An Algorithm for the Vehicle-dispatching Problem

N. CHRISTOFIDES and S. EILON
Imperial College of Science and Technology

The vehicle-scheduling problem involves the design of several vehicle tours to meet a given set of requirements for customers with known locations, subject to a capacity constraint for the vehicles and a distance (or time) constraint for vehicle tours. Three methods of solution are considered in this paper:
(A) A branch-and-bound approach.
(B) The "savings" approach.
(C) The 3-optimal tour method.
The excessive computation time and computer storage required for the first method renders it impracticable for large problems. Ten problems are examined and the results suggest that method C is superior to the other two methods.

INTRODUCTION

THE PROBLEM of vehicle scheduling was first formulated by Dantzig and Ramser[1] and may be stated as follows: A set of customers, each with a known location and a known requirement for some commodity, is to be supplied from a single depot by delivery vehicles of known capacity. The problem is to design the routes for these vehicles, subject to the following conditions and constraints:

(a) The requirements of all the customers must be met.
(b) The capacity of vehicles may not be violated (i.e. the total load allocated to each may not exceed its capacity).
(c) The total time (or alternatively distance) for each vehicle to complete its tour may not exceed some predetermined level (this is usually a legal or a contractual condition).

The objective of a solution may be stated in general terms as that of minimizing the total cost of delivery, namely the sum of costs associated with the fleet size and the costs of completing the delivery tours. Several subproblems and associated problems may be formulated:

(1) If the fleet consists of a single vehicle having a sufficiently large capacity, and if constraint (c) is ignored, find the shortest tour (in terms of time or distance) to visit all customers. This is the well-known travelling-salesman problem.

(2) If constraint (c) is ignored, find the smallest number of vehicles that would be needed. This problem of the fleet-size is reminiscent of the knapsack problem.

(3) For the smallest possible number of vehicles compatible with the conditions (a), (b) and (c), design the vehicle routes so that the total distance of the tours is minimized.

(4) If vehicles of different capacities can be chosen, find the optimal mix of vehicles, such that the total cost of delivery is minimized.

(5) For a given set of customers, whose requirements are not deterministic but are given in probabilistic terms, find the solution to problems 2 and 4.

These variations on the same theme highlight the numerous areas of application in controlling the operations of vehicle fleets, used either for delivery or for collection purposes. Routing of cargo vessels has similar problems, and certain facets of job-shop scheduling have the same mathematical structure. This paper is concerned with problem 3, namely with finding a solution that will satisfy the conditions (a), (b) and (c) to supply a given set of customers with known requirements, such that the number of vehicles is minimum, and for this number of vehicles the total distance travelled is minimum.

SOLUTIONS

The problem can be formulated in dynamic programming terms or as one of assignment in integer programming,[2,3] but these methods involve a great deal of computation and demand very large computer storage. The other methods considered here are:

(A) A branch-and-bound approach.

(B) The "savings" approach.

(C) The 3-optimal tour method.

(A) *A branch-and-bound approach*

This method is based on the branch-and-bound algorithm of Little *et al.*[4] for solving the travelling-salesman problem, where the final tour passes through each customer and through the depot only once. In the vehicle-scheduling problem, if the individual vehicle tours are combined into a single tour, this tour visits the depot as many times as there are vehicle tours. The vehicle-scheduling problem may therefore be formulated as a travelling-salesman problem by eliminating the real depot and replacing it by N artificial depots, all located in the same positions. Travelling from one artificial depot to another is prohibited by setting the distance between any two depots equal to infinity, as shown in the cost matrix in Figure 1, and a solution is now sought to this cost matrix, subject to the capacity constraints of the vehicles.

The number of artificial depots N is equal to the number of vehicles employed in the final solution, and this number has a lower bound determined by the vehicle capacity, namely:

$$N \geqslant \sum^{n} q_i / C$$

where q_i is the requirements for customer i ($i = 1, 2, \ldots, n$) and C the vehicle capacity.

The problems may now be solved for several values of N and the best solution is chosen. Our experience has been that usually no more than four values of N

need be considered (starting with the lowest possible value of N and then increasing it by one at a time) for the final solution to be obtained. It is, of course, possible that for the minimum value of N, and even for some higher

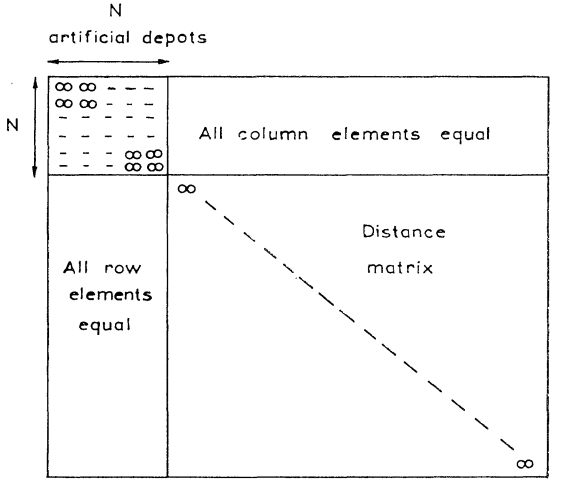

FIG. 1. *Matrix for the vehicle-scheduling algorithm.*

values, no solution is feasible, but as N is successively increased feasible solutions eventually emerge, and it is from these solutions that the final choice is made.

Before branching to a new node, it is necessary to check that the constraints have not been violated:

(1) Does the total load for the vehicle in question exceed its capacity?
(2) Does the total distance accumulated so far for this vehicle tour exceed the pre-set limit?
(3) If we are considering at present the kth tour (leaving $N-k$ tours to be planned), will the remaining customer requirements for the remaining n' customers exceed the capacity of $N-k$ vehicles, namely is:

$$\sum^{n'} q_i/C > N-k?$$

If the answer to any of these questions is in the affirmative, there is no need to proceed any further in exploring the branch under consideration.

In addition to the three questions which describe the capacity and distance constraints, it is necessary to determine bounds for nodes of the decision tree, in order to reduce the amount of search that is involved. The method of calculating bound values suggested by Little *et al.*[4] is quite efficient, but we found that we could marginally improve on these values by calculating the

minimal spanning tree in the following way. The travelling-salesman route through n points has the property that it is a closed loop and that each point has two links to two other points. A spanning tree is a configuration of $n-1$ straight lines passing through the n points and a minimal spanning tree is one with the shortest sum of links. Figure 2 shows an example of a minimal travelling-salesman tour in (a); the spanning tree in (b) is obtained by removing a link from

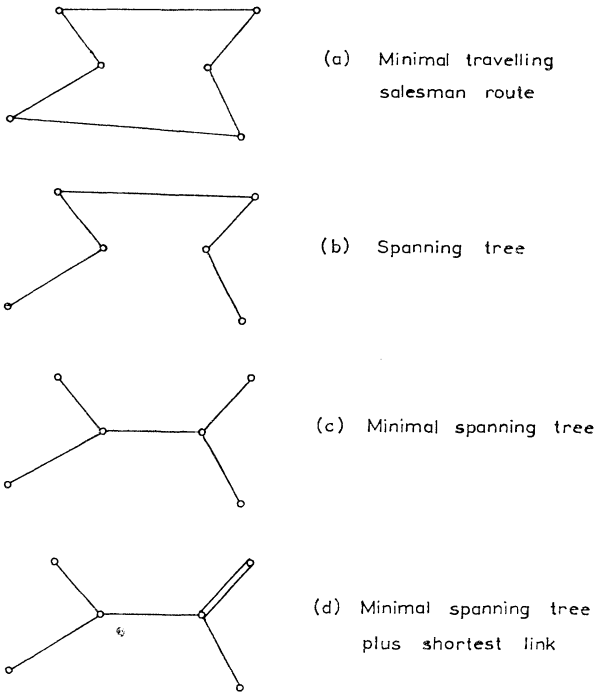

FIG. 2. *Lower bound to a travelling-salesman problem.*

(a), so that (b) is a lower bound for (a); the minimal spanning tree in (c) is also a lower bound. As the latter configuration consists of $n-1$ links, a better lower bound for the minimal travelling-salesman tour can be obtained by adding a suitable link, such as the shortest link in the network as in (d). Alternatively, one point may be added by placing it at the same position as an existing point but assigning an infinite distance between them, thus converting the set of n points to that of $n+1$ points. The minimal spanning tree of the $n+1$ points is a lower bound for the minimal travelling-salesman tour of the original n points. The minimal spanning tree can be readily computed by the method suggested by Kruskal[5] and the bounds obtained in this way are marginally better than those generated by the method of Little *et al.*[4]

(B) *The "savings" approach*

This method was developed by Clarke and Wright.[6] It does not guarantee optimality for the vehicle routes, but it is simple to use and practical constraints can be incorporated quite easily. Nevertheless, the results obtained by this method become progressively worse as the constraints are made more stringent. The method is described by the following procedure:

(1) Start by assuming that one truck visits exactly one customer and then returns to the depot.

(2) If customers i and j are joined together to form one tour, then: (a) one truck is eliminated and (b) there is a saving in mileage given by:

$$s_{ij} = d_{0i} + d_{0j} - d_{ij},$$

where d_{0i}, d_{0j} is the distance from depot to customers i and j respectively and d_{ij} the distance from i to j.

(3) If the link ij is feasible (i.e. it satisfies the constraints of capacity and distance), it is added, otherwise another link is examined, until no more links can be added, and we are then left with the final tours.

Clarke and Wright suggested that the savings s_{ij} for each possible link be calculated first, arranged in descending order and then each link examined in turn by going down the list.

(C) *The r-optimal tour method*

Flood[7] was first to observe that the minimal travelling-salesman tour does not intersect itself, a property that is perhaps obvious from geometrical considerations. Croes[8] went on to describe how an intersectionless tour can be obtained from any arbitrary initial tour by opening two links and reconnecting the "open" four points by two new links (or by "inversion", as he called it). Lin[9] noted that the intersectionless tour is equivalent to one which cannot be reduced in length by replacing any two of its links by two other links and such a tour may be called 2-optimal. The term "2-optimal" implies that no improvement is possible by eliminating two links and replacing them by two others. The 2-optimal tour is not necessarily identical to the optimal tour, and it is of course possible to have several 2-optimal tours for any given problem. Lin[9] then proceeded to show how a tour can be produced such that replacing any three of its links will not reduce the total distance, and he called such a tour a 3-optimal tour. Figure 3 shows an example in which three links are removed from a tour, leaving three disconnected chains, which can then be connected in eight different ways. When this exercise is repeated for all the possible combinations for removing three links, the 3-optimal tour for this example is then obtained.

The concept of a 3-optimal tour may be extended to an r-optimal tour, and clearly the minimal tour through n points has the property that it is an n-optimal tour. The number of checks required for all the alternatives to be compared

increases rapidly with r. For any given tour there are $\binom{n}{r}$ ways in which r links can be chosen and removed, leaving r disconnected chains, some of which (say x) consist of single points.

For the n-optimal tour $x = r = n$ and the number of checks that have to be made becomes $\frac{1}{2}(n-1)!$ which is the total number of possible tours, and this is equivalent to complete enumeration.

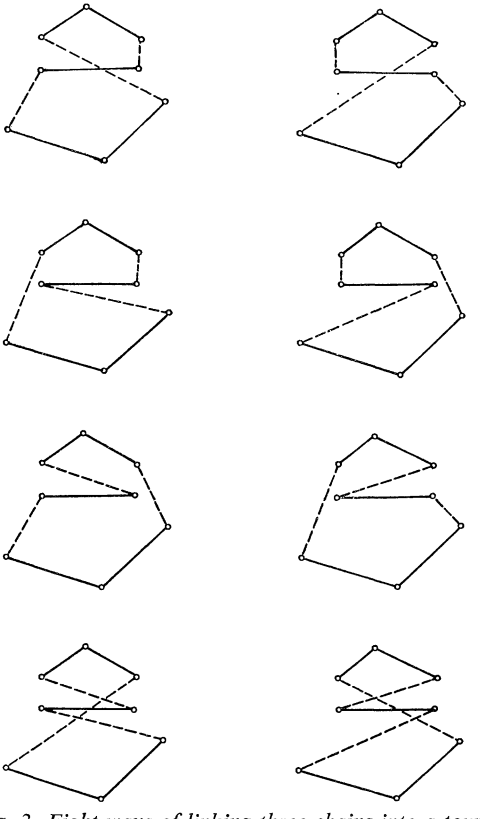

FIG. 3. *Eight ways of linking three chains into a tour.*

Experiments suggest that r-optimal tours give results that are very close to the minimal tour. The probability that an r-optimal tour is the minimal tour naturally depends on the distribution of the points in the particular problems. For example, in the special case when the points lie on the vertices of a convex polygon there is only one 2-optimal tour, one 3-optimal, one 4-optimal, etc., and all these tours are identical and form the minimal tour. In such a case, a procedure designed to produce the 2-optimal tour will automatically produce the minimal tour.

In general, the probability that the r-optimal tour is the minimal increases with r, but, as we have seen, the amount of computations required to produce the r-optimal tour increases very rapidly with r. Our experience with several problems suggested that the 3-optimal tour produces very good results and the procedure adopted here is as follows:

(1) Start with an arbitrary random tour.
(2) Find a 2-optimal tour; this tour serves as a starting point for (3).
(3) Find a 3-optimal tour.
(4) Repeat several times and select the best solution.

A flow diagram of this algorithm is shown in Figure 4. The constraints for vehicle capacity and allowed distance are checked in the same way as in the branch-and-bound method described earlier.

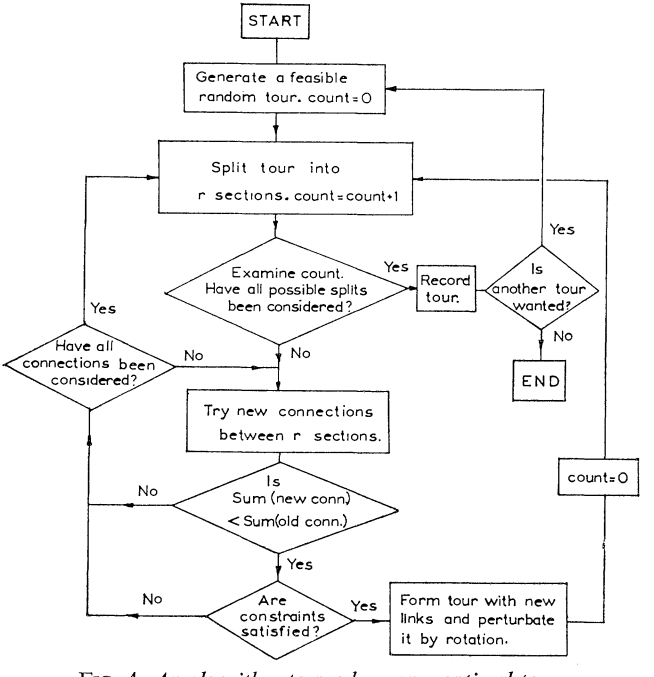

FIG. 4. *An algorithm to produce an r-optimal tour.*

SOME COMPUTATIONAL RESULTS

The three methods described earlier were used to solve ten vehicle scheduling problems that are listed in Table 1; the first seven problems were described in previous papers, and details about the last three are given in the Appendix.

The results are compared in Table 2, from which it is concluded that method (C) (namely the algorithm that produces 3-optimal tours) is by far the best. The branch-and-bound method was tested only for problems 1 and 2, because both

the computation time and memory-space requirements of this method became prohibitive for the other problems.

Clearly, the computational efficiency of the branch-and-bound algorithm when applied to the vehicle-scheduling problem is substantially reduced compared with its efficiency in solving an equivalent travelling-salesman problem.

Only for problem 2 is the solution found by the savings method as good as that found by the 3-optimal method. For problems 1, 3, 4, 6, 9 and 10 the least cost tours found by the 3-optimal method are shorter, and for problems 5, 7 and 8 the solutions are better with respect to both distance and number of vehicles used.

TABLE 1. LIST OF PROBLEMS

Problem No.	Customers	Origin
1	6	Hayes[3]
2	13	Dantzig and Ramser[1]
3, 4, 5	21, 22, 29	Gaskell[10]
6	30	Clarke and Wright[6]
7	32	Gaskell[10]
8	50	New problems—see Appendix
9	75	
10	100	

TABLE 2. RESULTS FOR TEN VEHICLE-SCHEDULING PROBLEMS

	Mileage			No. of vehicles			Computation time†		
	Method (A)	Method (B)	Method (C)	Method (A)	Method (B)	Method (C)	Method (A)	Method (B)	Method (C)‡
Problem									
1	114	119	114	2	2	2	1·5	0·1	0·1
2	290	290	290	4	4	4	15	0·1	0·1
3	—	598	585§	—	4	4	—	0·1	0·6
4	—	955	949	—	5	5	—	0·1	0·5
5	—	963	875§	—	5	4	—	0·2	0·8
6	—	1427	1414	—	8	8	—	0·2	0·8
7	—	839	810	—	5	4	—	0·2	0·8
8	—	585	556	—	6	5	—	0·6	2·0
9	—	900	876	—	10	10	—	1·3	4·0
10	—	887	863	—	8	8	—	2·5	10·0

† IBM 7090 (min).

§ Best of ten runs.

‡ Time for one run (a run is defined as a computation procedure starting from a random tour, followed by determining the 2-optimal tour, and finally the 3-optimal tour).

Method (A) A branch-and-bound approach; (B) the "savings" approach; (C) the 3-optimal tour method.

APPENDIX

Details of problem 8

No.	x	y	q	No.	x	y	q	No.	x	y	q
1	37	52	7	18	17	33	41	35	62	63	17
2	49	49	30	19	13	13	9	36	63	69	6
3	52	64	16	20	57	58	28	37	32	22	9
4	20	26	9	21	62	42	8	38	45	35	15
5	40	30	21	22	42	57	8	39	59	15	14
6	21	47	15	23	16	57	16	40	5	6	7
7	17	63	19	24	8	52	10	41	10	17	27
8	31	62	23	25	7	38	28	42	21	10	13
9	52	33	11	26	27	68	7	43	5	64	11
10	51	21	5	27	30	48	15	44	30	15	16
11	42	41	19	28	43	67	14	45	39	10	10
12	31	32	29	29	58	48	6	46	32	39	5
13	5	25	23	30	58	27	19	47	25	32	25
14	12	42	21	31	37	69	11	48	25	55	17
15	36	16	10	32	38	46	12	49	48	28	18
16	52	41	15	33	46	10	23	50	56	37	10
17	27	23	3	34	61	33	26				

Depot co-ordinates: (30, 40); customer demands (q) in cwt.
Truck capacity: 8 tons. No distance constraint on vehicle tours.

Details of problem 9

No.	x	y	q	No.	x	y	q	No.	x	y	q
1	22	22	18	26	41	46	18	51	29	39	12
2	36	26	26	27	55	34	17	52	54	38	19
3	21	45	11	28	35	16	29	53	55	57	22
4	45	35	30	29	52	26	13	54	67	41	16
5	55	20	21	30	43	26	22	55	10	70	7
6	33	34	19	31	31	76	25	56	6	25	26
7	50	50	15	32	22	53	28	57	65	27	14
8	55	45	16	33	26	29	27	58	40	60	21
9	26	59	29	34	50	40	19	59	70	64	24
10	40	66	26	35	55	50	10	60	64	4	13
11	55	65	37	36	54	10	12	61	36	6	15
12	35	51	16	37	60	15	14	62	30	20	18
13	62	35	12	38	47	66	24	63	20	30	11
14	62	57	31	39	30	60	16	64	15	5	28
15	62	24	8	40	30	50	33	65	50	70	9
16	21	36	19	41	12	17	15	66	57	72	37
17	33	44	20	42	15	14	11	67	45	42	30
18	9	56	13	43	16	19	18	68	38	33	10
19	62	48	15	44	21	48	17	69	50	4	8
20	66	14	22	45	50	30	21	70	66	8	11
21	44	13	28	46	51	42	27	71	59	5	3
22	26	13	12	47	50	15	19	72	35	60	1
23	11	28	6	48	48	21	20	73	27	24	6
24	7	43	27	49	12	38	5	74	40	20	10
25	17	64	14	50	15	56	22	75	40	37	20

Depot co-ordinates: (40, 40); customer demands (q) in cwt.
Truck capacity: 7 tons. No distance constraint on vehicle tours.

Details of problem 10

No.	x	y	q	No.	x	y	q	No.	x	y	q
1	41	49	10	35	63	65	8	68	56	39	36
2	35	17	7	36	2	60	5	69	37	47	6
3	55	45	13	37	20	20	8	70	37	56	5
4	55	20	19	38	5	5	16	71	57	68	15
5	15	30	26	39	60	12	31	72	47	16	25
6	25	30	3	40	40	25	9	73	44	17	9
7	20	50	5	41	42	7	5	74	46	13	8
8	10	43	9	42	24	12	5	75	49	11	18
9	55	60	16	43	23	3	7	76	49	42	13
10	30	60	16	44	11	14	18	77	53	43	14
11	20	65	12	45	6	38	16	78	61	52	3
12	50	35	19	46	2	48	1	79	57	48	23
13	30	25	23	47	8	56	27	80	56	37	6
14	15	10	20	48	13	52	36	81	55	54	26
15	30	5	8	49	6	68	30	82	15	47	16
16	10	20	19	50	47	47	13	83	14	37	11
17	5	30	2	51	49	58	10	84	11	31	7
18	20	40	12	52	27	43	9	85	16	22	41
19	15	60	17	53	37	31	14	86	4	18	35
20	45	65	9	54	57	29	18	87	28	18	26
21	45	20	11	55	63	23	2	88	26	52	9
22	45	10	18	56	53	12	6	89	26	35	15
23	55	5	29	57	32	12	7	90	31	67	3
24	65	35	3	58	36	26	18	91	15	19	1
25	65	20	6	59	21	24	28	92	22	22	2
26	45	30	17	60	17	34	3	93	18	24	22
27	35	40	16	61	12	24	13	94	26	27	27
28	41	37	16	62	24	58	19	95	25	24	20
29	64	42	9	63	27	69	10	96	22	27	11
30	40	60	21	64	15	77	9	97	25	21	12
31	31	52	27	65	62	77	20	98	19	21	10
32	35	69	23	66	49	73	25	99	20	26	9
33	53	52	11	67	67	5	25	100	18	18	17
34	65	55	14								

Depot co-ordinates: (35, 35); customer demands (q) in cwt.
Truck capacity: 10 tons. No distance constraints on vehicle tours.

REFERENCES

[1] G. B. DANTZIG and J. H. RAMSER (1959) The truck dispatching problem. *Mgmt Sci.* **6**, 80.
[2] M. HELD and R. M. KARP (1962) A dynamic programming approach to sequencing problems. *J. Soc. ind. appl. Math.* **10**, 196.
[3] R. L. HAYS (1967) The Delivery Problem. Carnegie Inst. of Tech., Graduate School of Industrial Admin., Report MSR 106.
[4] J. D. C. LITTLE, K. G. MURTY, D. W. SWEENEY and C. KAREL (1963) An algorithm for the travelling salesman problem. *Ops Res.* **11**, 979.
[5] J. B. KRUSKAL (1956) On the shortest spanning subtree of a graph. *Proc. Am. Math. Soc.* 7, 48.
[6] G. CLARKE and J. W. WRIGHT (1963) Scheduling of vehicles from a central depot to a number of delivery points. *Ops Res.* **11**, 568.
[7] M. FLOOD (1956) The travelling salesman problem. *Ops Res.* **4**, 61.
[8] G. A. CROES (1958) A method of solving travelling salesman problems. *Ops Res.* **5**, 791.
[9] S. LIN (1965) Computer solutions of the travelling salesman problem. *Bell Syst. tech. J.* **44**, 2245.
[10] T. J. GASKELL (1967) Bases for vehicle fleet scheduling. *Opl Res. Q.* **18**, 281.

References

[5] **On the Shortest Spanning Subtree of a Graph and the Traveling Salesman Problem**
Joseph B. Kruskal, Jr.
Proceedings of the American Mathematical Society, Vol. 7, No. 1. (Feb., 1956), pp. 48-50.
Stable URL:
http://links.jstor.org/sici?sici=0002-9939%28195602%297%3A1%3C48%3AOTSSSO%3E2.0.CO%3B2-M