THE TRUCK DISPATCHING PROBLEM*

G. B. DANTZIG[1] AND J. H. RAMSER[2]

The paper is concerned with the optimum routing of a fleet of gasoline delivery trucks between a bulk terminal and a large number of service stations supplied by the terminal. The shortest routes between any two points in the system are given and a demand for one or several products is specified for a number of stations within the distribution system. It is desired to find a way to assign stations to trucks in such a manner that station demands are satisfied and total mileage covered by the fleet is a minimum. A procedure based on a linear programming formulation is given for obtaining a near optimal solution. The calculations may be readily performed by hand or by an automatic digital computing machine. No practical applications of the method have been made as yet. A number of trial problems have been calculated, however.

1. Introduction

The "Truck Dispatching Problem" formulated in this paper may be considered as a generalization of the "Traveling-Salesman Problem".(1) In its simplest form the Traveling-Salesman Problem is concerned with the determination of the shortest route which passes through each of n given points once. Assuming that each pair of points is joined by a link, the total number of different routes through n points is $\frac{1}{2}n!$. Even for small values of n the total number of routes is exceedingly large, e.g. for $n = 15$, there are 653,837,184,000 different routes. One of the authors has collaborated with Fulkerson and Johnson in developing a "cutting plane" approach for testing whether a proposed tour is optimal or finding an improved solution if it is not.(2), (3)

The Traveling-Salesman Problem may be generalized by introducing additional conditions. Thus, the salesman may be required to return to the "terminal point" whenever he has contacted m of the $n - 1$ remaining points, m being a divisor of $n - 1$. For given n and m the problem is to find loops such that all loops have a specified point in common and total loop length is a minimum. Since the loops have one point in common, this problem may be called the "Clover Leaf Problem". If m is small, optimal sets of m points may often be determined by inspection of a map which contains the points and the arcs connecting them. One would look for "clusters of points" and determine by trial and error the order in which they should be traversed, taking care that no loop crosses itself. However, when clusters are not present in sufficient numbers or when m is large, this procedure becomes inapplicable. In this case near-best solutions may be obtained by the algorithm in this paper.

2. The Truck Dispatching Problem

The Traveling Salesman Problem may also be generalized by imposing the condition that specified deliveries q_i be made at every point P_i (excepting the terminal point). If the capacity of the carrier

* Received November 1958

[1] *The* RAND *Corporation, Santa Monica, California*

[2] *The Atlantic Refining Company, Philadelphia, Pennsylvania*

$$C \geqq \sum_i q_i , \tag{1}$$

the problem is formally identical with the Traveling-Salesman Problem in its original form since the carrier can serve every delivery point on one trip which links all the points.

In the "Truck Dispatching Problem" the relation between C and the q_i is such that the carrier can only make between 1 and t deliveries on each trip. Thus, the Truck Dispatching Problem is characterized by the relation

$$C \ll \sum_i q_i . \tag{2}$$

It is obvious that the number of carriers does not enter the problem when they have the same capacity. Even when carriers of different capacities are involved, or when a number of different products are to be delivered to every point, the same mathematical model may be used as will be shown below. For simplicity of presentation it will be assumed first that only one product is to be delivered and that all trucks have the same capacity C.

The method of solution starts from the basic idea to synthesize the solution in a number of "stages of aggregation" in which suboptimizations are carried out on pairs of points or groups. The number of stages of aggregations to be employed is determined as follows: order the deliveries, q_i, in a sequence q_1, q_2, $\cdots$ q_i, q_{i+1}, $\cdots$ q_n such that $q_i \leqq q_{i+1}$ for any $i = 1, \cdots n - 1$. Then determine t such that

$$\sum_{i=1}^{t} q_i \leqq C \quad \text{and} \quad \sum_{i=1}^{t+1} q_i > C.$$

t obviously represents the maximum number of deliveries which a truck of capacity C can make for a given set of q_i's. Since the sequence q_1, q_2, $\cdots$ q_t represents a feasible combination it may be in the optimal solution. Therefore, the method of calculating the number of aggregations to be employed must admit the combination q_1, q_2, $\cdots$ q_t in the final aggregation. This will be the case if the number of stages of aggregations, N, is determined such that

$$N \simeq \log_2 t$$

since 2^N is the largest number of points aggregated in the Nth and final stage of aggregation.

For example if $C = 6000$ and the q_i are the values shown in column Q of Table 1, the ordered sequence of q_i's is

$$1100, 1200, 1200, 1400, 1400, 1500, \cdots 1900$$

and $t = 4$. Therefore $N = \log_2 4 = 2$. In the first stage of aggregation only those points are allowed to pair up whose combined demand does not exceed $\frac{1}{2}C$. In the second stage of aggregation any pair contained in the suboptimal solution of stage 1 may then be combined with any other pair without exceeding the truck capacity. If $C = 7000$ and the q_i's are again the values listed in Table 1, $t = 5$ and $N = 3$. In the first stage of aggregation only those points are allowed to pair up whose combined demand does not exceed $\frac{1}{4}C$. The pairings in the

TABLE 1

Q	P_0											
1200	1 9	P_1										
1700	1 14	5	P_2									
1500	1 21	12	* 7	P_3								
1400	1 23	22	* 17	10	P_4							
1700	1 22	21	* 16	* 21	* 19	P_5						
1400	1 25	24	* 23	30	28	* 9	P_6					
1200	1 32	31	26	27	25	10	7	P_7				
1900	1 36	* 35	* 30	* 37	* 35	* 16	* 11	* 10	P_8			
1800	1 38	37	* 36	* 43	* 41	* 22	* 13	16	* 6	P_9		
1600	1 42	41	* 36	* 31	29	* 20	17	10	* 6	* 12	P_{10}	
1700	1 50	49	* 44	* 37	* 31	* 28	* 25	18	* 14	* 12	* 8	P_{11}
1100	1 52	51	46	39	29	30	27	20	16	20	10	10	P_{12}

second stage then do not exceed $\frac{1}{2}C$ and those in the third stage do not exceed C. Although $N = 2$ is a closer approximation to $\log_2 5$ than $N = 3$, three stages must be employed since a total of 2 stages does not admit the feasible combination 1100, 1200, 1200, 1400, 1400, which may be optimal.

The reader should note that if a truck were constrained to visit precisely two points and return, the total distance covered would be the constant sum of distances from the terminal to each point P_i plus the *sum of interpair distances* (we would therefore seek to minimize the latter). Accordingly, in each intermediate stage the optimum pairings corresponding to minimum interpair distances are determined. In the final stage optimum pairings are determined such that the sum of trip lengths is a minimum. The details of this procedure will be illustrated below by way of a numerical example.

The Truck Dispatching Problem may now be formally stated as follows:

[1] Given a set of n "station points" P_i $(i = 1, 2 \cdots n)$ to which deliveries are made from point P_0, called the "terminal point".

[2] A "Distance Matrix" $[D] = [d_{ij}]$ is given which specifies the distance $d_{ij} = d_{ji}$ between every pair of points $(i, j = 0, 1, \cdots n)$.

[3] A "Delivery Vector" $(Q) = (q_i)$ is given which specifies the amount q_i to be delivered to every point P_i $(i = 1, 2 \cdots n)$.

[4] The truck capacity is C, where $C > \max. q_i$.

[5] If $x_{ij} = x_{ji} = 1$ is interpreted to mean that points P_i and P_j are paired

$(i, j = 0, 1 \cdots n)$ and if $x_{ij} = x_{ji} = 0$ means that the points are not paired, one obtains the condition

$$\sum_{j=0}^{n} x_{ij} = 1 \qquad (i = 1, 2 \cdots n) \tag{3}$$

since every point P_i is either connected with P_0 or at most one other point P_j. Furthermore, by definition, $x_{ii} = 0$ for every $i = 0, 1, \cdots n$.

[6] The problem is to find those values of x_{ij} which make the total distance

$$D = \sum_{i,j=0}^{n} d_{ij}x_{ij} \tag{4}$$

a minimum under the conditions specified in [2] to [5].

3. Computational Procedure

3.1. *General Remarks*

According to condition [5] x_{ij} can only assume values which are either 1 or 0. At the present time there exists no general applicable method for solving discrete variable problems except possibly some variant of a recent proposal of R. Gomory (4) which is in too early a stage of development to evaluate for the problem at hand. However, one may determine "best solutions" if one admits the weaker condition

$$0 \leqq x_{ij} \leqq 1 \tag{5}$$

and then applies the well-known methods of linear programming. (5) The possible appearance of fractional values in the "solution" indicates the existence of alternative pairings of points or groups of points. Experience has shown that the number of such alternative pairings is small, so that the pairing yielding least mileage can be readily found by trial and error. The "solution" then obtained satisfies the requirement that x_{ij} be either 0 or 1; however, the method, like other now available algorithms for the solutions of discrete-variable problems, does not guarantee that the absolute optimum solution is obtained. It is likely that the "best solution" obtained by this method approaches the optimum as the number of points increases. Moreover, an estimate is at hand on the error for the minimum D since $x_{ij} = 0$ or 1 lies between the "best solution" obtained by the method of this paper and the minimum satisfying $0 \leqq x_{ij} \leqq 1$.

3.2. *First-Stage Aggregations*

The detailed computational procedure may best be illustrated by the numerical example shown in Table 1. Deliveries are made from point P_0 to points $P_1, \cdots P_{12}$. The entry in the lower right corner of each box is the shortest distance d_{ij} between points P_i and P_j. These entries are the elements of the distance matrix discussed in [2]. The delivery vector (Q) is shown at the left of the triangular array. If $C = 6000$ gal., the number of stages is 2 (see Section 2). The x_{ij} are entered into the left upper corner of each box. As a start all delivery points $P_1, \cdots P_{12}$ may be paired with the terminal P_0 so that there are

12 entries $x_{0,i} = 1$ where $i = 1, \cdots 12$. These 12 entries constitute the "basic set" at the start of computation. During each following iteration exactly one element of the basic set is eliminated and replaced by a new element. The total number of basic x_{ij}-entries remains therefore constant during stage 1. The remaining left upper corners in each box remain either empty or are permanently supplied with a star. The empty boxes constitute the "non-basic set" of entries which are all equal to 0. However, the zeros are not actually entered in order to distinguish them from the zeros which belong to the basic set. This distinction is necessary because there are no more than 12 basic entries in this particular example. A starred entry indicates that pairing of the respective points is not admissible during first stage aggregations since the combined demand for such points exceeds one half the truck capacity. Thus, the combined demand for P_5 and P_8 is $1700 + 1900 = 3600$ which is greater than $C/2 = 3000$.

3.3. *Rapid Corrections*

The starting solution, in which each point is paired with the terminal, may be readily improved by a series of rapid corrections. This is desirable since the number of subsequent iterations decreases as the difference between the starting and optimal solution decreases.

Rapid corrections may be made by bringing into the solution non-basic entries which correspond to relatively small d_{ij}-values. Thus, $d_{6,7} = 7$ is relatively small and may be brought into the basic set by entering the, as yet undetermined, value θ into the corresponding left upper corner. In order to satisfy equations (3), the sums of basic entries for any $i = 1, 2, \cdots n$ must be equal to 1. Since the distance matrix (D) is symmetrical, it follows that the sum of basic values in row 6 and column 6 in the triangular array shown in Table 1 must also be equal to 1. The same holds true for row 7 and column 7. Therefore, the following entries are made:

$$x_{6,7} = \theta \qquad \text{(non-basic entry } x_{6,7} = 0 \text{ increased)}$$

$$x_{0,6} = 1 - \theta \qquad \text{(basic entry } x_{0,6} = 1 \text{ corrected)}$$

$$x_{0,7} = 1 - \theta \qquad \text{(basic entry } x_{0,7} = 1 \text{ corrected).}$$

The largest value of θ consistent with the inequalities (5) is 1. Therefore

$$x_{6,7} = 1 \qquad x_{0,6} = 0 \quad \text{and} \quad x_{0,7} = 0.$$

Since one basic entry must be dropped, there is a choice between $x_{0,6}$ and $x_{0,7}$. Since the distance $d_{0,7}$ is larger than $d_{0,6}$, the basic entry $x_{0,7}$ is dropped from the basic set in order to reduce total distance as much as possible. The overall effect of the sequence of operations just discussed is that the entry $x_{6,7} = 1$ has been added to the basic set, that the value of $x_{0,6}$ has been changed from 1 to 0 and that $x_{0,7}$ has been dropped. Geometrically this means that points 6 and 7 have been severed from the terminal and connected with each other. This process of making rapid corrections is repeated as long as non-basic entries with obviously low d_{ij}-values are available. A good rule for obtaining a quick improvement of the solution is to skip over any d_{ij}-values if entries $x_{0,i}$ or $x_{0,j}$ have

TABLE 2

P_0 0												
0 9	P_1 9											
14	1 5	P_2 -4										
0 21	12		P_3 21									
23	22		1 10	P_4 -11								
1 22	21				P_5 22							
0-θ 25	24		30	28		P_6 25						
32	31	26	27	25	10	1 7	P_7 -18					
1 36								P_8 36				
1 38	37						16		P_9 38			
1-θ 42	41			29		θ 17	10			P_{10} 42		
0 50	49						18				P_{11} 50	
52	51	46	39	29	30	27	20	16	20	10	1 10	P_{12} -40

already been dropped from the basic set or have the value 0. Starting with Table 1 the first four iterations are as follows:

Iteration (1): $x_{1,2} = 1$ replaces $x_{0,2}$; $x_{0,1} = 1$ becomes $x_{0,1} = 0$

Iteration (2): $x_{6,7} = 1$ replaces $x_{0,7}$; $x_{0,6} = 1$ becomes $x_{0,6} = 0$

Iteration (3): $x_{3,4} = 1$ replaces $x_{0,4}$; $x_{0,3} = 1$ becomes $x_{0,3} = 0$

Iteration (4): $x_{11,12} = 1$ replaces $x_{0,12}$; $x_{0,11} = 1$ becomes $x_{0,11} = 0$

The new basic set is shown in Table 2 in which the θ-entries should be disregarded for the moment. As a result of the first four rapid corrections the sum of interpair distances has been reduced from 364 to 170 units.

3.4. *The Delta-Function (relative cost factors)*

When a sufficient number of pairs of points with small interpoint distances have been brought into the solution, it will become increasingly difficult to bring in additional pairs of points without calculating the total distance in every case. This is equivalent to a trial and error procedure which would necessitate an enormous amount of computation even if the number of points is only moderately large. What is needed therefore is a criterion which enables the computer to either accept or reject a non-basic entry as the array in Table 2 is scanned box by box. This criterion is provided by the "Delta-Function" defined as

$$\delta_{ij}^{(n)} = \pi_i^{(n)} + \pi_j^{(n)} - d_{ij} \tag{6}$$

where $\pi_i^{(n)}$ and $\pi_j^{(n)}$ are suitably determined constants characteristic for the nth iteration. By definition $\pi_i^{(n)}$ and $\pi_j^{(n)}$ are determined so that

$$\delta_{ij}^{(n)} = 0 \tag{7}$$

for all d_{ij} corresponding to basic entries. For non-basic entries

$$\delta_{ij}^{(n)} \gtrless 0. \tag{8}$$

The delta-function indicates how much the total distance D will decrease per unit increase of a non-basic entry x_{ij}. Thus, if $\delta_{ij}^{(n)} > 0$, the total distance D decreases if the value of the non-basic entry x_{ij} is increased; if $\delta_{ij}^{(n)} < 0$ the total distance D increases if the corresponding non-basic value is increased. Therefore, in scanning through the array in Table 2, boxes with $\delta_{ij}^{(n)} < 0$ are disregarded unless the $\delta_{ij}^{(n)}$ for all non-basic entries are negative. In this case every possible choice of non-basic entries leads to an increase of total distance D represented by the basic set of entries. The particular set obtained at this point represents then the "best solution".

However, as long as at least one $\delta_{ij}^{(n)}$ is positive, a reduction of total distance is possible. If several $\delta_{ij}^{(n)}$ are larger than 0, the largest possible reduction of distance relative to increase in x_{ij} is obtained with the largest value of $\delta_{ij}^{(n)}$. Therefore, in scanning through the array a non-basic entry is accepted if its $\delta_{ij}^{(n)}$ is greater than the largest previously considered $\delta_{ij}^{(n)}$ and a non-basic entry is rejected if its $\delta_{ij}^{(n)}$ is smaller than the largest previously noted $\delta_{ij}^{(n)}$. In the case of equality it may also be rejected in order to eliminate arbitrariness of formal procedure.

The constants $\pi_i^{(n)}$ and $\pi_j^{(n)}$ in the delta-function (6) may be determined from condition (7). This yields the equation

$$\pi_i^{(n)} + \pi_j^{(n)} - d_{ij} = 0 \tag{9}$$

in which one may arbitrarily set $\pi_0 = 0$ (any choice of an integer $\geqq 0$ is admissible). Since there are 12 basic entries in the numerical example of Table 2, there are 12 equations of type (9) from which the 12 π_i-values corresponding to each delivery point P_i may be determined. These values and $\pi_0 = 0$ are entered in the lower right corner of the boxes containing the point identifications as shown in Table 2.

After obtaining the $\pi_i^{(n)}$-values, the $\delta_{ij}^{(n)}$ are calculated from (6) and max $\delta_{ij}^{(n)} = \delta_{rs}^{(n)}$ determined in the manner described above. The non-basic entry $x_{rs}^{(n)}$ is then set equal to θ and the basic entries corrected where necessary by substracting θ in such a way that the sum of entries in the rth row and rth column, as well as in the sth row and the sth column, are separately equal to 1. In Table 2 max $\delta_{ij} = \delta_{6,10} = 25 + 42 - 17 = 50$; accordingly, the non-basic entry $x_{6,10} = 0$ was replaced by $x_{6,10} = \theta$. The θ-corrections of basic entries are then made as shown. It should be noted that a θ-correction cannot be applied to the basic entry $x_{6,7} = 1$ since there is no other basic entry in the 7th row and 7th column to which a corresponding correction can be applied. From this point on the

procedure is exactly the same as discussed above except for one additional provision: the maximum value of θ which leaves basic entries non-negative is determined and of those basic entries which are zero, but which would have gone negative if θ would have been chosen larger, exactly one corresponding to the largest d_{ij} is dropped from the basic set. In this case $\theta = 0$ and $x_{0,6} = 0$ is dropped. Although the d_{ij} corresponding to $x_{0,11} = 0$ is larger than the d_{ij} associated with $x_{0,6}$, the basic value $x_{0,6} = 0$ must be dropped instead of $x_{0,11}$ since a small positive value of θ would have driven $x_{0,6}$ negative but would not have affected $x_{0,11}$. If a larger value of θ would have driven both $x_{0,6}$ and $x_{0,11}$ negative, then $x_{0,11}$ should be dropped instead of $x_{0,6}$. This provision concerning the elimination of basic entries which are zero is necessary to prevent cycling.

By iterating this procedure non-basic entries are brought into the basic set until no further improvement is possible. As noted above, this is the case when the delta-function is negative or zero for all non-basic entries. This concludes the first stage aggregations, in which points were paired whose combined demand does not exceed one half the truck capacity. The pairings so obtained are shown in Figure 1. It is seen that 5 aggregations contain 2 station points each whereas 2 contain only one station point, the other being the terminal point P_0.

3.5. *Second-Stage Aggregation*

The pairs of points obtained in the first-stage aggregation for the example may now be combined with each other without exceeding the truck capacity. Therefore, in the second-stage aggregation there will, in general, be no starred entries. The problem is to combine the first-stage aggregations in such a way that the total distance becomes a minimum. The first step will be to set up a triangular distance table for pairs of aggregates similar to Table 1. Designating first-stage aggregations with A_1, A_2, $\cdots$ A_7 (see Figure 1) and the point P_0 with A_0, one obtains the triangular array shown in Table 3. The distances shown in the lower right corner of the column headed by A_0 correspond to the minimal routes from the terminal to each point in the respective aggregate and back to the terminal. The remaining entries represent the distances of the minimal routes from the terminal to each point in the pairs of aggregates, A_i and A_j, and back to the terminal. These closed-loop routes can be readily determined from the distance matrix D if care is taken that no loop crosses itself. For example, the minimal route involving aggregates A_4 and A_5 is $P_0P_6P_7P_{10}P_9P_0$ and not $P_0P_6P_{10}P_7P_9P_0$.

The procedure for finding the combination of aggregates which yields minimum mileage is exactly the same as the one used for first-stage aggregations. Table 3 shows the optimal solution obtained. If it were not for the appearance of fractional values for some x_{ij}, the second stage problem would be solved.

3.6. *Treatment of Fractional x_{ij}'s*

According to relations (5) fractional values are permitted during the computation and no special treatment is required unless fractional values appear in the

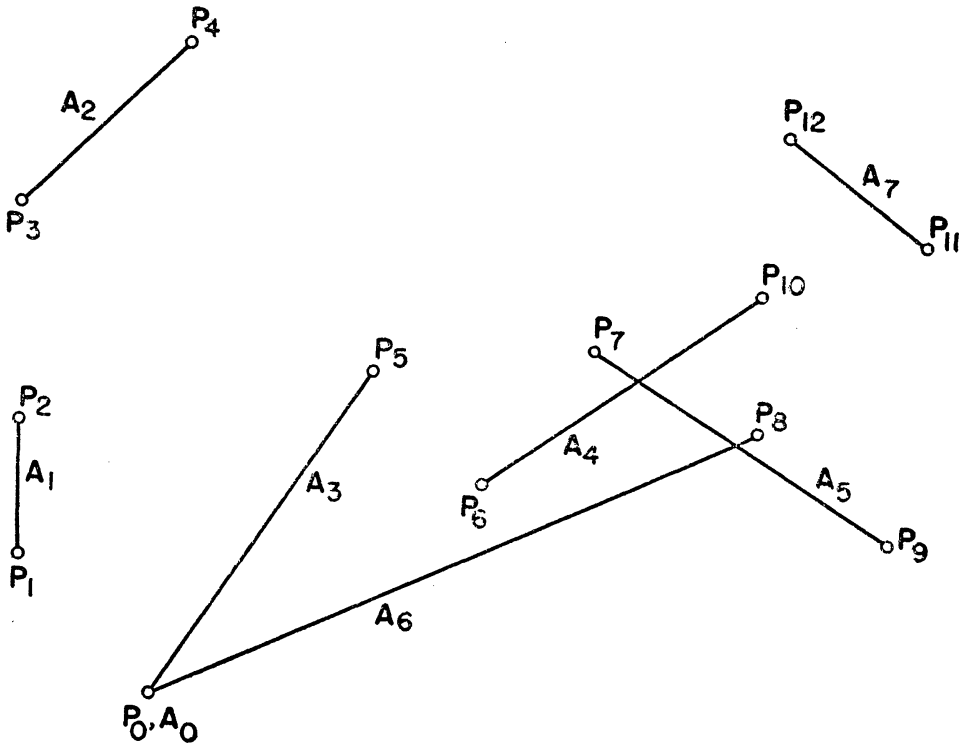

FIG. 1

TABLE 3

A_0 0							
28	A_1 17						
54	1/2 54	A_2 37					
44	1/2 52	1/2 72	A_3 35				
84	92	102	84	A_4 45			
86	94	110	87	0 92	A_5 47		
72	80	102	0 74	1 84	86	A_6 39	
112	120	120	112	112	1 112	112	A_7 65

optimal solution. A fractional value for any x_{ij} in the optimal solution means that no decision has been reached as to whether points P_i and P_j, or aggregates A_i and A_j, are to be linked or not. Fractional solutions always involve one or more groups of fractional x_{ij}, each composed of an odd number of basic entries. Table 3 shows that the optimal solution contains three fractional basic entries involving the group of aggregates A_1, A_2 and A_3. Any two of them may be aggregated with each other leaving the third to be aggregated with the terminal

A_0 . By trial and error it may be readily decided which one of the three alternatives corresponds to minimum mileage. The symbolic diagram shown in Figure 2 may be useful for carrying out the computation most expeditiously. In this diagram A_1 , A_2 and A_3 are placed at the vertices of a triangle; A_0 is placed inside and connected with the vertices. The distances corresponding to the 6 possible aggregates are then taken from Table 3 and placed on the corresponding lines in Figure 2. It is readily seen that the aggregates yielding minimum mileage are (A_1A_2) and (A_0A_3). If the number of fractional entries is 5, the symbols A_i are placed at the vertices of a five-sided polygon and A_0 is placed inside. The value of such diagrams, particularly when the number of fractional entries exceeds three, lies in the fact that the distance relations may be seen at a glance. If there are one or more such odd groups of fractional values at the end of stage 2 or higher, it is probably best to select arbitrarily one of the groups, determine the optimum aggregation for the group and then recompute the stage after the paired A_i's ($i \neq 0$) have been removed from the array. Thus, in the example

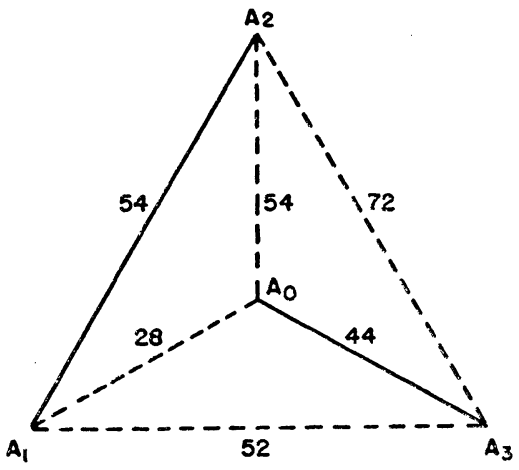

FIG. 2

shown in Table 3, it is recommended that stage 2 be re-solved under the assumption that A_1 and A_2 are aggregated, but that A_3 , A_4 , A_5 , A_6 , A_7 are free to pair up as they please. This leads to the array shown in Table 4, in which there is again an odd group involving the pairs of points A_4 , A_5 , A_7 . By the procedure illustrated in Figure 2, one finds that A_5 must be aggregated with A_7 , leaving A_3 , A_4 , A_6 to pair up as they please. Finally A_3 , A_4 , A_6 are resolved by the same method.

It is possible, of course, to get fractional solutions in stage 1. If there are one or more such odd groups of fractional values at the end of stage 1, each should be resolved separately. Then one may proceed to stage 2.

3.7. *Comparison of Final Solution with True Optimum*

The final solution of the numerical problem discussed in this paper is (A_1A_2), (A_5A_7), (A_4A_6), A_3 . According to Figure 1 this results in the following trip assignments: $(P_1P_2P_3P_4)$, $(P_7P_{12}P_{11}P_9)$, $(P_6P_{10}P_8)$, P_5 . According to Table 3,

the total distance covered on this trip is 294 units. It is believed that the trip assignment $(P_1P_2P_3P_4)$, $(P_7P_{12}P_{11}P_{10})$, $(P_6P_8P_9)$, P_5 with a total distance of 290 units is actually the true optimum solution of this problem. If so, the algorithm developed in this paper has resulted in a "best solution" which comes very close to the true optimum for the numerical example in Table 1. Experience with the method has shown that similar results may be obtained in other numerical cases particularly if the station demands do not differ too widely. It is also conjectured that the difference between the distance for the "best solution" and that of the true optimum decreases as the number of station points increases.

4. Multiple Product Demand

In the Truck Dispatching Problem, as formulated above, it was assumed that a demand for only one product is specified at each point. This restriction is not necessary. If demands $q_{i,k}$ for n different products are specified at each point P_i $(k = 1, 2, \cdots n)$, the total demand for each point is

TABLE 4

A_0 0					
44	A_3 36				
84	84	A_4 46			
86	87	1/2 92	A_5 46		
72	1 74	0 84	86	A_6 38	
112	112	1/2 112	1/2 112	112	A_7 66

$$Q_i = \sum_{k=1}^{n} q_{i,k} \,. \tag{10}$$

If one product can be exchanged for an equal quantity (weight or volume) of any other quantity as far as the carrier is concerned, it is clear that for the solution of the problem only the total demand Q_i at any point P_i needs to be considered. In the case of liquid products, which are delivered in bulk, the ideal carrier, satisfying the condition of complete product interchangeability, must obviously have moveable partitions. In practice this ideal may be closely approximated by suitable truck compartmentation.

5. Multiple Truck Capacity

It was noted above that the mathematical problem of minimizing truck mileage is independent of the number of trucks assigned to the terminal when all trucks have the same capacity. This is no longer the case when the fleet consists of n_i trucks of capacity C_i where $i = 1, 2, \cdots m$. While the tacit assumption was

made above that unused capacity incurs no loss, it is obvious that in practice "maximum use" should be made of the entire capacity of the fleet. To incorporate unused capacity into the model, it would be necessary to introduce the cost for unused unit volume and the cost of unit mileage. The problem then becomes one of minimizing total cost rather than total mileage. In practice the cost of unused unit volume becomes of minor importance if one permits a certain percentage of over- or underdelivery. For this reason it may be best to solve the problem by assuming that all trucks have capacity C_m where C_m is the largest available truck capacity. The assignment of trucks of different capacity to the various trips must then be made in accordance with the total demand for each trip. No general statements concerning such truck assignments can be made since the availability of trucks of different capacity at any one time depends not only on the length of the various trips but also on random conditions such as the severity of traffic.

6. Other Formulations of the Problem

In the problem as formulated above it was assumed that at every station point P_i a demand for certain products was specified and that this demand must be satisfied by one delivery. By relaxing the condition that demand must be satisfied in full, it may be possible to reduce total truck mileage still further. Thus, it may be permissible to under- or overdeliver up to a fixed percentage based on demand. Underdelivery may permit the pairing of near-lying station points which could not be paired if demand is to be satisfied in full. Overdelivery will reduce unused truck capacity. The ultimate in "underdelivery" is reached when deliveries to station points are made in quantities such that no station ever runs dry. This formulation of the problem would not only reduce total mileage to an absolute minimum but should also considerably reduce the cost of computation since daily computation of a truck dispatch plan becomes unnecessary. The feasibility of this approach depends, of course, on the degree of regularity of consumption at the various station points and the several fixed charges incurred by several unloadings to satisfy a single order.

Literature Cited

1. Flood, Merrill M., "The Traveling-Salesman Problem", *J. Opns. Res. Soc. Am. 4*, 61–75 (1955).
2. Dantzig, G. B., R. Fulkerson and J. Johnson, "Solution of a Large-Scale Traveling-Salesman Problem", *J. Opns. Res. Soc. Am. 2*, 393–410 (1954).
3. Dantzig, G. B., "Discrete-Variable Extremum Problems", *J. Opns. Res. Soc. Am. 5*, 266–277 (1957).
4. Gomory, R., "Essentials of an Algorithm for Integer Solutions to Linear Programs", (communicated to *Bull. Amer. Math. Society* in a letter from Princeton, dated April 23, 1958).
5. Dantzig, G. B., "Maximization of a Linear Function of Variables Subject to Linear Inequalities", Chapter XXI in "Activity Analysis of Production and Allocation", edited by T. C. Koopmans, Cowles Commission Monograph Nr. 13, John Wiley & Sons, New York, 1951.