45

J. Opl Res. Soc. Vol. 44, No. 3, pp. 289-296
Printed in Great Britain. All rights reserved

0160-5682/93 $8.00+0.00

Extensions of the Petal Method for Vehicle Routeing

DAVID M. RYAN[1], CURT HJORRING[1] and FRED GLOVER[2]
[1]Department of Engineering Science, University of Auckland, New Zealand, and [2]University of Colorado at Boulder, USA

The petal method for vehicle routeing imposes special structure on the form of a feasible route. In this paper we show that by extending the definition of a petal route, more general forms of vehicle route can be generated without invalidating the important underlying property that optimal petal solutions can be produced very easily. It will also be shown that the optimal generalized petal solution can be produced efficiently by multiple applications of a shortest path algorithm.

Key words: distribution, networks and graphs, shortest path, travelling salesman, vehicle-routeing

INTRODUCTION

The Vehicle Routeing Problem (VRP) and the related Vehicle Scheduling Problem (VSP) have been the subject of extensive research due to their practical importance in the transportation industry (Bodin *et al.*[1] and Laporte and Norbert[2]). The VRP involves the construction of vehicle routes to visit a number of delivery points from a central depot. The VSP also includes temporal aspects associated with time windows for visits which impose extra structure on the vehicle routes. The problems are NP-hard, so much of the research has centred on the development of heuristics.

A special class of heuristics that provides the focus of this paper involves the construction of routes with a restricted petal-like structure. Petal routes visit all delivery points in a geographic sector centred on the depot. In other words, no customer is left unvisited in the sector. This restriction on the structure of a route reflects the observation that in many problems optimal routes exhibit a petal or near petal structure.

An early attempt to exploit this restricted structure was carried out by Gillet and Miller[3] with a heuristic procedure called the sweep method. A subsequent advance was then provided by Foster and Ryan[4] who showed that an optimal petal solution can easily be found from the set of all possible petal routes by solving a linear programme in which, under a mild assumption, all basic feasible solutions are naturally integer. They also showed that some subsequent relaxations of the strict petal route structure could be considered although the inclusion of non-petal routes invalidated the natural integer properties of the LP making it more difficult to find optimal integer solutions.

Building on the work of Foster and Ryan, this paper introduces a generalized framework that permits consideration of non-petal routes without destroying the natural integer properties of the underlying linear programme. This framework provides a basis for theorems establishing that an optimal generalized petal solution can be found efficiently by solving a small number of shortest path problems in an underlying directed graph representation of the set of all generalized petals. In addition, we provide a result that allows the graph to be progressively reduced to yield further efficiency gains. These outcomes are susceptible to exploitation by either branch-and-bound or meta-level heuristics such as tabu search, and are also applicable to settings beyond vehicle routeing.

AN EXAMPLE OF THE PETAL METHOD

The concept of the petal method will be illustrated using the problem shown in Figure 1. Each of the 13 delivery points has an associated delivery demand shown in brackets. The distances

Correspondence: D. M. Ryan, Department of Engineering Science, University of Auckland, Auckland, New Zealand

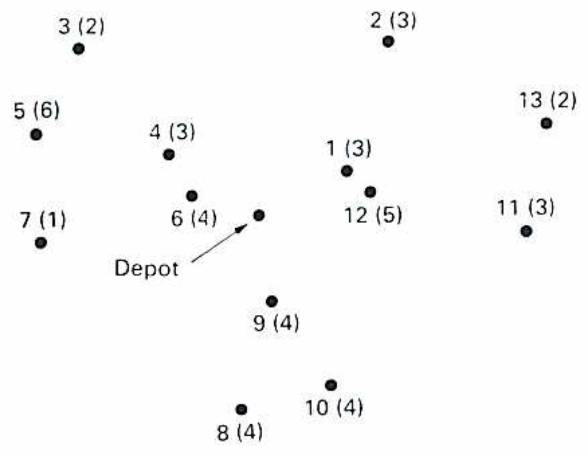

FIG. 1. *Example problem.*

between delivery points are Euclidean and all distance calculations are performed using real arithmetic. The goal is to minimize the number of vehicles required to deliver from a central depot and, for that number of vehicles, to minimize the total distance travelled. Each vehicle has a capacity of ten units of demand. The coordinates for the depot and the delivery points are given in the appendix.

The delivery points are numbered in radial order about the depot. More precisely, consider the ordering that results by taking the depot as the centre of a circle (for convenience, one that circumscribes the points), and consider the projection of each point onto the circle created by the endpoint of the radius that passes through the point. Starting with an arbitrary (projected) point on the circle as point 1, and then sweeping around the circle in a chosen direction (counter-clockwise in Figure 1) each point is assigned a number from 1 to n representing the sequence in which it is encountered. Tied orderings, where more than one delivery has the same point of projection, are broken arbitrarily, say by assigning the lower number to the delivery closest to the depot. In the example there are no ties and the ordering is unique. Each radially contiguous subset taken from this ordering is a *petal*.

The enumeration of the set of all petals can be implemented very efficiently as discussed by Foster and Ryan[4]. A petal is feasible if the quantity of goods delivery on the route does not exceed the capacity of the vehicle and if the total distance travelled, as determined by the Travelling Salesperson sequence of the deliveries, does not exceed the imposed distance limit. We will call such a TSP route for a petal a *petal route*, and refer to it as feasible if the petal itself is feasible. An example of a feasible petal route is shown in Figure 2(a). The petal route in Figure 2(b) is infeasible because the sum of customer deliveries (12) exceeds the capacity of the vehicle (10). The route in Figure 2(c) is not a petal route because the deliveries do not form a radially contiguous subset. However, if delivery 1 is added to the subset, filling in the gap to ensure the points are contiguous, the resulting TSP tour on this larger subset would then become a petal route.

Given the deliveries in the radial order 1, 2, 3, . . ., 13, the enumerated subsets of contiguous deliveries corresponding to all feasible petals are given in Table 1.

It should be noted that petal generation treats the order as cyclic in that the subsets beginning with deliveries 12 and 13 wrap around to include the deliveries at the beginning of the order. For

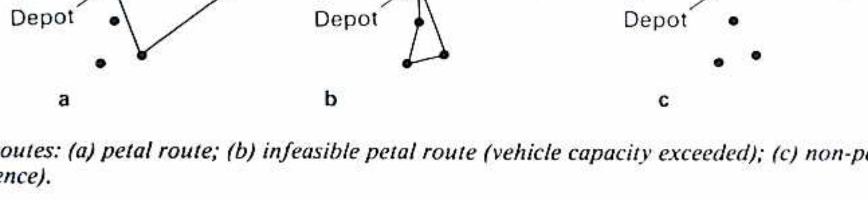

FIG. 2. *Example routes: (a) petal route; (b) infeasible petal route (vehicle capacity exceeded); (c) non-petal route (gap for point out of sequence).*

TABLE 1. *Contiguous subsets from the natural radial order*

(1)	(1,2)	(1,2,3)	
(2)	(2,3)	(2,3,4)	
(3)	(3,4)		
(4)	(4,5)		
(5)	(5,6)		
(6)	(6,7)	(6,7,8)	
(7)	(7,8)	(7,8,9)	
(8)	(8,9)		
(9)	(9,10)		
(10)	(10,11)		
(11)	(11,12)	(11,12,13)	
(12)	(12,13)	(12,13,1)	
(13)	(13,1)	(13,1,2)	(13,1,2,3)

this reason we will refer to orders as *cyclic orders*. The cost of a petal is defined to be the cost of the corresponding TSP petal route, which is typically, though not necessarily, proportional to the distance travelled. It is important to keep in mind that this cost is not known directly from the identity of the petal itself, except by first applying a TSP procedure (Lawler *et al.*[5]).

It should also be noted that the delivery sequence in the petal route determined by the TSP solution will differ in general from the cyclic order, the order used to generate the elements of the associated subset. For example, the petal (11, 12, 13) generates a petal route (depot-11-13-12-depot). Each petal route is either a single delivery or a route which results from the addition of the next delivery in the order to the subset of deliveries considered on the previous petal route. Given this sequential building process, it is possible to develop efficient heuristic TSP algorithms which add the extra delivery to the previous route and then attempt to improve the TSP solution.

We define a *spanning petal* set to be a collection of petals that contains every delivery point exactly once. For a given cyclic order a spanning petal set can be conveniently represented by a sub-sequence of deliveries taken from the cyclic order. Each delivery in the sub-sequence is the first delivery from each petal. For instance, the set of petals {(**9**, 10), (**11**, 12), (**6**, 7, 8), (**4**, 5), (**13**, 1, 2, 3)} from Table 1 is a spanning petal set and can be written as {4, 6, 9, 11, 13}. It is evident that the problem of identifying an optimal set of routes from the set of petals, can be solved by restricting attention to finding a minimum cost spanning petal set which we will call a minimum spanning set or an optimal petal solution. The optimal petal solution can be easily determined once the cost of each petal is obtained. Foster and Ryan[4] proposed the formulation of a set partitioning model and the use of the LP simplex method for this purpose, but in the next section we discuss an alternative and more efficient shortest path technique for producing the optimal petal solution.

The optimal petal solution, shown in Figure 3, requires five vehicle routes and a total distance of 68.74. It is clear however that when more general non-petal forms of vehicle route are permitted, an improved solution with total distance of 60.47 can be found as shown in Figure 4.

A first key contribution of this paper is to demonstrate that the petal method outlined above can be used to produce the optimal solution shown in Figure 4. All that is required is to reorder the

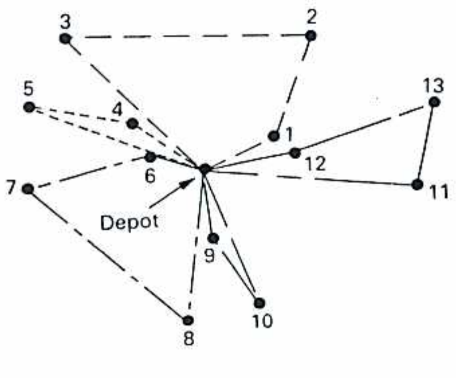

FIG. 3. *Optimal petal solution.*

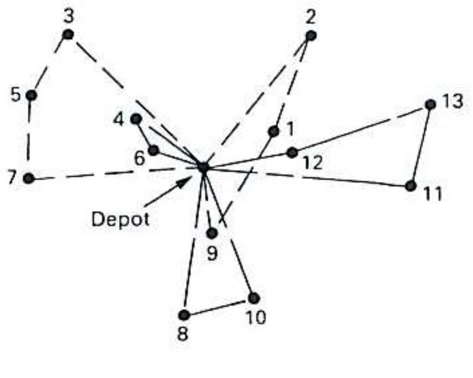

FIG. 4. *Optimal solution.*

deliveries in a non-radial cyclic order before applying the petal generation scheme. The sets generated will be referred to as *generalized petals*. In geographic terms, the sets will no longer be pure petals but will involve contiguous subsets of deliveries from the non-radial cyclic order. For example, consider the deliveries in the cyclic order 3, 7, 5, 8, 10, 6, 4, 2, 9, 1, 12, 13, 11. Given this order, the set of all subsets of contiguous deliveries corresponding to feasible generalized petals are given in Table 2.

TABLE 2. *Contiguous subsets for the cyclic order (3,7,5,8,10,6,4,2,9,1,12,13,11)*

(3)	(3,7)	**(3,7,5)**	
(7)	(7,5)		
(5)	(5,8)		
(8)	**(8,10)**		
(10)	(10,6)		
(6)	**(6,4)**	(6,4,2)	
(4)	(4,2)	(4,2,9)	
(2)	(2,9)	**(2,9,1)**	
(9)	(9,1)		
(1)	(1,12)	(1,12,13)	
(12)	(12,13)	**(12,13,11)**	
(13)	(13,11)	(13,11,3)	(13,11,3,7)
(11)	(11,3)	(11,3,7)	

It can be seen that the five subsets making up the optimal solution of Figure 4 are contained within the set of generalized petals and the optimal solution corresponds to the minimum spanning set $\{3, 8, 6, 2, 12\}$. From an optimization point of view, the determination of an optimal generalized petal solution from the set of all generalized petals is no more difficult than the determination of the optimal petal solution from the set of all petals. The shortest path method discussed in the next section can also be applied to the set of generalized petals to find the optimal solution.

It should be noted that there are many different cyclic orders of deliveries which contain the optimal routes of Figure 4 within the corresponding sets of generalized petals. It is sufficient to find any cyclic order in which the deliveries on each route in the optimal solution are contiguous. The order of the deliveries within each contiguous subset and the order of the subsets within the cyclic order is unimportant. For example, an alternative cyclic order of 1, 2, 3, 5, 7, 4, 6, 8, 10, 11, 12, 13, 9, which is a small perturbation of the natural radial order, will also produce the optimal solution of Figure 4. From this discussion we see that there exists a family of 'optimal cyclic orders' which are equivalent in the sense that any members of the family will produce the optimal generalized petal solution. Provided the optimal generalized petal solution can be found efficiently for a given cyclic order, the optimal solution of the vehicle routeing problem can be found by examining a subset of permutations of the cyclic order (although the identity of this subset is not known in advance). To guarantee optimality this subset might need to be large. However, good quality solutions should be generated by considering a smaller subset of permutations of the radial order. Given the radial structure underlying most optimal solutions, it is likely that such a subset would contain a member of the family of optimal cyclic orders. We are currently investigating tabu search techniques to control the generation of cyclic order permutations. In the next section we discuss an efficient method for determining the optimal petal solutions.

OPTIMAL PETAL SELECTION

Given a set of petals, Foster and Ryan[4] select an optimal subset by formulating the problem as a set partitioning problem in which each row of the constraint matrix corresponds to a delivery point and each column to a petal. Foster and Ryan show that for petals, the LP relaxation of the set partitioning problem is totally unimodular and the optimal integer solution can be found by just solving the LP. By reordering the constraints to reflect the chosen cyclic order the proof used by Foster and Ryan can be easily extended to show that the set partitioning problem for generalized petals remains totally unimodular. While the LP method has the advantage that extra constraints can be added to model an inhomogeneous fleet or a multi-period problem (Pedder and Philpott[6]), the LP solution time remains the major bottleneck.

A second key contribution of this paper is to identify a more efficient method for selecting optimal petals (and hence their associated routes). The petals are represented by a weighted cyclic digraph $G = (N, A)$ as follows. The nodes N of the digraph correspond to deliveries. The arcs A correspond to generalized petals. Each generalized petal is represented by an arc from node i to node j, where node i is the first delivery point in the petal (in relation to the cyclic order) and node j is the first subsequent delivery from the cyclic order that is not on that petal. For example the petal (11, 12, 13) in Table 1 is represented by the arc from node 11 to node 1 as shown in Figure 5(a). The cyclic petal digraphs corresponding to the petals of Table 1 and the generalized petals of Table 2 are shown in Figures 5(a) and 5(b) respectively.

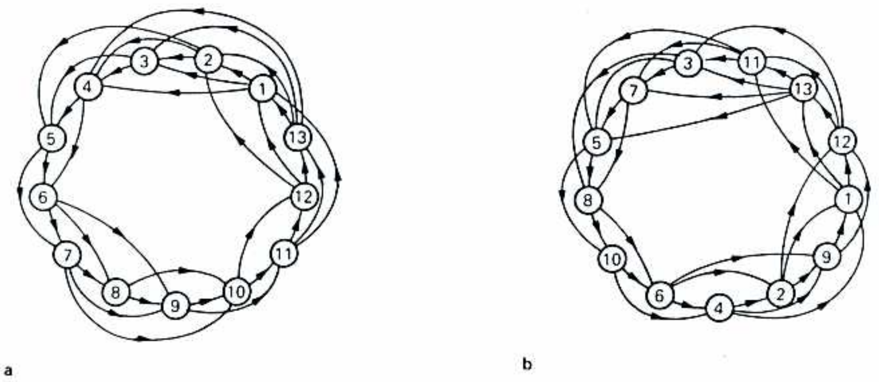

FIG. 5. *Cyclic petal digraphs. (a) For petals of Table 1. (b) for genealized petals of Table 2.*

Hereafter, because our representation of the petals as a digraph applies to generalized petals as well as petals, we will drop the adjective 'generalized' and understand the word 'petal' to refer to the general case. In standard terminology, a (non-degenerate, node simple) cycle is a path that contains at least one arc and returns to its point of origin without duplicating any nodes except the starting point. It the context of the petal digraph, we introduce the notion of a special type of cycle, called a *compact cycle*, defined to include the following property: for each arc (i, j) of the cycle, there is no node of the cycle that lies strictly between nodes i and j in the cyclic order.

For example, the cycle $\{1, 4, 6, 9, 11\}$ in Figure 5(a) (described by naming the sequence of nodes visited but omitting the final node) corresponds to the spanning petal set (1, 2, 3), (4, 5), (6, 7, 8), (9, 10), (11, 12, 13). The cycle notation $\{1, 4, 6, 9, 11\}$ also represents the spanning petal set as discussed in the previous section. This relationship between spanning petal sets and compact cycles is formalized by the following theorem.

Theorem 1

There is a one to one correspondence between spanning petal sets and compact cycles.

Proof

For a given spanning petal set, each delivery (node in the cyclic digraph) occurs on exactly one of the petals in the spanning petal set. The arcs of the digraph corresponding to the petals of the spanning set therefore form a compact cycle. Conversely, for a given compact cycle, the arcs correspond to petals which have the property that they include all deliveries and no delivery occurs on more than one petal. The petals therefore form a spanning petal set.

Thus, the problem of finding the optimal petal solution (i.e. minimum spanning petal set) can be considered as the problem of finding the shortest compact cycle in the petal digraph.

We now consider the way in which such a shortest compact cycle can be identified. If we restrict our attention to only those cycles that contain node k then all arcs that bypass k can be ignored.

By splitting node k into an origin node k_o, and a destination node k_d, where k_o initiates all arcs out of k and k_d receives all arcs into k, the cyclic petal digraph becomes an acyclic digraph. This acyclic digraph can be formed by the following algorithm.

Acyclic digraph induced by node k

(1) Split node k into an origin node k_o, and a destination node k_d, where k_o initiates all arcs out of k and k_d receives all arcs into k.
(2) Let $p(h)$ denote the sequential position, as determined by the cyclic order, of each node h in the graph created from G by Step (1), where $p(\text{origin}) = 1$ and $p(\text{destination}) = n + 1$, and delete each arc (i, j) such that $p(i) > p(j)$ or $i = j, i \neq k$.

An example of the acyclic digraph from Figure 5(a) induced by node 1 is shown in Figure 6.

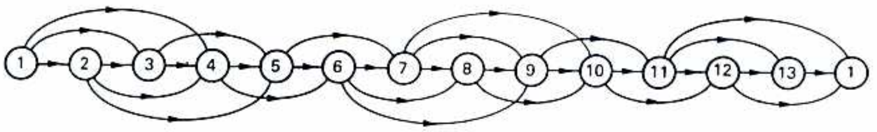

FIG. 6. *Acyclic digraph induced by node 1.*

The shortest compact cycle that contains node k can be generated as the shortest path from k_o to k_d in the acyclic digraph induced by node k. The acyclic property of each induced digraph enables a shortest path to be found by a very simple and quick shortest path (SP) algorithm, which simply scans each node in the sequence from origin to destination exactly once, in the cyclic order. Since a compact cycle must contain at least one node, the shortest compact cycle can be found by finding the shortest path from k_o to k_d in the n acyclic digraphs induced by each node in turn. For most vehicle routeing problems, however, it is impossible to satisfy all deliveries in one route. This implies that the shortest compact cycle can be found by solving a smaller number of shortest path problems. The actual number of shortest path problems required is established by the following argument.

First, assume that it is impossible to satisfy all deliveries in one route. Let the cyclic order be represented by a function $s(i)$ where $s(i)$ is the node following i in the cyclic order. Also denote $s^2(i) = s(s(i)), s^j(i) = s(s^{j-1}(i))$ and $s^{-1}(j) = i$ where $s(i) = j$.

When we restrict our attention to compact cycles containing node k we ignore arcs that correspond to petals containing both deliveries k and $s^{-1}(k)$. Let P_k be the proposition that in an optimal solution, deliveries k and $s^{-1}(k)$ are in the same petal. Let $\overline{P_k}$ denote the converse, that in an optimal solution, deliveries k and $s^{-1}(k)$ are not in the same petal. If $\overline{P_k}$ is true then the optimal solution can be found by a shortest path calculation on the acyclic digraph induced by node k. More generally if deliveries $k, s(k), \ldots, s^j(k)$ are on the same petal then $P_{s(k)}$ & $P_{s^2(k)}$ & ... $P_{s^j(k)}$ is true.

Theorem 2. (Number of shortest paths required)

If arc $a_{i,s^{j+1}(i)}$ does not exist then the optimal petal solution can be found in j shortest paths.

Proof

Since arc $a_{i,s^{j+1}(i)}$ does not exist there is no feasible petal which contains $i, s(i), s^2(i), \ldots, s^j(i)$ so the statement $\overline{(P_{s(i)} \,\&\, P_{s^2(i)} \,\&\, \ldots \,\&\, P_{s^j(i)})}$ is true. Therefore, $\overline{P_{s(i)}} \mid \overline{P_{s^2(i)}} \mid \ldots \mid \overline{P_{s^j(i)}}$ is true. We can investigate each of these j cases separately and this involves finding j shortest paths. The minimum of these j shortest paths gives the optimal petal solution.

To find the optimal petal solution in a minimum number of shortest paths we must find that delivery i for which $j \in N$ is minimized and $a_{i,s^{j+1}(i)} \notin A$. Then, exactly j shortest path calculations will be required. In practice it is sufficient to find a delivery i which seeds the fewest number of petals of the form (i), $(i, s(i))$, $(i, s(i), s^2(i))$, ..., $(i, s(i), \dots, s^{j-1}(i))$ and then find shortest paths in the acyclic digraphs induced by nodes $s(i), s^2(i), \dots, s^j(i)$.

It is also possible to improve the efficiency of solving the shortest path problems by means of the following theorem.

Theorem 3

After solving the problem on the digraph induced by k, the following two reductions of G are admissible as a basis for defining all induced digraphs subsequently to be examined.

(1) Delete node k and its incident arcs from G.

(2) Successively delete all nodes of G, and their remaining incident arcs, whose set of entering arcs or whose set of leaving arcs becomes empty either as a result of Step (1) or as a result of this step applied to other nodes.

Proof

Reduction 1 is correct because after solving the problem we have examined all compact cycles containing node k. If there is any shorter compact cycle it will not contain node k and thus any of its incident arcs. Reduction 2 is correct since a node with no entering arcs can never be reached and a node with no leaving arcs can never be left. Therefore, the node cannot be part of a cycle.

In the next section we will show that the shortest path method for finding a shortest compact cycle is significantly faster than the LP method for finding an equivalent optimal petal solution. However, a disadvantage of the shortest path method is that the side constraints associated with an inhomogeneous fleet or a multi-period problem become more difficult to accommodate. These difficulties are currently under further investigation.

RESULTS

The performance of the shortest path method for finding a shortest compact cycle and the LP method for finding an equivalent optimal petal solution have been compared by solving the problems defined in Altinkemer and Gavish[7]. The natural radial order was used to define the petal set for each problem. The ZIP package[8], which is known to be very efficient for solving LP relaxations of set partitioning problems, was used to solve the petal LP and the shortest compact cycles were found by an implementation of the method discussed in this paper.

The tests were performed on a SGI 4D/240 and the times given exclude the time to generate

TABLE 3. *Comparison of the shortest path and LP methods*

Problem	Number of deliveries	Number of routes generated	Number of vehicles in solution	Time in SP (CPU in hundredths of a second)	Time in LP (CPU in hundredths of a second)
p1	50	495	5	0.23	41
p2	75	548	11	0.23	78
p3	100	1321	8	0.93	300
p4	150	1961	12	1.34	622
p5	199	2412	17	1.63	1068
p6	50	437	6	0.21	46
p7	75	486	12	0.23	67
p8	100	1140	9	0.82	275
p9	150	1568	15	1.02	476
p10	199	2148	19	1.42	784
p11	120	2050	7	2.29	586
p12	100	1077	10	0.72	255
p13	120	1258	12	0.92	266
p14	100	968	11	0.63	185

the feasible petal routes. The results given in Table 3 show conclusively that the shortest path method outperforms the LP method by two orders of magnitude. An additional advantage of the shortest path method is that the computer code is simpler to write than for the LP method.

SUMMARY

In this paper we have shown that;

(1) the optimal solution of a VRP corresponds to a family of 'optimal cyclic orders';

(2) for a given cyclic order the optimal generalized petal set can be determined very efficiently.

This raises the possibility of a new method for the VRP which starts with an initial cyclic order. The cyclic order is then modified so as to improve the objective. The modification of the cyclic order could be guided by search methods such as simulated annealing or tabu search. Results from an initial version of the method using tabu search are encouraging and indicate that it should efficiently produce good quality solutions to the VRP. Although we have used the VRP to illustrate the concepts of this approach the ideas can also be extended to include more complex problems such as the VSP and the multiple depot vehicle routeing problem.

APPENDIX

Data for example problem

Vehicle capacity = 10

Delivery Number	X	Y	Order Size
1	10.0	−5.0	3
2	11.0	−2.0	3
3	4.0	−2.0	2
4	6.0	−4.5	3
5	3.0	−4.0	6
6	6.5	−5.5	4
7	3.0	−6.5	1
8	7.5	−10.5	4
9	8.4	−8.0	4
10	9.5	−10.0	4
11	14.0	−6.5	3
12	10.5	−5.5	5
13	14.5	−4.0	2

The depot is located at 8.0, −6.0.

REFERENCES

1. L. Bodin, B. Golden, A. Assad and M. Ball (1983) Routing and scheduling of vehicles and crews: the state of the art. *Comps Opns Res.* **10**, 63–211.
2. G. Laporte and Y. Nobert (1987) Exact algorithms for the vehicle routing problem. *Ann. Discrete Math.* **31**, 147–184.
3. B. Gillet and L. Miller (1974) A heuristic algorithm for the vehicle-dispatch problem. *Opns Res.* **22**, 340–349.
4. B. A. Foster and D. M. Ryan (1976) An integer programming approach to the vehicle scheduling problem. *Opl Res. Q.* **27**, 367–384.
5. E. L. Lawler, J. K. Lenstra, A. K. G. Rinnooy Kan and D. B. Shmoys (Eds) (1985) *The Traveling Salesman Problem: A Guide Tour of Combinatorial Optimization.* Wiley, Chichester.
6. S. J. Pedder and A. B. Philpott (1989) A vehicle routing model based on set-partitioning. Presented at the *25th annual conference of the Operational Research Society of New Zealand*, Wellington, August 1989.
7. K. Altinkemer and B. Gavish (1991) Parallel savings based heuristics for the delivery problem. *Opns Res.* **39**, 456–469.
8. D. M. Ryan (1980) ZIP—a zero-one integer programming package for scheduling. Research Report. #CSS85, AERE, Harwell.