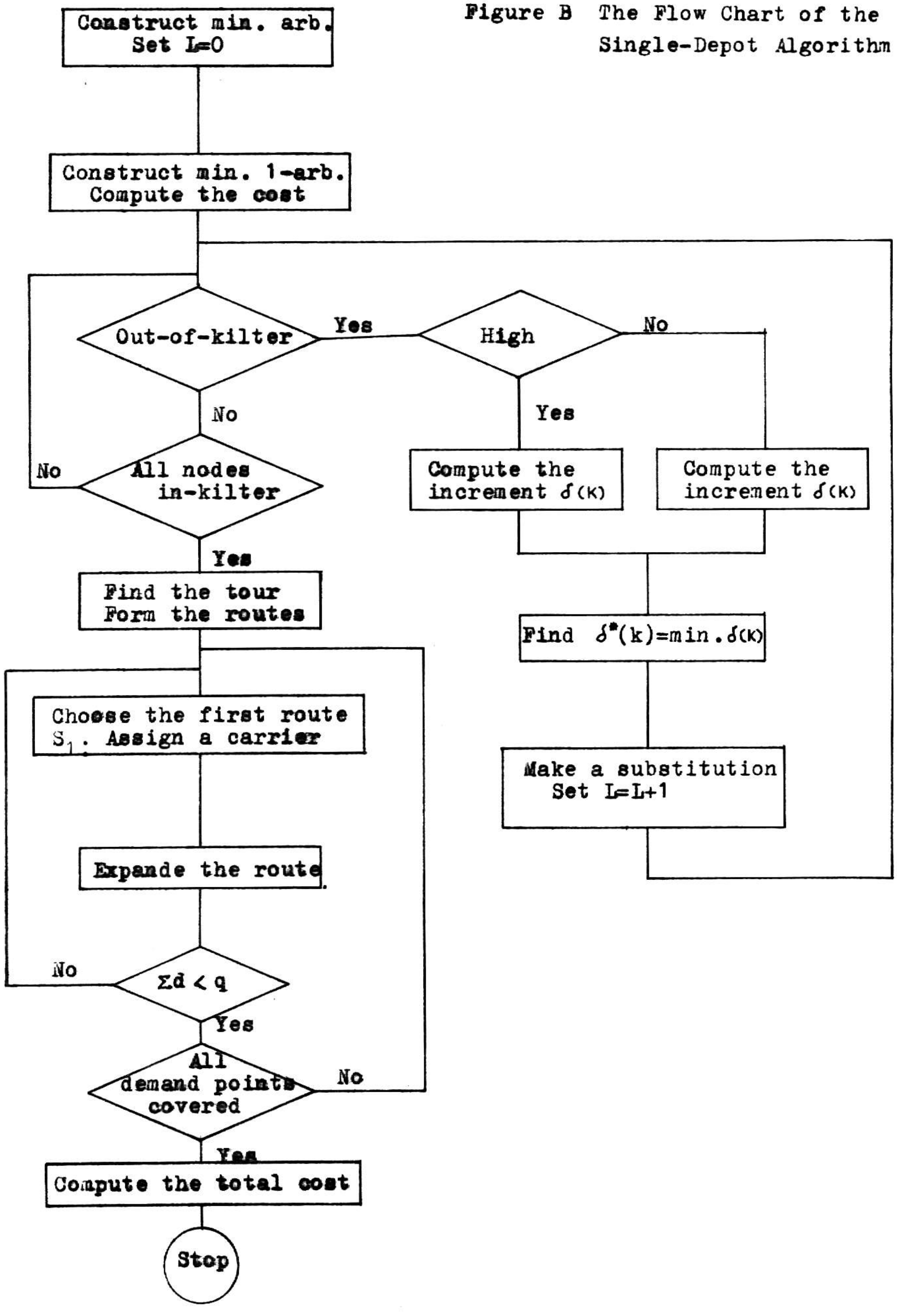
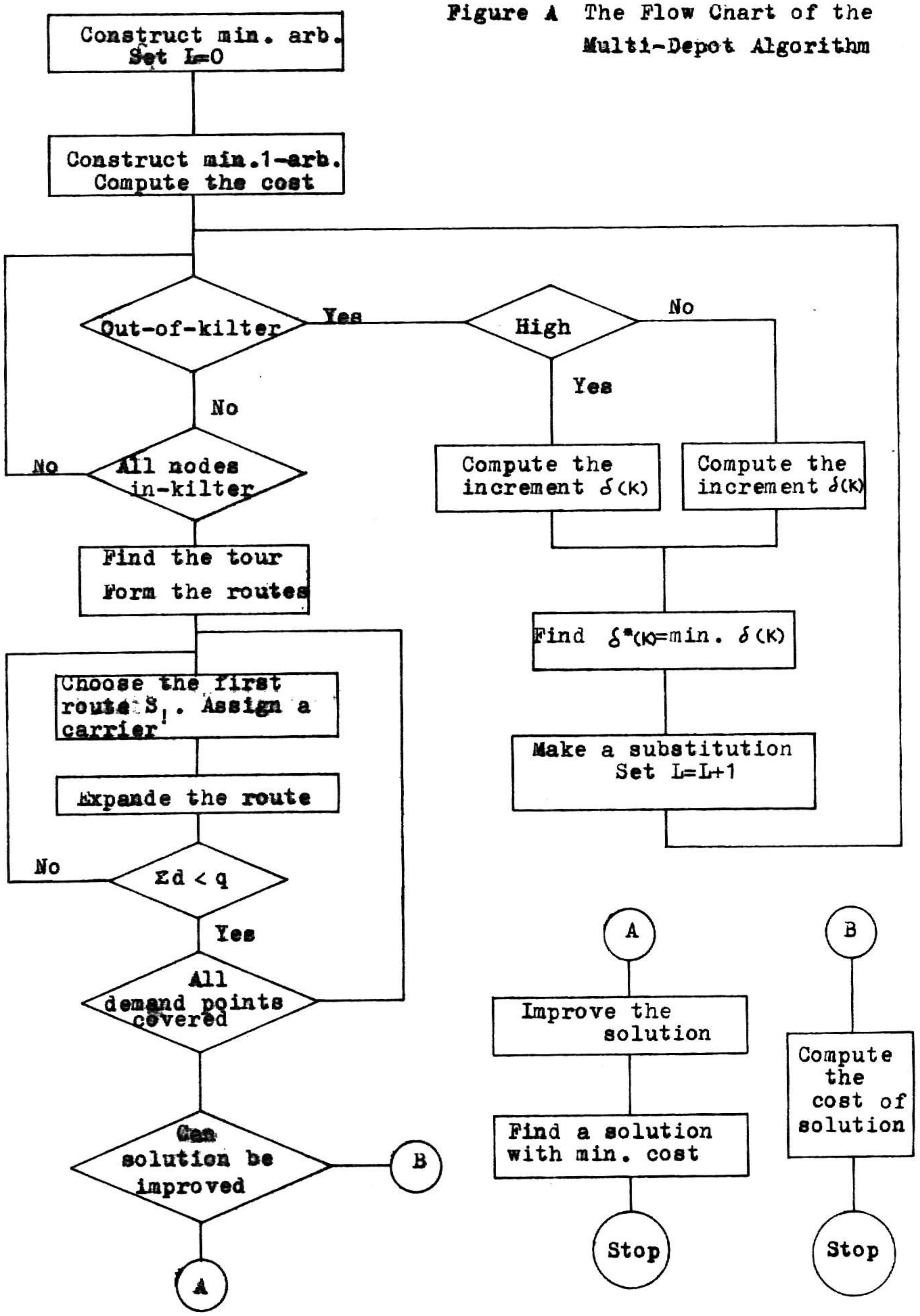

Construct min. arb. Set L=0
Construct min. 1-arb. Compute the cost
Out-of-kilter
Yes
High
No
Yes
Compute the increment $\delta(K)$
Compute the increment $\delta(K)$
Find $\delta^*(k)=\min.\delta(K)$
Make a substitution Set L=L+1
No
All nodes in-kilter
No
Yes
Find the tour Form the routes
Choose the first route S_1. Assign a carrier
Expande the route
No
$\Sigma d < q$
Yes
All demand points covered
No
Yes
Compute the total cost
Stop

Figure A The Flow Chart of the Multi-Depot Algorithm



Construct min. arb. Set L=0
Construct min.1-arb. Compute the cost
Out-of-kilter
Yes
High
No
Yes
No
No
All nodes in-kilter
Compute the increment δ(K)
Compute the increment δ(K)
Find the tour Form the routes
Find δ*(K)=min. δ(K)
Choose the first route S1. Assign a carrier
Make a substitution Set L=L+1
Expande the route
No
Σd < q
Yes
All demand points covered
A
B
Improve the solution
Can solution be improved
B
Compute the cost of solution
Find a solution with min. cost
Stop
Stop
A

APPENDIX II

CARRIER-DISPATCHING PROBLEMS

PROBLEM 1

1 DEPOT 4 DEMAND POINTS

DEMAND POINT	DEMAND AT THE POINT
1	6
2	6
3	4
4	4

CARRIERS CAPACITY	NUMBER
15	1
10	1

COST MATRIX

	T	1	2	3	4
T	-	100	100	100	100
1	100	-	40	50	120
2	100	50	-	20	70
3	100	60	10	-	60
4	100	180	60	50	-

FINAL SOLUTION

No.	DEPOT	ROUTES	LOAD	COST
1	1	T 1 3 4 T	14	310
2	1	T 2 T	6	200

TOTAL COST=510

CARRIER CAPACITY	15	10
AVAIBLE	1	1
ALLOCATED	1	1

PROBLEM 2

1 DEMAND 5 DEMAND POINTS

DEMAND POINT	DEMAND AT THE POINT
1	30
2	20
3	10
4	10
5	20

CARRIER CAPACITY	NUMBER
50	2
40	1
30	1

COST MATRIX

	T	1	2	3	4	5
T	-	30	28	40	35	20
1	28	-	10	25	25	10
2	27	1	-	10	15	2
3	30	8	9	-	20	10
4	30	14	10	24	-	15
5	25	10	8	25	27	-

FINAL SOLUTION

No.	DEPOT	ROUTES	LOAD	COST
1	1	T 5 2 3 T	50	68
2	1	T 4 1 T	40	77

TOTAL COST=145

CARRIER CAPACITY	50	40	30
AVAILABLE	2	1	1
ALLOCATED	1	1	0

PROBLEM 3

1 DEPOT 6 DEMAND POINTS

DEMAND POINT	DEMAND AT THE POINT
1	17
2	11
3	27
4	16
5	11
6	21

CARRIER CAPACITY	NUMBER
60	2
50	1
40	1

COST MATRIX

	T	1	2	3	4	5	6
T	-	40	20	30	15	30	30
1	38	-	27	43	16	30	26
2	40	7	-	16	1	30	25
3	40	20	13	-	35	5	6
4	20	21	16	25	-	18	18
5	15	12	46	27	48	-	5
6	20	23	5	5	9	5	-

FINAL SOLUTION

No.	DEPOT	ROUTES	LOAD	COST
1	1	T 2 1 4 T	44	63
2	1	T 3 5 T	39	50
3	1	T 6 T	21	50

TOTAL COST=163

CARRIER CAPACITY	60	50	40
AVAILABLE	2	1	1
ALLOCATED	1	1	1

PROBLEM 4

1 DEPOT 10 DEMAND POINTS

DEMAND POINT	DEMAND AT THE POINT
1	20
2	25
3	8
4	16
5	7
6	9
7	17
8	10
9	14
10	5

CAPACITY OF CARRIERS	NUMBER
60	2
50	2
30	2
20	1

COST MATRIX

	T	1	2	3	4	5	6	7	8	9	10
T	-	52	53	50	70	5	75	57	60	40	30
1	51	-	51	55	90	41	63	77	69	0	23
2	52	50	-	0	69	8	53	0	46	73	72
3	48	30	77	-	21	25	51	47	16	0	60
4	68	65	0	6	-	2	9	17	5	26	42
5	10	0	94	0	5	-	0	41	31	59	48
6	70	79	65	0	0	15	-	17	5	26	42
7	60	76	96	48	27	34	0	-	0	25	0
8	50	0	17	9	27	46	15	84	-	0	24
9	51	56	7	45	39	0	93	67	79	-	38
10	55	30	0	42	56	49	77	76	49	25	-

FINAL SOLUTION

No.	DEPOT	ROUTES	LOAD	COST
1	1	T 5 6 4 7 T	49	82
2	1	T 10 2 3 8 1 T	58	97
3	1	T 9 T	14	91

TOTAL COST = 270

CARRIER CAPACITY	60	50	30	20
AVAILABLE	2	2	2	1
ALLOCATED	1	1	0	1

PROBLEM 5

2 DEPOTS 4 DEMAND POINTS

DEMAND POINT	DEMAND AT THE POINT
1	18
2	12
3	16
4	14

CARRIER CAPACITY	NUMBER
40	3

COST MATRIX

	1	2	3	4
1	-	4	2	3
2	5	-	2	6
3	3	5	-	4
4	4	3	5	-

COST TO DEPOT

DEMAND POINT	T_1	T_2
1*	10 8	8 10
2	9 7	7 9
3	8 6	7 9
4	9 10	10 7

* An illustratation, cost from demand point 1 to depot T_1 is 8 and from depot T_1 to demand 1 is 10.

FINAL SOLUTION

No.	DEPOT	ROUTES	LOAD	COST
1	2	T_2 2 3 T_2	28	18
2	2	T_2 1 4 T_2	32	18

TOTAL COST= 36

CARRIER CAPACITY	40
AVAILABLE	3
ALLOCATED	2

PROBLEM 6

2 DEPOTS 5 DEMAND POINTS

DEMAND POINT	DEMAND AT THE POINT
1	7
2	8
3	8
4	7
5	6

CARRIER CAPACITY	NUMBER
18	1
15	1
12	1

COST MATRIX

	1	2	3	4	5
1	-	5	13	13	15
2	2	-	3	5	2
3	7	5	-	1	3
4	5	9	4	-	7
5	5	2	8	4	-

COST TO DEPOT

DEMAND POINT	T_1		T_2	
1	3	4	4	4
2	8	7	8	7
3	9	10	10	10
4	9	8	8	7
5	6	7	7	6

FINAL SOLUTION

NO.	DEPOT	ROUTES	LOAD	COST
1	1	T_1 3 5 T_1	14	19
2	2	T_2 4 1 T_2	14	17
3	1	T_1 2 T_1	8	15

TOTAL COST=51

CARRIER CAPACITY	18	15	12
AVAILABLE	1	1	1
ALLOCATED	1	1	1

PROBLEM 7

2 DEPOTS 6 DEMAND POINTS

DEMAND POINT	DEMAND AT THE POINT
1	17
2	11
3	27
4	16
5	11
6	21

CARRIER CAPACITY	NUMBER
60	2
50	1
40	1

COST MATRIX

	1	2	3	4	5	6
1	-	27	43	16	30	26
2	7	-	16	1	30	25
3	20	13	-	35	5	0
4	21	16	25	-	18	18
5	12	46	27	48	-	5
6	23	5	5	9	5	-

COST TO DEPOTS

DEMAND POINT	T_1		T_2	
1	40	38	35	35
2	20	10	20	15
3	30	40	25	30
4	15	20	15	15
5	30	15	25	10
6	30	20	35	25

FINAL SOLUTION

NO.	DEPOT	ROUTES	LOAD	COST
1	2	T_2 2 1 4 T_2	44	58
2	2	T_2 3 5 T_2	39	40
3	1	T_1 6 T_1	21	50

TOTAL COST=148

CARRIER CAPACITY	60	50	40
AVAILABLE	2	1	1
ALLOCATED	1	1	1

PROBLEM 8

2 DEPOTS 10 DEMAND POINTS

(see problem 4)

COST TO DEPOTS

DEMAND POINT	T_1		T_2	
1	52	51	30	40
2	53	52	80	60
3	50	48	60	70
4	70	68	50	65
5	5	10	40	45
6	75	70	51	35
7	57	60	41	55
8	40	50	45	31
9	40	51	35	40
10	30	55	25	51

FINAL SOLUTION

NO.	DEPOT	ROUTES	LOAD	COST
1	1	T_1 7 10 2 3 T_1	45	110
2	2	T_2 8 1 9 T_2	44	60
3	1	T_1 5 6 4 T_1	32	86

CARRIER CAPACITY	60	50	30	20
AVAILABLE	2	2	2	1
ALLOCATED	1	2	0	0

PROBLEM 9

1 DEPOT 4 DEMAND POINTS

(see problem 1)

FINAL SOLUTION
OF GEOMETRICAL SEARCH OF SAVINGS

NO.	DEPOT	ROUTES	LOAD	COST
1	1	T 1 2 T	12	240
2	1	T 3 4 T	8	260

TOTAL COST=500

CARRIER CAPACITY	15	10
AVAILABLE	1	1
ALLOCATED	1	1

PROBLEM 10

1 DEPOT 6 DEMAND POINTS

(see problem 3)

FINAL SOLUTION
OF GEOMETRICAL SEARCH OF SAVINGS

NO.	DEPOT	ROUTES	LOAD	COST
1	1	T 5 1 6 T	49	88
2	1	T 4 5 3 T	54	63

TOTAL COST=151

CARRIER CAPACITY	60	50	40
AVAILABLE	2	1	1
ALLOCATED	1	1	1

PROBLEM 11

1 DEPOT 5 DEMAND POINTS

DEMAND POINT	DEMAND AT THE POINT
1	7
2	8
3	8
4	6
5	9

CARRIER CAPACITY	NUMBER
40	1
30	1

COST MATRIX

$$\begin{array}{cccccc}
P_0 & & & & & \\
70 & P_1 & & & & \\
60 & 30 & P_2 & & & \\
50 & 26 & 24 & P_3 & & \\
50 & 30 & 40 & 24 & P_4 & \\
45 & 40 & 50 & 26 & 30 & P_5
\end{array}$$

FINAL SOLUTION (GEOMETRICAL SEARCH)

NO.	DEPOT	ROUTES	LOAD	COST
1	1	T 3 1 4 T	21	126
2	1	T 2 5 T	17	155

TOTAL COST=281

CARRIER CAPACITY	40	30
AVAILABLE	1	1
ALLOCATED	1	1

PROBLEM 12

1 DEPOT 10 DEMAND POINTS

DEMAND POINT	DEMAND AT THE POINT
1	10
2	4
3	8
4	7
5	12
6	6
7	5
8	10
9	8
10	15

CARRIER CAPACITY	NUMBER
30	3

COST MATRIX

P_0										
8	P_1									
5	6	P_2								
3	5	3	P_3							
2	10	7	5	P_4						
12	7	11	10	14	P_5					
8	6	8	16	10	5	P_6				
3	8	7	4	4	10	6	P_7			
8	16	15	11	6	19	14	9	P_8		
7	9	10	7	8	8	4	4	7	P_9	
7	13	12	9	7	14	9	6	7	6	P_{10}

FINAL SOLUTION OF GEPMETRICAL SEARCH OF SAVINGS

NO.	DEPOT	ROUTES	LOAD	COST
1	1	T 10 2 8 1 T	39	56
1	1	T 7 9 5 4 6 3 T	38	48

TOTAL COST=104

CARRIER CAPACITY	40
AVAILABLE	3
ALLOCATED	2

A MINIMUM 1-ARBORESCENCE ALGORITHM FOR NON-SYMMETRIC CARRIER-DISPATCHING PROBLEMS

by

PETER JIN-HER WAN

B.E. (I.E.), Chung Yuang Christian College of Science and Engineering, Chung Li, Taiwan, Republic of China, 1968

AN ABSTRACT OF A MASTER'S REPORT

submitted to partial fulfillment of the requirement for the degree

MASTER OF SCIENCE

Department of Industrial Engineering

Kansas State University

Manhattan, Kansas

1972

ABSTRACT

This paper is concerned with carrier-dispatching problems which consist of a number of destinations or demand points to be covered by a number of carriers. It is required to determine a set of delivery or pick-up routes which satisfy the imposed constraints and such that the total traveling cost is minimized. The problem under consideration has deterministic demands and non-symmetric cost matrix. One of the practical approaches is based on savings concepts. The computation by such a procedure, however, is tedious because the links are chosen from all possible potential candidates.

A heuristic algorithm based on the minimum 1-arborescence concept has been developed for solving both single- and multi-depot carrier-dispatching problems. In addition, a geometrical search has been examined for both symmetric and non-symmetric problems. Several sample problems with single and multiple depots are used to illustrate the algorithm. The results show that the proposed method requires little time when calculated by hand.