Solving a Family of Multi-Depot Vehicle Routing and Location-Routing Problems

GILBERT LAPORTE*

Université de Montréal, Montréal, Québec, Canada

YVES NOBERT†

Université du Québec à Montréal, Montreal, Québec, Canada

SERGE TAILLEFER*

Université de Montréal, Montreal, Québec, Canada

This paper examines a class of asymmetrical multi-depot vehicle routing problems and location-routing problems, under capacity or maximum cost restrictions. By using an appropriate graph representation, and then a graph extension, the problems are transformed into equivalent constrained assignment problems. Optimal solutions are then found by means of a branch and bound tree. Problems involving up to 80 nodes can be solved without difficulty.

The *vehicle routing problem* (VRP) is commonly defined as the problem of designing optimal delivery or collection routes from one or several depots to a set of geographically scattered customers, under a variety of side conditions. *Location-routing problems* (LRPs) are VRPs in which the optimal depot locations and route design must be decided simultaneously. Both the VRP and LRPs have been studied extensively in the last few years. For recent surveys, see [12] for the VRP and [8] for LRPs.

In order to define these problems in mathematical terms, consider a graph $G = (N, N^2, C)$ where $N = \{1, \ldots, n\}$ is a set of *nodes*, N^2 is a set of *arcs* and $C = (c_{ij})$ is a matrix of costs, distances or travel times associated with N^2. Since these three measures are in most practical situations strongly correlated, we will interpret in this paper the c_{ij}'s and all distance and time constraints imposed on the vehicle routes in terms of *costs*. If $c_{ij} = c_{ji}$ for all $i, j \in N$, the matrix (and the problem) is said to be symmetrical; otherwise, it is asymmetrical. C satisfies the triangle inequality if and only if $c_{ik} + c_{kj} \geqslant c_{ij}$ for all $i, j, k \in N$. N is partitioned into $\{R, N - R\}$ where $R = \{l, \ldots, r\}$ is a set of *potential depots* and $N - R$ represents *customers*.

* Centre de recherche sur les transports, Université de Montréal, C.P. 6128, Succursale A, Montréal H3C 3J7, Canada.

† Département des Sciences administratives, Université du Québec à Montréal, 1495, rue Saint-Denis, Montréal H3C 3P8, Canada.

There can be at most $\bar{m}_k$ identical vehicles of capacity D based at depot k and the fixed cost of using a vehicle is equal to g. The cost of operating depot k is equal to h_k. It is assumed that the c_{ij}'s, g and the h_k's are non-negative and calibrated so that they relate to the same planning horizon. Every customer $j \in N - R$ has a non-negative demand d_j.

There exist several variants of these problems. We consider three cases in this paper

(i) *Capacity constrained VRPs*: all nodes of R are used as depots; the problem consists of establishing vehicle routes from these depots in such a way that
  a) each customer is visited once and exactly once by one vehicle;
  b) all vehicle routes start and end at the same depot;
  c) vehicle capacities are never exceeded;
  d) the number of vehicles based at each depot lies within prespecified bounds;
  e) the sum of vehicle costs and routing costs is minimized.

(ii) *Cost constrained VRPs*: these problems are defined by replacing constraint c) of (i) by
  f) the cost of a vehicle route (including the vehicle fixed cost) does not exceed a prespecified limit L.

Transportation Science
Vol. 22, No. 3, August 1988

0041-1655/88/2203-0161 $01.25

(iii) *Cost constrained LRPs*: here, depot sites must be selected in R and vehicle routes satisfying a), b), d), f) must be established simultaneously in order to minimize the sum of vehicle costs, depot operating costs and routing costs.

Note that a special case occurs when there is only one prespecified depot, only one vehicle, and no capacity or maximum cost constraints: the problem then reduces to a *traveling salesman problem* (TSP) on which an extensive literature exists (see for example [16]).

Applications, formulations and approximate algorithms for LRPs under capacity and maximum cost restrictions have been described by GOLDEN et al.,[7] OR and PIERSKALLA,[20] SRIKAR and SRIVASTAVA[22] and PERL and DASKIN[21] but, as far as the authors are aware, no exact method has ever been provided for this general case. The only exact algorithms for these problems appear to be those of LAPORTE et al. who have solved to optimality symmetrical capacity constrained LRPs[14] and symmetrical multi-depot VRP.[13]

Most studies related to this family of problems have been carried out in the context of the single depot VRP. Literature on this problem is rather abundant but relatively little headway toward the solution of the problem by exact methods has been made. The most promising approaches appear to be

(i) The use of dynamic programming with state-space relaxation[3, 4] in the case of tightly constrained problems (CHRISTOFIDES[4] reports that capacity constrained VRPs of up to 50 cities can be solved easily using this approach);
(ii) The use of integer linear programming (ILP) in conjunction with constraint relaxation in the case of loosely constrained symmetrical problems[15] (VRPs under capacity and distance restrictions were solved exactly for sizes of up to $n = 60$[15]);
(iii) Branch and bound methods in which every subproblem is an assignment problem (AP); these methods are most appropriate to the solution of asymmetrical loosely constrained problems[10, 11] (in the case of single depot capacitated VRPs, problems involving up to 100 nodes were solved to optimality by LAPORTE et al.[10]).

Heuristic methods, in contrast, can solve much larger problems and often take into account a greater variety of constraints. A detailed account of such methods can be found in the 1983 survey by BODIN et al.[1] Since then, a number of interesting developments have occurred. These are mostly related to the increase in the number of successful implementations of computerized routing systems and in the added realism of mathematical models (GOLDEN and ASSAD[6]). On the algorithmic side, NELSON et al.[18] have shown how algorithms such as that of CLARKE and WRIGHT[5] can be vastly improved by carefully exploiting data structures and computer science techniques. The authors show that the time required to run the Clarke and Wright algorithm on 1000 city problems can be reduced to between 175 and 200 seconds on an IBM 4341 computer, using a Pascal 8000 compiler.

We consider in this paper three types of multi-depot asymmetrical problems. We show how, after an appropriate graph transformation, they can be formulated in a unified way as constrained APs which are then solved by adapting the CARPANETO and TOTH[2] branch and bound procedure for the asymmetrical TSP. In this algorithm, all subproblems are APs and can therefore be solved efficiently. The various side constraints are taken into account in the branching process, by fixing at zero or at one some of the variables, thereby preserving the AP structure. When the relaxed constraints are not too stringent, the expansion of the search tree is limited and the procedure is highly efficient. With this approach, problems of relatively large sizes can be solved to optimality.

The three cases under study will henceforth be treated as *one* problem possessing several variants. The graph transformation, formulation, algorithm and computational results respectively are presented in the next four sections.

1. GRAPH TRANSFORMATION

GRAPH TRANSFORMATIONS have been used by a number of authors to solve several TSP extensions (see for example [10, 11, 17 and 19] for a survey). These transformations are such that every feasible solution to the original problem corresponds to one or more Hamiltonian circuits on a subgraph of the new graph. Any feasible solution on the transformed graph can be interpreted as a solution to the original problem on the original graph. In order to achieve the desired transformation for our problem, we proceed as follows.

Graph G is transformed into $G' = (N', A', C')$. The new set of nodes N' is the union of three pairwise disjoint sets of nodes:

(i) A singleton M consisting of a dummy node: $M = \{0\}$;
(ii) A set R' consisting of the union of $|R|$ sets of nodes R_k $(k = 1, \ldots, |R|)$ where each R_k corresponds to a depot and contains as many nodes as one plus the maximum number of vehicles based at that facility: $R_k = \{r_{ks}: s = 1, \ldots, \bar{m}_k + 1\}$;
(iii) The set $N - R$ of customers.

The set of nodes can be partitioned into $N' = \{N_1', N_2'\}$ where N_1' corresponds to *specified nodes*, i.e., nodes which must necessarily appear in the optimal solution and N_2' corresponds to *unspecified nodes*, i.e., nodes which may or may not be used. Formally, N_1' is defined as $N_1' = M \cup (N - R) \cup (\bigcup_{k \in R^*} R_k)$ where R^* is the index set of depots which must be included in the solution. Since in the VRP, all depots are used, $R^* = R$ and $N_1' = N'$. In both the VRP and the LRP, N_2' is equal to $N' - N_1'$.

The new set of arcs A' can be defined as $A' = \bigcup_{t=1}^{8} A_t$ where

$A_1 = M \times \{r_{k,1}: k = 1, \ldots, |R|\}$ is a set of arcs from the dummy node to the first node of every potential depot;

$A_2 = \{r_{k,\bar{m}_k+1}: k = 1, \ldots, |R|\} \times M$ is a set of arcs from the last node of every potential depot to the dummy node;

$A_3 = \{r_{ks}: k = 1, \ldots, |R|; s = 1, \ldots, \bar{m}_k\} \times (N - R)$ is a set of arcs from potential depots to customers;

$A_4 = (N - R) \times \{r_{ks}: k = 1, \ldots, |R|; s = 2, \ldots, \bar{m}_k + 1\}$ is a set of arcs from customers to potential depots;

$A_5 = \bigcup_{k=1}^{|R|} \bigcup_{s=1}^{\bar{m}_k} \{(r_{ks}, r_{k,s+1})\}$ is a set of arcs connecting successive nodes within each potential depot;

$A_6 = \bigcup_{k,k'=1,k\neq k'}^{|R|} \{(r_{k,\bar{m}_k+1}, r_{k',1})\}$ is a set of arcs connecting different potential depots;

$A_7 = \{(r_{ks}, r_{ks}): k = 1, \ldots, |R|; s = 1, \ldots, \bar{m}_k + 1\}$ is a set of loops associated with potential depot nodes;

$A_8 = \{(i, j): i, j \in N - R, i \neq j\}$ is a set of inter-customer arcs.

The cost matrix $C' = (c'_{uv})$ associated with A' is defined by the following formulas:

$$c'_{uv} = \begin{cases} h_k & (u, v) \in A_1 \cup A_6, \\ & \qquad v = r_{k,1} \\ g + c_{kv} & (u, v) \in A_3, \quad u = r_{ks} \\ & \qquad \text{for some } k, s \\ c_{uk} & (u, v) \in A_4, \quad v = r_{ks} \\ & \qquad \text{for some } k, s \\ c_{uv} & (u, v) \in A_8 \\ e_k & (u, v) = (r_{ks}, r_{k,s+1}) \in A_5 \\ 0 & (u, v) \in A_2 \cup A_7 \\ \infty & \text{otherwise} \end{cases} \tag{1}$$

where, as in [17],

if $e_k = \infty$, all $\bar{m}_k$ vehicles based at potential depot k will be used if the depot is open;

if $e_k = -\infty$, the minimum possible number of vehicles based at potential depot k will be used;

if $e_k = 0$, the number of vehicles used at potential depot k is unspecified; it may take any value between 1 and $\bar{m}_k$.

Before proceeding to the problem formulation, it appears useful to provide an interpretation of this transformation. First consider the VRP solution depicted by Figure 1. Here, there are three depots, with a maximum of two vehicles each. All nodes are specified, i.e., $N_1' = N'$. Figure 1 shows that a cost of h_k is incurred whenever depot k is entered and a cost of g added to the basic travel cost whenever a vehicle leaves a depot for a customer. Arcs connecting two nodes within the same potential depot correspond to unused

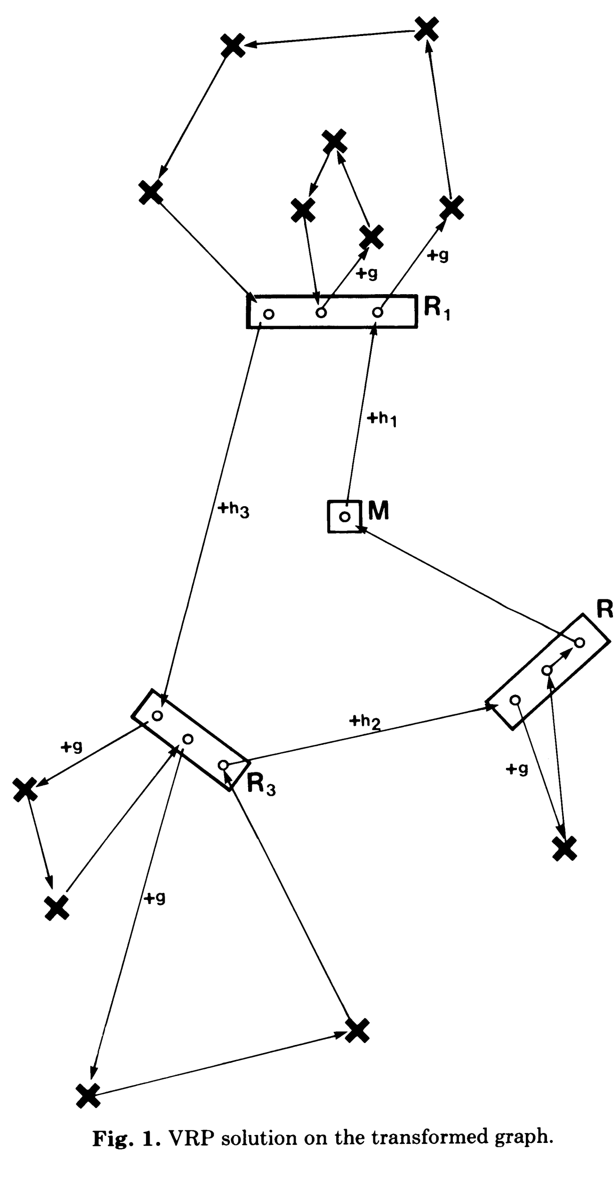

Fig. 1. VRP solution on the transformed graph.

vehicles. Note that the Hamiltonian circuit starting and ending at the second node of the first depot is legal since it corresponds to a feasible vehicle route. Also observe that in this problem, the dummy node is not really necessary since there exists an equivalent solution such that all depots are visited and in which depot 1 is entered from depot 2. Now consider the LRP solution represented by Figure 2. Here, only two potential depots (1 and 3) are used. The fixed costs g and h_k must be interpreted as in the previous example. The loops around the nodes of potential depot 2 indicate that this depot is not used in the solution. The usefulness of these loops will become explicit in the next section. The role played by the dummy node will also become clearer from the formulation: it prevents solutions consisting of a Hamiltonian circuit passing through the customer nodes only; such a situation cannot occur since the dummy node is a specified node connected to potential depots only.

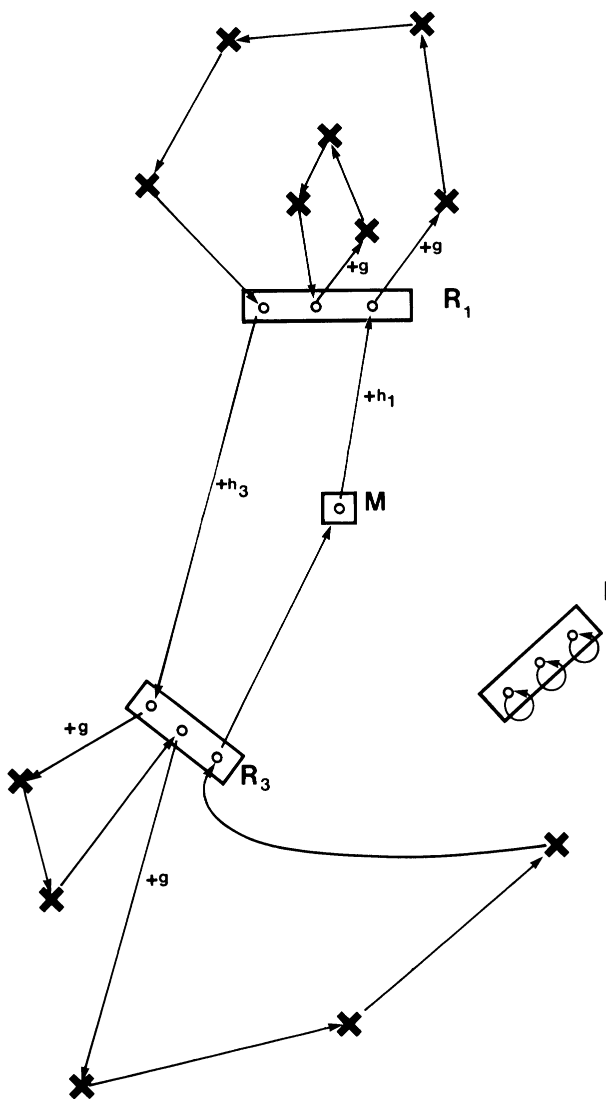

Fig. 2. LRP solution on the transformed graph.

Formally, any feasible solution on G' can easily be transformed back into a solution on G: it suffices (i) to remove the dummy node, all unused potential depots, as well as arcs belonging to $A_1 \cup A_2 \cup A_6$; (ii) to coalesce every R_k into a single depot node k.

2. FORMULATION

THE PROBLEM can now be formulated as a TSP variant in which

(i) It is required to visit all specified nodes exactly once and all nonspecified nodes at most once (such a problem was studied by LAPORTE et al.[9]);
(ii) There exist capacity and maximum cost constraints on the vehicle routes;
(iii) All vehicles start and end their journey at a depot, visit a number of customers and return to the same depot.

Now, suppose that conditions (ii) and (iii) which define legal vehicle routes are temporarily relaxed. A solution satisfying condition (i) consists of a number of Hamiltonian circuits over all nodes of N_1' and possibly some nodes of N_2'. If we now remove the dummy node and its two incident arcs from the solution, we obtain

(i) A number of Hamiltonian circuits including one depot node (element of R') and some customer nodes (elements of $N - R$); every such circuit corresponds to a vehicle route P_t and can be relabeled $(u_{\alpha(t)}, u_{\alpha(t)+1}, \ldots, u_{\beta(t)})$ where $u_{\alpha(t)} = u_{\beta(t)} \in R'$ and $u_{\alpha(t)+1}, \ldots, u_{\beta(t)-1} \in N - R$.
(ii) A sequence of nodes which can be broken into a number of subsequences $P_t = (u_{\alpha(t)}, u_{\alpha(t)+1}, \ldots, u_{\beta(t)})$, each corresponding to a vehicle route. Here $u_{\alpha(t)}, u_{\beta(t)} \in R'$, $u_{\alpha(t)} \neq u_{\beta(t)}$ and $u_{\alpha(t)+1}, \ldots, u_{\beta(t)-1} \in N - R$.

Route P_t is illegal if

(i) $u_{\alpha(t)} \in R_{k_1}$, $u_{\beta(t)} \in R_{k_2}$ and $k_1 \neq k_2$, or
(ii) $\sum_{l=\alpha(t)+1}^{\beta(t)-1} d_{u_l} > D$, or
(iii) $\sum_{l=\alpha(t)}^{\beta(t)-1} c'_{l,l+1} > L$.

Any illegal vehicle route can be eliminated by removing at least one of its arcs, i.e., by requiring that the number of its arcs does not exceed $\beta(t) - \alpha(t) - 1$ if $u_{\alpha(t)} = u_{\beta(t)}$ and $\beta(t) - \alpha(t) - 2$ if $u_{\alpha(t)} \neq u_{\beta(t)}$.

In order to formulate the problem, first define binary variables x_{uv} as follows:

(i) If $u \neq v$, $x_{uv} = 1$ if arc $(u, v) \in A'$ belongs to the solution and 0 otherwise;

(ii) If $u = v$, $x_{uv} = x_{uu} = 0$ if node $u \in N_2'$ belongs to the solution and 1 otherwise; if $u \in N_1'$, x_{uu} automatically takes the value 0; the x_{uu} variables correspond to the loops of A_7.

The problem formulation is then:

$$\text{(P)} \quad \text{minimize} \quad \sum_{u,v \in N'} c'_{uv}\, x_{uv} \tag{2}$$

subject to

$$\sum_{v \in N'} x_{uv} = 1 \quad (u \in N') \tag{3}$$

$$\sum_{u \in N'} x_{uv} = 1 \quad (v \in N') \tag{4}$$

$$\sum_{u,v \in S} x_{uv} \leq |S| - 1 \quad (S \subset N';\ S \neq N_1';\ S \cap N_1' \neq \phi;\ |S \cap R'| \neq 1) \tag{5}$$

$$\sum_{l=\alpha(t)}^{\beta(t)-1} x_{u_l, u_{l+1}} \leq \beta(t) - \alpha(t) - 1 \quad \text{(if } (u_{\alpha(t)}, \ldots, u_{\beta(t)}) \text{ is an illegal route and } u_{\alpha(t)} = u_{\beta(t)}) \tag{6}$$

$$\sum_{l=\alpha(t)}^{\beta(t)-1} x_{u_l, u_{l+1}} \leq \beta(t) - \alpha(t) - 2 \quad \text{(if } (u_{\alpha(t)}, \ldots, u_{\beta(t)}) \text{ is an illegal route and } u_{\alpha(t)} \neq u_{\beta(t)}) \tag{7}$$

$$x_{uv} = 0 \text{ or } 1. \quad (u, v \in N') \tag{8}$$

Constraints (3), (4) and (8) of (P) are assignment constraints: they ensure that every specified node will be entered and left exactly once. Now consider a nonspecified node u: if $x_{uu} = 1$, the sum of incoming and outgoing variables will then be forced to zero and the node will be left unvisited in the optimal solution; on the other hand, if $x_{uu} = 0$, this node will be forced into the solution. Constraints (5) are subtour elimination constraints. They are imposed for all proper subsets N' except (i) subsets of N_2', since it is never economical to create a subtour made up exclusively of nonspecified nodes; (ii) subsets including one node in R' and all their other nodes in $N - R$ since these correspond to vehicle routes; if such routes are illegal, they are eliminated through constraints (6). Constraints (6) and (7) forbid the formation of illegal routes. Their derivation has already been explained.

3. ALGORITHM

AS MENTIONED in the introduction, we use for the solution of (P) a modified version of the Carpaneto and Toth[2] branch and bound algorithm for the TSP. The algorithm first solves the AP relaxation of (P) defined by (2), (3), (4) and (8). Checks for violated constraints (5), (6) and (7) are then made and any infeasibility is dealt with by creating subproblems in which some of the variables are fixed at 0 or 1. Each of these subproblems is a constrained AP. The procedure just described is applied to all nodes of the search tree, according to the usual branch and bound rules. Further, in order to take advantage of the problem at hand, various tests are executed at every subproblem to permanently fix at 0 some variables in that subproblem and in all its descendants. We now introduce some notation and proceed to the step by step description of the algorithm.

z^*: the cost of the best solution to (P) so far identified. Initially, z^* can be set equal to a known upper bound to the problem or to infinity. At the end of the algorithm, z^* is the value of the optimal solution.
z_p: the value of the objective function of the subproblem at node p of the search tree
$\underline{z}_p$: a lower bound on z_p
I_p: the set of included arcs in subproblem p
E_p: the set of excluded arcs in subproblem p.

Step 1 (node 1 of the search tree). Set $I_1 = E_1 = \phi$ and solve the AP defined by (2), (3), (4) and (8). If $z_1 = z^*$, terminate.

Step 2 (feasibility check). Check whether the solution contains any illegal subtour or any infeasible route. If the solution is feasible, the optimum has been reached; terminate. If not, insert node 1 of the search tree in a queue.

Step 3 (node selection). If the queue is empty, terminate. Otherwise, select the next node (node p) from which to branch: branching is always done on the pending node having the smallest $\underline{z}_p$.

Step 4 (branching). The solution found at node p is illegal and must be eliminated. Consider the illegal subtour or route having the least number of arcs not already included in I_p. Let these arcs be $(u_1, v_1), \ldots, (u_a, v_a)$, listed in the order in which they appear on the subtour or route. Then define

$$I_q = \begin{cases} I_p & (q = 1) \\ I_p \cup \{(u_t, v_t): t = 1, \ldots, q-1\} & (q = 2, \ldots, a) \end{cases}$$

$$E_q^0 = E_p \cup \{(u_q, v_q)\} \quad (q = 1, \ldots, a)$$

Step 5 (excluding additional arcs). In subproblem q, $(N' - M)$ can be partitioned into $B(N' - M) = \{B_1, \ldots, B_b\}$ where $B_1, \ldots, B_b$ are singletons or subsets of nodes linked by the arcs of I_q. These subsets of nodes correspond to paths of arcs belonging to I_q. Now consider all paths having at least two nodes in R'. As in Section 3, these are fractioned into smaller paths having either two end nodes in

R' and possibly some middle nodes in $N - R$, or having one end node in R' and the remaining nodes in $N - R$. Now define $B' = B^0 \cup B^* \cup (\bigcup_k B^k)$ where B^0 is the set of nodes of $N - R$ not belonging to any path of included arcs; B^* is a set of paths of included arcs having all their nodes in $N - R$; B^k is the set of paths of included arcs which, after fractioning, have exactly one end node in R_k and all their remaining nodes in $N - R$.

Arcs excluded in order to prevent the occurrence of subtours disconnected from R'

Let $P_t = (u_{\alpha(t)}, \ldots, u_{\beta(t)}) \in B^*$. Then arc $(u_{\beta(t)}, u_{\alpha(t)})$ can be excluded.

Arcs excluded in order to prevent the occurrence of routes joining two different depots

a) First consider a path $P_t = (u_{\alpha(t)}, \ldots, u_{\beta(t)}) \in B^k$ for some $k > 0$ where $u_{\beta(t)} \in N - R$. Then $(u_{\beta(t)}, r_{k's})$ can be excluded for all $k' \neq k$ and $s \geqslant 2$. Similarly, $(r_{k's}, u_{\alpha(t)})$ can be excluded if $u_{\alpha(t)} \in N - R$.

b) Now consider two paths: $P_t = (u_{\alpha(t)}, \ldots, u_{b(t)}) \in B^k$ and $P_{t'} = (u_{\alpha(t')}, \ldots, u_{\beta(t')}) \in B^{k'}$ where $k, k' > 0$, $k \neq k'$ and $\beta(t)$, $\alpha(t') \in N - R$. Then arc $(u_{\beta(t)}, u_{\alpha(t')})$ can be excluded.

Arcs excluded in order to prevent the occurence of routes having a total weight exceeding D

Let P_t and $P_{t'}$ be two paths or singletons belonging to B'. If P_t is a path $(u_{\alpha(t)}, \ldots, u_{\beta(t)})$, its weight is equal to $w(\mathrm{P}_t) = \sum_{l=\alpha(t)}^{\beta(t)} d_{u_l}$ where $d_{u_l} = 0$ if $u_l \in R'$. A singleton may be considered as a degenerate path P_t with $u_{\alpha(t)} = u_{\beta(t)}$. Such a singleton has a weight $w(P_t) = d_{u_{\alpha(t)}}$. Then arc $(u_{\beta(t)}, u_{\alpha(t')})$ can be excluded if $w(P_t) + w(P_{t'}) > D$.

Arcs excluded in order to prevent the occurrence of routes having a cost exceeding L

Let u and v be two nodes of N' and define $\lambda(u, v)$, the value of a least cost path from u to v.

a) Consider a path $P_t = (u_{\alpha(t)}, \ldots, u_{\beta(t)}) \in B^*$ having a cost $\gamma(P_t) = \sum_{l=\alpha(t)}^{\beta(t)-1} c'_{u_l u_{l+1}}$. Then problem q has no feasible descendant if $\lambda(r_{k,1}, u_{\alpha(t)}) + \gamma(P_t) + \lambda(u_{\beta(t)}, r_{k,2}) > L$ for all $k \in R$. In such a case, subproblem q can be eliminated from consideration in the remainder of the algorithm.

b) Consider a path $P_t = (u_{\alpha(t)}, \ldots, u_{\beta(t)}) \in B^k$ for some k. Suppose $u_{\alpha(t)} = r_{ks}$ for some s and $u_{\beta(t)} \in N - R$. Then $(u_{\beta(t)}, r_{ks})$ can be excluded for $s = 1, \ldots, \bar{m}_k + 1$ if $\gamma(P_t) + \lambda(u_{\beta(t)}, r_{k,2}) > L$. Similarly, suppose $u_{\beta(t)} = r_{ks}$ for some k and $u_{\alpha(t)} \in N - R$. Then $(r_{ks}, u_{\alpha(t)})$ can be excluded for $s = 1, \ldots, \bar{m}_k + 1$ if $\gamma(P_t) + \lambda(r_{k,1}, u_{\alpha(t)}) > L$.

c) Now consider two paths $P_t = (u_{\alpha(t)}, \ldots, u_{\beta(t)}) \in B^*$ and $P_{t'} = (u_{\alpha(t')}, \ldots, u_{\beta(t')}) \in B^*$. Arc $(u_{\beta(t)}, u_{\alpha(t')})$ can be excluded if $\gamma(P_t) + c'_{u_{\beta(t)}, u_{\alpha(t')}} + \gamma(P_{t'}) > L$ or if $\lambda(r_{k,1}, u_{\alpha(t)}) + \gamma(P_t) + c'_{u_{\beta(t)}, u_{\alpha(t')}} + \gamma(P_{t'}) + \lambda(u_{\beta(t')}, r_{k,2}) > L$ for all $k \in R$.

d) Consider two paths $P_t = (u_{\alpha(t)}, \ldots, u_{\beta(t)}) \in B^*$ and $P_{t'} = (u_{\alpha(t')}, \ldots, u_{\beta(t')}) \in B^k$ for some k. Let $u_{\alpha(t')} \in R'$; then arc $(u_{\beta(t')}, u_{\alpha(t)})$ can be excluded if $\gamma(P_{t'}) + c'_{u_{\beta(t')}, u_{\alpha(t)}} + \gamma(P_t) + \lambda(u_{\beta(t)}, r_{k,2}) > L$. Similarly, let $u_{\beta(t')} \in R'$; then arc $(u_{\beta(t)}, u_{\alpha(t')})$, can be excluded if $\lambda(r_{k,1}, u_{\alpha(t)}) + \gamma(P_t) + c'_{u_{\beta(t)}, u_{\alpha(t')}} + \gamma(P_{t'}) > L$.

e) Finally, let $P_t = (u_{\alpha(t)}, \ldots, u_{\beta(t)}) \in B^k$ and $P_{t'} = (u_{\alpha(t')}, \ldots, u_{\beta(t')}) \in B^k$ for some k. Let $u_{\alpha(t)} = r_{ks}$ for some s and $u_{\beta(t')} = r_{ks'}$, for some $s' \neq s$. Then arc $(u_{\beta(t)}, u_{\alpha(t')})$ can be excluded if $\gamma(P_t) + c'_{u_{\beta(t)} u_{\alpha(t')}} + \gamma(P_{t'}) > L$.

At the end of Step 5, E_q is defined by including in E_q^0 all arcs excluded by one of the above criteria.

For each remaining subproblem q, execute Steps 6 to 8. Then go to Step 3

Step 6 (AP bound). Compute a lower bound $\underline{z}_q$ on z_q according to Step 3 of [1]. If $\underline{z}_q \geqslant z^*$, consider the next q and repeat Step 6.

Step 7 (AP solution). Solve the AP constrained by E_q and I_q associated with node q. If $z_q \geqslant z^*$, consider the next q and proceed to Step 6.

Step 8 (feasibility check). Check whether the current solution contains any illegal subtours or routes. If not, set $z^* = z_q$ and store the solution; if $z^* = z_p$, proceed to Step 3; if $z^* \neq z_p$, consider the next q and proceed to Step 6.

4. COMPUTATIONAL RESULTS

THE ALGORITHM described in Section 3 was tested over a number of randomly generated problems. The three cases under study were tested separately: (i) capacity constrained VRPs, (ii) cost constrained VRPs, and (iii) cost constrained LRPs.

In each case, problems were generated according to two kinds of cost matrices (C): problems in which C satisfied the triangle inequality (type Δ) and problems in which C did not satisfy the triangle inequality (type $\bar{\Delta}$). In the first case, C was obtained as follows: points were first generated in $[0,100]^2$ according to a uniform distribution and a cost matrix $C_1 = (c_{ij}^1)$ was obtained by computing the Euclidean distances between these points; a second set of points was then generated in a similar fashion, with the corresponding Euclidean distance matrix $C_2 = (c_{ij}^2)$. Matrix $C = (c_{ij})$ was obtained by first defining

$$c_{ij} = \begin{cases} 0 & \text{if } i = j \\ [c_{ij}^1] & \text{if } i < j \\ [c_{ij}^2] & \text{if } i > j \end{cases} \tag{9}$$

where $[c]$ denotes the integer nearest to c. Then in order to obtain a type Δ matrix, all c_{ij}'s for

TABLE I

Computational results for capacity constrained VRPs

TYPE	|N|	|R|	PROBLEM	θ MAX	LOAD	NODES	QUEUE	DESC	ELIM	TIME
$\bar{\Delta}$	20	1	1	1	.90	75	15	11	109	0.7
			2	3	.97	172	139	22	444	3.5
			3	2	.88	38	23	5	65	0.7
		2	1	3	.91	6149	2672	1725	26094	260.9
			2	2	.76	106	60	27	423	2.5
			3	3	.80	1838	808	473	7036	44.4
		3	1	2	.84	1310	625	445	8644	48.9
			2	3	.82	5813	2620	1968	40131	317.9
			3	3	.71	551	422	194	3312	21.8
	40	1	1	2	.61	78	41	11	66	2.7
			2	0	.50	20	5	3	16	0.5
			3	1	.80	673	87	62	1736	15.2
		2	1	2	.84	833	281	143	3174	29.1
			2	1	.62	172	35	33	502	4.4
			3	2	.73	426	74	48	2077	11.0
		3	1	1	.64	3141	1696	719	17473	164.8
			2	2	.73	4444	1500	965	26649	227.2
			3	3	.64	1391	691	341	6891	63.2
	60	1	1	3	1.00	10288	1537	649	24519	454.8
			2	2	.99	380	251	20	751	27.5
			3	3	.90	291	71	24	590	13.8
		2	1	2	.92	10405	2168	1314	48578	542.1
			2	2	.71	2487	597	250	10352	107.3
			3	3	.83	460	305	54	2035	34.7
		3	1	1	.60	323	145	75	1412	15.5
			2	2	.71	2290	701	379	13146	100.1
			3	2	.62	397	52	50	2446	16.6
	80	1	1	0	.50	5	4	2	2	0.5
			2	1	.96	519	93	26	1387	24.4
			3	1	.99	3197	215	173	6063	122.7
		2	1	1	.88	482	312	48	1490	41.6
			2	3	.61	528	180	33	1883	33.5
			3	2	.74	1737	600	175	6239	97.5
		3	1	3	.70	3587	1430	548	20236	229.6
			2	2	.63	1568	1490	209	8815	193.9
			3	3	.62	1004	497	178	4512	54.7
Δ	20	1	1	3	.91	203	53	24	563	2.7
			2	1	.87	14	4	2	10	0.2
			3	2	.90	42	13	5	50	0.4
		2	1	1	.79	62	47	20	232	1.7
			2	2	.80	962	376	259	3859	19.4
			3	0	.25	1185	523	472	2240	22.9
		3	1	3	.67	1758	1049	539	10234	48.8
			2	2	.64	1339	473	394	8617	32.1
			3	3	.79	4247	2686	1715	26845	277.5
	40	1	1	3	.87	1324	290	117	2897	26.0
			2	1	.57	5323	624	369	4978	80.8
			3	2	.81	48	9	6	49	0.9
		2	1	3	.90	1545	497	288	5092	51.4
			2	0	.25	260	134	87	366	10.6
			3	3	.71	6363	1511	1149	30109	192.4
		3	1	2	.61	2395	1627	758	11438	142.2
			2	2	.53	67	42	13	418	3.8
			3	1	.50	3112	925	776	16206	125.2
	60	1	1	0	.50	48	6	3	44	2.6
			2	3	.94	1180	208	98	3263	37.6
			3	1	.74	112	65	32	77	7.1
		2	1	2	.79	2744	1021	291	14502	163.9
			2	2	.66	1224	1051	138	5114	117.2
			3	2	.78	2048	607	297	10122	94.4
		3	1	3	.41	1377	399	169	6732	49.1
			2	3	.62	589	561	83	4526	70.5
			3	2	.61	985	382	158	7347	64.1
	80	1	1	3	.97	31	30	3	. 5	4.4
			2	2	.85	4476	758	206	5343	214.6
			3	3	.92	6358	1091	288	13781	361.7
		2	1	0	.25	95	81	34	94	9.0
			2	3	.64	138	133	21	679	18.7
			3	1	.79	583	272	63	2716	51.0
		3	1	2	.59	4526	752	546	33750	249.2
			2	2	.62	3732	1373	583	19321	288.4
			3	2	.32	4493	730	364	16576	195.1

which $c_{ij} > c_{ih} + c_{hj}$ for some h were replaced by $\min_h\{c_{ih} + c_{hj}\}$. This process was repeated until C satisfied the triangle inequality. Type $\bar{\Delta}$ problems were obtained by generating the c_{ij}'s according to a discrete uniform distribution on [1,100].

For each type of problem, three different sets of data were produced and the problem was solved for a number of parameters. In all cases, the initial number of points (before transforming G into G') was taken as $|N| = 20, 40, 60, 80$. There were between 1 and 3 depots in the two types of VRPs and either 2 or 3 potential depots in the LRPs. Moreover:

(i) In the capacity constrainted VRPs, the demands d_j were generated according to a discrete uniform distribution on [1,100]. The number of vehicles was fixed at 2 for each depot, thus the number of nodes per vehicle is deterministic. There were no vehicle and no depot fixed costs. In order to determine the vehicle capacity, we used successive levels of tightness $\theta(\theta = 0, 1, 2, 3)$. At level $\theta = 0$, D was set equal to $-1 + \sum_{j\in N-R} d_j$. At level $\theta > 1$, we first considered the optimal solution obtained at level $\theta - 1$ and computed $K_{\theta-1}$, the weight carried by the most heavily loaded vehicle. We then set $D = 0.9 \times K_{\theta-1}$.

(ii) In cost constrained VRPs, the number of vehicles was also set at 2 for each depot and there were no vehicle and no depot fixed costs. We used successive values of L, with four increasing degrees of tightness. At level $\theta = 0$, L was equal to infinity. Letting $H_{\theta-1}$ denote the cost of the most expensive route at level $\theta - 1$, we then set $L = 0.9 \times H_{\theta-1}$ for $\theta > 1$.

(iii) In cost constrained LRPs, we allowed a maximum of two vehicles per potential depot. The fixed cost of a vehicle was $g = 10$ and that of depot k was $h_k = 20$. The successive values of L were determined as for the VRP.

All problems were solved on the University of Montreal CYBER 835 computer, but all reported CPU times are CYBER 173 equivalent. We used a $86{,}000_{10}$ word memory and allowed a maximum of 600 CPU seconds per problem. However, this time limit never constituted the cause for problem failure, but computer memory limits prevented us from solving problems larger than those reported below: the size of the search tree became too large.

The main computational results are reported in Tables I–III. The meanings of the various column headings are as follows.

TYPE: cost matrix C satisfying the triangle inequality (Δ) or not ($\bar{\Delta}$)

|N|: total number of customer and depot nodes in G

|R|: number of depots (in VRPs) or of potential depots (in LRPs) in G

PROBLEM: identification number for the three problems attempted in each case

θMAX: tightest level θ for which the problem was solved

LOAD: total demand divided by total vehicle capacity

TIGHTNESS: average route cost divided by maximum allowed cost (L)

DEPOTS: in LRPs, number of depots used in optimal solution

VEHICLES: in LRPs, total number of vehicles used in optimal solution

NODES: number of nodes p in the search tree for which a lower bound z_p was computed in Step 6 of the algorithm

TABLE II

Computational results for cost constrained VRPs

TYPE	|N|	|R|	PROBLEM	θ MAX	TIGHTNESS	NODES	QUEUE	DESC	ELIM	TIME
$\bar{\Delta}$	20	1	1	1	.98	510	136	92	4365	10.4
			2	2	.93	181	34	32	2165	3.2
			3	2	.77	43	8	5	382	0.7
		2	1	2	.80	216	75	50	2091	6.2
			2	3	.69	163	86	38	1633	4.9
			3	3	.94	226	133	73	2285	8.0
		3	1	1	.64	242	104	87	2503	9.6
			2	1	.44	45	29	15	419	1.5
			3	1	.74	80	49	33	810	3.2
	40	1	1	2	.92	122	90	19	2884	7.5
			2	1	.61	61	26	4	1296	2.5
			3	3	.97	210	78	20	5297	11.1
		2	1	0	0	11	6	2	8	0.4
			2	3	.82	1100	479	163	31340	72.2
			3	2	.85	444	110	87	10565	25.3

TABLE II—(*Continued*)

TYPE	|N|	|R|	PROBLEM	θ MAX	TIGHTNESS	NODES	QUEUE	DESC	ELIM	TIME
		3	1	1	.54	348	131	94	8443	21.5
			2	0	0	92	71	23	415	5.6
			3	3	.71	1184	541	298	27476	92.1
	60	1	1	2	.90	219	24	15	7653	16.0
			2	2	.96	270	148	18	9652	20.6
			3	3	.99	1475	268	94	61294	110.6
		2	1	2	.61	75	25	9	2426	7.1
			2	1	.60	517	72	64	16444	33.0
			3	1	.77	321	60	41	13945	21.3
		3	1	0	0	98	65	24	398	7.0
			2	2	.62	2049	510	455	70077	171.8
			3	1	.61	280	108	52	10435	22.5
	80	1	1	3	.95	2156	350	149	116310	219.5
			2	1	.93	344	84	17	18827	42.5
			3	0	0	24	9	4	19	1.4
		2	1	2	.72	3022	508	241	193331	397.6
			2	3	.73	1479	395	135	82487	174.7
			3	1	.69	142	134	15	8041	18.1
		3	1	1	.57	1608	285	246	86464	142.3
			2	3	.62	1718	310	252	91573	149.0
			3	1	.53	2847	727	453	139319	307.1
Δ	20	1	1	1	.65	29	7	2	122	0.3
			2	2	.89	4602	1725	894	26813	69.4
			3	0	0	3	1	1	1	0.1
		2	1	3	.79	160	64	41	1123	3.5
			2	3	.88	1627	608	428	14311	32.8
			3	3	.81	2692	670	670	18781	48.1
		3	1	1	.85	25	14	9	255	0.9
			2	1	.60	424	127	113	3941	9.8
			3	1	.40	478	221	214	1907	12.9
	40	1	1	3	.96	1853	271	185	11504	46.4
			2	2	.91	156	32	12	629	3.6
			3	3	.89	3194	614	312	22319	94.1
		2	1	2	.76	2552	731	380	31691	94.3
			2	3	.72	433	116	60	6443	18.0
			3	2	.62	155	43	26	881	5.4
		3	1	0	0	269	80	76	1174	9.8
			2	1	.65	766	518	185	8385	50.3
			3	3	.59	1066	590	287	7687	61.8
	60	1	1	0	0	10	1	1	8	0.5
			2	1	.90	450	50	47	2752	11.6
			3	2	.85	926	161	45	6775	45.6
		2	1	0	0	106	34	23	256	5.2
			2	0	0	30	6	5	66	1.3
			3	2	.83	497	269	70	4693	40.2
		3	1	0	0	367	176	96	1654	24.7
			2	2	.56	972	483	191	6551	66.3
			3	1	.54	1524	595	353	9513	99.1
	80	1	1	2	.86	2099	241	128	7850	89.1
			2	3	.94	484	103	28	2040	22.4
			3	2	.91	14029	988	834	89852	514.5
		2	1	3	.61	1383	750	141	8169	164.0
			2	1	.83	418	274	43	2353	52.6
			3	2	.78	75	35	11	559	6.7
		3	1	1	.54	1453	989	238	10862	202.3
			2	0	0	418	282	95	1871	55.1
			3	1	.66	166	139	26	2872	25.9

TABLE III

Computational results for cost constrained LRPs

TYPE	|N|	|R|	PROBLEM	θ MAX	DEPOTS	VEHICLES	TIGHTNESS	NODES	QUEUE	DESC	ELIM	TIME
$\bar{\Delta}$	20	2	1	3	1	2	.92	346	72	63	3652	8.7
			2	3	1	2	.97	3828	1247	768	45205	117.8
			3	2	1	2	.78	5004	1008	946	56679	130.9
		3	1	3	2	3	.76	1943	841	455	22295	66.3
			2	2	1	2	.84	695	168	144	6572	19.1
			3	3	2	2	.94	355	89	81	3652	10.7
	40	2	1	0	1	1	0	113	15	8	177	3.0
			2	0	1	1	0	338	41	28	937	10.1
			3	0	1	1	0	12	2	1	10	0.4
		3	1	0	1	1	0	19	5	4	57	0.6
			2	1	2	2	.55	2109	550	189	34665	117.4
			3	0	1	1	0	22	16	3	18	0.8
	60	2	1	0	1	1	0	76	27	10	65	3.2
			2	0	1	1	0	34	25	9	58	2.0
			3	0	1	1	0	49	17	4	44	1.8
		3	1	0	1	1	0	81	14	11	69	2.0
			2	0	1	1	0	150	57	13	417	9.6
			3	0	1	1	0	4	3	1	2	0.4
	80	2	1	0	1	1	0	516	100	29	858	27.4
			2	0	1	1	0	1444	103	97	1806	56.7
			3	0	1	1	0	210	115	67	433	21.5
		3	1	0	1	1	0	239	126	13	641	21.7
			2	0	1	1	0	180	30	5	1042	9.2
			3	0	1	1	0	23	16	4	18	1.5
Δ	20	2	1	2	2	2	.57	1103	366	246	8931	25.5
			2	1	2	2	.55	357	63	50	1512	5.3
			3	1	2	2	.56	3997	680	630	21608	57.4
		3	1	0	1	1	0	90	38	17	286	1.9
			2	1	2	2	.57	1106	266	212	6675	19.5
			3	2	2	2	.83	4423	1341	932	32861	108.3
	40	2	1	0	1	1	0	1019	112	67	951	18.4
			2	0	1	1	0	7	3	1	5	0.5
			3	0	1	1	0	19	6	3	15	0.7
		3	1	0	1	1	0	376	163	25	1640	17.9
			2	1	2	2	.56	12190	1144	985	80321	354.0
			3	0	1	1	0	1609	474	164	7468	57.6
	60	2	1	0	1	1	0	46	12	7	38	1.6
			2	0	1	1	0	29	1	1	27	0.7
			3	0	1	1	0	320	57	9	984	16.0
		3	1	0	1	1	0	168	20	13	154	4.4
			2	0	1	1	0	82	65	28	53	8.3
			3	0	1	1	0	17	12	4	12	1.7
	80	2	1	0	1	1	0	76	75	9	157	9.8
			2	0	1	1	0	186	28	22	430	9.4
			3	0	1	1	0	108	27	9	115	4.8
		3	1	0	1	1	0	417	400	13	2571	84.2
			2	0	1	1	0	42	8	2	39	3.3
			3	0	1	1	0	86	11	6	79	3.9

QUEUE: number of these nodes with a value z_p less than z^* at the time of their generation and which were later inserted in the queue

DESC: number of nodes in the queue which were later examined

ELIM total number of arcs which were excluded in Step 5 of the algorithm, in the whole of the search tree (the same arc may have been eliminated a number of times)

TIME: number of CPU seconds.

The results shown in Tables I to III indicate that the algorithm was quite successful. Problems involving up to 80 nodes were solved to optimality within relatively short computing times. To the authors' knowledge, this is the first time such large multi-depot problems have been solved by an exact algorithm. The main features of the results are:

(i) Difficulty increases with problem size and with the number of depots. The latter observation is easily explained by the fact that in problems involving more depots, a larger number of illegal vehicle routes joining two different depots will be generated and will require elimination and more branching on the algorithm.

(ii) Problems satisfying the triangle inequality were also generally more difficult to solve than problems not possessing this property. This can be seen by the generally lower values of θMAX in problems of type Δ. We can offer the following explanation for this phenomenon (known to occur in a variety of TSP extensions): in problems satisfying the triangle inequality, the lengths of the various feasible solutions have a smaller variance than in purely random problems. This translates into less dominance in the search tree and into more branching.

(iii) The success of the various arc exclusion criteria is quite striking, as indicated by the large entries in the ELIM column.

(iv) In many of the cost constrainted VRPs, the value of θMAX may appear quite low. The reason for this is that for the next value (θMAX + 1), the problem was simply infeasible, but this does not appear directly in Table II.

(v) The method allowed the solution of very large LRPs, but most of these were solved for the unconstrained case only, i.e., for θMAX = 0.

(vi) The columns LOAD and TIGHTNESS measure average vehicle use and the extent to which the route maximum cost constraints were binding. In tables 2 and 3, a zero value in the TIGHTNESS column means that this constraint could not be applied (i.e., the problem was solved at level θMAX = 0). The values reported in these columns are relatively large and often indicate that going to the next level of tightness would have yielded an infeasible solution. In some instances where the LOAD or TIGHTNESS value was low, the problem was solved for θMAX = 3. Since $\theta = 3$ was the largest value for which problems were attempted, there is a possibility that a more constrained problem may have been solved. Finally, there remain some cases where both LOAD or TIGHTNESS and θMAX are low. It is the distance matrix configuration of these problems that made them difficult to solve.

As was the case for the TSP,[16] asymmetrical problems such as those treated in this paper appear to be much easier to solve than symmetrical problems. Prior to this study, the largest single depot symmetrical capacitated or cost constrained VRPs that could be solved to optimality by using a constraint relaxation approach contained 60 nodes.[15] The largest multi-depot symmetrical capacitated VRPs involved 25 nodes.[13] In this paper, sizes of up to 80 can be reached without difficulty.

ACKNOWLEDGMENTS

THE AUTHORS are grateful to the Canadian Natural Sciences and Engineering Research Council (grants A4747 and A5486) for their financial support. Thanks are also due to Giorgio Carpaneto and Paolo Toth who made their TSP code available to the authors.

REFERENCES

1. L. D. BODIN, B. L. GOLDEN, A. A. ASSAD AND M. O. BALL, "Routing and Scheduling of Vehicles and Crews. The State of the Art," *Comput. Opns. Res.* **10,** 62–212 (1983).
2. G. CARPANETO, P. TOTH, "Some New Branching and Bounding Criteria for the Asymmetric Travelling Salesman Problem," *Mgmt. Sci.* **26,** 736–743 (1980).
3. N. CHRISTOFIDES, A. MINGOZZI AND P. TOTH, "State Space Relaxation Procedures for the Computation of Bounds to Routing Problems," *Networks* **11,** 145–164 (1981).
4. N. CHRISTOFIDES, "Vehicle Scheduling and Routing," presented at the 12th International Symposium on Mathematical Programming, Cambridge, Mass., 1985.
5. G. CLARKE AND J. WRIGHT, "Scheduling of Vehicles from a Central Depot to a Number of Delivery Points," *Opns. Res.* **12,** 568–581 (1964).
6. B. L. GOLDEN AND A. A. ASSAD, "Perspectives on Vehicle Routing: exciting new Developments," *Opns. Res.* **34,** 803–810 (1986).
7. B. L. GOLDEN, T. L. MAGNANTI AND H. Q. NGUYEN, "Implementing Vehicle Routing Algorithms," *Networks* **7,** 113–148 (1977).
8. G. LAPORTE, "Location-Routing Problems," in *Vehicle Routing: Methods and Studies*, pp. 163–198, B. L. Golden and A. A. Assad (eds.), North-Holland, Amsterdam, 1988.
9. G. LAPORTE, H. MERCURE AND Y. NOBERT, "Optimal Tour Planning with Specified Nodes," *R.A.I.R.O. (Rech. Opnl.)* **18,** 203–210 (1984).
10. G. LAPORTE, H. MERCURE AND Y. NOBERT, "An Exact Algorithm for the Asymmetrical Capacitated Routing Problem," *Networks* **16,** 33–46 (1986).
11. G. LAPORTE, H. MERCURE AND Y. NOBERT, "An Opti-

mal Algorithm for a Class of Asymmetrical Vehicle Routing Problems," Cahier du GERAD G-86-14, Ecole des Hautes Etudes Commerciales de Montréal, 1986.
12. G. LAPORTE AND Y. NOBERT, "Exact Algorithms for the Vehicle Routing Problem," in *Surveys in Combinatorial Optimization*, pp. 147–184, S. Martello et al. (eds.), North-Holland, Amsterdam, 1987.
13. G. LAPORTE, Y. NOBERT AND D. ARPIN, "Optimal Solutions to Capacitated Multidepot Vehicle Routing Problems," *Congr. Num.* **44,** 283–292 (1984).
14. G. LAPORTE, Y. NOBERT AND D. ARPIN, "An Exact Algorithm for Solving a Capacitated Location-Routing Problem." *Ann. Opns. Res.* **6,** 239–310 (1986).
15. G. LAPORTE, Y. NOBERT AND M. DESROCHERS, "Optimal Routing Under Capacity and Distance Restrictions," *Opns. Res.* **33,** 1050–1073 (1985).
16. E. L. LAWLER, J. K. LENSTRA, A. H. G. RINNOOY KAN AND D. B. SCHMOYS, *The Traveling Salesman Problem. A Guided Tour of Combinatorial Optimization*, Wiley, Chichester, U.K., 1985.
17. J. K. LENSTRA AND A. H. G. RINNOOY KAN, "Some Simple Applications of the Travelling Salesman Problem," *Opnl. Res. Quart.* **26,** 717–734 (1975).
18. M. D. NELSON, K. E. NYGARD, J. H. GRIFFIN, AND W. E. SHREVE, "Implementation Techniques for the Vehicle Routing Problem," *Comput. Opns. Res.* **12,** 273–283 (1985).
19. Y. NOBERT, "Construction d'algorithmes optimaux pour des extensions au problème du voyageur de commerce," Thèse de doctorat, Département d'informatique et de recherche opérationnelle," Université de Montréal, 1982.
20. I. OR AND W. P. PIERSKALLA, "A Transportation Location-Allocation Model for the Original Blood Banking," *AIIE Trans.* **11,** 86–95 (1979).
21. J. PERL, AND M. S. DASKIN, "A Warehouse Location-Routing Problem," *Trans. Res.* **19B,** 381–396 (1985).
22. B. SRIKAR AND R. SRIVASTAVA, "Solution Methodology for the Location-Routing Problem," presented at the ORSA/TIMS Conference, Orlando, Fla., 1983.

(Received, March 1987; revision received November 1987; accepted December 1987)