Mathematical Programming 20 (1981) 255-282.
North-Holland Publishing Company

EXACT ALGORITHMS FOR THE VEHICLE ROUTING PROBLEM, BASED ON SPANNING TREE AND SHORTEST PATH RELAXATIONS

N. CHRISTOFIDES

Imperial College, London SW7 2BX, England

A. MINGOZZI

SOGESTA, Urbino, Italy

P. TOTH

University of Bologna, Bologna, Italy

Received 17 November 1978
Revised manuscript received 17March 1980

We consider the problem of routing vehicles stationed at a central facility (depot) to supply customers with known demands, in such a way as to minimize the total distance travelled. The problem is referred to as the vehicle routing problem (VRP) and is a generalization of the multiple travelling salesman problem that has many practical applications.

We present tree search algorithms for the exact solution of the VRP incorporating lower bounds computed from (i) shortest spanning k-degree centre tree (k-DCT), and (ii) q-routes. The final algorithms also include problem reduction and dominance tests.

Computational results are presented for a number of problems derived from the literature. The results show that the bounds derived from the q-routes are superior to those from k-DCT and that VRPs of up to about 25 customers can be solved exactly.

Key words: Vehicle Routing, Lagrangean Relaxation, Shortest Spanning Trees, Dynamic Programming Relaxation.

1. Introduction

We are considering a problem in which a set of geographically dispersed "customers" with known requirements must be served with a fleet of "vehicles" stationed at a central facility or depot in such a way as to minimize some distribution objective. It is assumed that all vehicle routes must start and finish at the depot.

The *vehicle routing problem* (VRP) is a generic name given to a whole class of problems involving the visiting of "customers" by "vehicles". The VRP (also known in the literature as the "vehicle scheduling" [8, 9, 10], "vehicle dispatching" [4, 11, 20] or simply as the "delivery" problem [1, 14, 21]) appears very frequently in practical situations not directly related to the physical delivery of goods. For example, the collection from mail-boxes, the pickup of children by school buses, house-call tours by a doctor, preventive maintenance inspection tours, the delivery of laundry, etc. are all VRPs in which the "delivery" operation may be a collection, collection and/or delivery, or neither, and which may not even be of a physical nature.

The basic VRP considered here is as follows.

A graph $G = (X, A)$ is defined by the set X of its vertices and the set A of its arcs.

Let $X' = \{x_i \mid i = 1, \dots, N\}$ be used for the set of N customers and let x_0 be the depot. $X = X' \cup \{x_0\}$.

A customer x_i has the following requirements:

(a) a quantity q_i of some product to be delivered by a vehicle,

(b) a "cost" u_i required by a vehicle to unload the quantity q_i, at x_i.

We assume that M identical vehicles each of capacity Q are stationed at the depot and that the total "cost" (e.g. "distance" or "time") of a vehicle route must be less than or equal to a given number T.

The number of vehicles is assumed to be large enough for a feasible solution to exist.

We further assume that the "cost" of the least cost path from every vertex x_i to every vertex x_j is given as c_{ij}. It is required that the total quantity on each vehicle route is less than or equal to Q, and that the total "cost" of each route (computed as the sum of the costs c_{ij} of the arcs (x_i, x_j) forming the route, plus the sum of the u_i for those customers x_i on the route) is less than or equal to T.

The objective in the VRP that is considered here is to design feasible routes—one for each vehicle—in order to supply all of the customers and minimize the total "cost" of all the routes. For the purpose of this paper, the "cost" c_{ij} mentioned above can be taken to be either travel distances or travel times between the customers.

The VRP defined above is a generalization of the travelling salesman problem (TSP). However, although for this latter problem exact methods of solution have been developed which can solve problems of one or two hundred customers [5], for the VRP no such algorithms exist. In fact the largest size of general VRPs reported solved in the literature involve problems with ten or twelve customers [9] although very special types of VRPs have been solved for larger sizes [4].

In this paper we develop a number of exact branch and bound algorithms for the general VRP defined above. These algorithms are based on bounds derived from: (i) the shortest spanning tree with a fixed degree at one specified vertex, and (ii) minimum q-routes, $q = 1, \dots, Q$. These are routes, on which the total load is exactly q, starting from the depot passing through a subset of the customers and returning back to the depot.
Langrangean penalty procedures are used to compute these bounds.

We also consider and computationally evaluate two different branching schemes for the tree search and introduce some dominance tests. Computational results on a number of problems are presented.

2. Problem formulation

We give below a formulation of the VRP as an integer program. This formulation is a simplification of the ones given in [6, 12].

Let $\xi_{ijk} = 1$ if vehicle k visits customer x_j immediately after visiting customer x_i, $\xi_{ijk} = 0$ otherwise.
The VRP is:

$$\text{minimize} \quad z = \sum_{i=0}^{N} \sum_{j=0}^{N} \left(c_{ij} \sum_{k=1}^{M} \xi_{ijk} \right), \tag{1}$$

$$\text{subject to} \quad \sum_{i=0}^{N} \sum_{k=1}^{M} \xi_{ijk} = 1, \quad j = 1, \dots, N, \tag{2}$$

$$\sum_{i=0}^{N} \xi_{ipk} - \sum_{j=0}^{N} \xi_{pjk} = 0, \quad k = 1, \dots, M, \quad p = 0, \dots, N, \tag{3}$$

$$\sum_{i=1}^{N} \left(q_i \sum_{j=0}^{N} \xi_{ijk} \right) \leq Q, \quad k = 1, \dots, M, \tag{4}$$

$$\sum_{i=0}^{N} \sum_{j=0}^{N} c_{ij} \xi_{ijk} + \sum_{i=1}^{N} \left(u_i \sum_{j=0}^{N} \xi_{ijk} \right) \leq T, \quad k = 1, \dots, M, \tag{5}$$

$$\sum_{j=1}^{N} \xi_{0jk} = 1, \quad k = 1, \dots, M, \tag{6}$$

$$y_i - y_j + N \sum_{k=1}^{M} \xi_{ijk} \leq N - 1, \quad i \neq j = 1, \dots N \tag{7}$$

$$\xi_{ijk} \in \{0, 1\} \quad \text{for all } i, j, k, \tag{8}$$

$$y_i \ \text{arbitrary}.$$

Expression (2) states that a customer must be visited exactly once. Expression (3) states that if a vehicle visits a customer, it must also depart from it. Expressions (4) and (5) are the capacity and "cost" limitations on each route. Expression (6) states that a vehicle must be used exactly once. Expression (7) is the subtour-elimination condition derived for the travelling salesman problem by Miller, Tucker and Zemlin [19], and which also forces each route to pass through the depot; (8) are the integrality conditions.

It is quite clear that the above formulation is too complex to be useful in solving VRPs of non-trivial size.

3. The computation of lower bounds

In any branch and bound procedure the calculation of bounds on the value of the solution to a remaining problem (at some node of the tree) is of the utmost importance to the efficiency of the algorithm. In this section we describe two different lower bounds for the VRP. These bounds are imbedded into tree search procedures and the resulting algorithms are described in the following section. In order to simplify the presentation we will henceforth assume symmetric $[c_{ij}]$ matrices, and use both c_{ij} and c_l to mean the cost of arc (i, j) or arc l respectively.

3.1. *Bound from the minimum k-degree centre tree* (k-DCT)

It is quite apparent that the value of the solution to the M-TSP is a lower bound to the value of the solution to the VRP using M vehicles, since the VRP is the M-TSP with additional constraints. We will, therefore, in this section derive a bound for the M-TSP and then specialise it for the VRP.

Consider the solution to an M-TSP shown in Fig. 1, where the depot is numbered x_0, and where $M = 4$ is the number of routes. The removal of arcs (x_0, A), (x_0, B), (x_0, C) and (D, E), one from each route—from this solution—produces a tree where the degree of x_0 is $k = 5$. We will call a tree where the degree of x_0 is k a *k-degree centre tree* (k-DCT). In general, if any set S_0 of $y \leq M$ arcs adjacent to x_0 and any set S_1 of $M - y$ arcs not adjacent to x_0 are removed—one arc from each route—the resulting graph is a k-DCT with $k = 2M - y$. The arcs forming the solution to the M-TSP are, therefore, distinguished into three sets: those arcs forming a k-DCT (with an associated variable $\xi_l = 1$ if arc l is in the k-DCT and $\xi_l = 0$ otherwise); those arcs forming a set S_0 (with $\xi_l^0 = 1$ if arc l is in set S_0 and $\xi_l^0 = 0$ otherwise); and those arcs forming a set S_1 (with $\xi_l^1 = 1$ if l is in set S_1 and $\xi_l^1 = 0$ otherwise).

The M-TSP can now be formulated as follows:

$$\text{minimize} \quad z = \sum_{l=1}^{m} c_l(\xi_l + \xi_l^0 + \xi_l^1), \tag{9}$$

$$\text{subject to} \quad \sum_{l \in (S_t, \bar{S}_t)} \xi_l \geq 1, \quad \forall S_t \subset X, \quad S_t \neq \emptyset, \tag{10}$$

$$\sum_{l \in A_0} \xi_l = 2M - y, \tag{11}$$

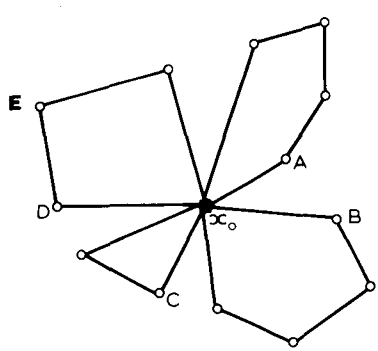

Fig. 1.

$$\sum_{l=1}^{m} \xi_l = N, \tag{12}$$

$$\sum_{l \in A_0} \xi_l^0 = y, \tag{13}$$

$$\sum_{l \in A - A_0} \xi_l^1 = M - y, \tag{14}$$

$$\sum_{l \in A_i} (\xi_l + \xi_l^0 + \xi_l^1) = 2, \quad i = 1, \dots, N, \tag{15}$$

$$\xi_l \in \{0, 1\}, \tag{16a}$$

$$\xi_l^0 \in \{0, 1\}, \tag{16b}$$

$$\xi_l^1 \in \{0, 1\}. \tag{16c}$$

y arbitrary.

where $m = |A|$ is the total number of arcs;
$(S_t, \bar{S}_t)$ is written for the set of all arcs with one terminal vertex in the vertex set S_t and the other terminal vertex in the complement set $\bar{S}_t$;
A_i is the set of all arcs incident at x_i.

Constraints (10) impose the connectivity of the k-DCT, constraint (11) imposes the degree on vertex x_0, constraints (12), (13) and (14) specify the number of arcs required and constraints (15) impose that the degree of every vertex (other than x_0) in the M-TSP solution is 2. In the above formulation, y was considered as a variable, although it could have been fixed a priori to the constant value M.

Alternatively, if it is known (by any other means) that the maximum number of single-vertex tours (i.e. tours of the type $x_0 - x_i - x_0$) in the M-TSP solution is M_1, then y can be fixed a priori to any integer in the range $M_1 \le y \le M$. The solution to the problem defined by equations (9) to (16) would then remain optimal for any such fixing of y. This is so because at least $y = M_1$ arcs adjacent to x_0 (set S_0) must be removed from the optimal solution to the M-TSP (together with another $M - y$ arcs not adjacent to x_0, i.e. set S_1) to get a k-DCT.

For an M-TSP, with $N \ge M + 1$, we must have $M_1 \le M - 1$ and hence y can be chosen as $M - 1 \le y \le M$. In the above formulation, we have not substituted a constant (M or $M - 1$) for y, because in the M-TSP relaxation that follows the optimum solution is not invariant with y and we wish to choose that value of y ($M_1 \le y \le M$) which maximizes the bound.

Let λ_i, $i = 1, \dots, N$, be non-negative penalties associated with constraints (15). The Lagrangean relaxation of these constraints produces three decomposed problems P, P_0 and P_1 for a given value of y, with the general objective:

$$V(\lambda, y) = \sum_{l=1}^{m} \bar{c}_l w_l - 2 \sum_{i=1}^{N} \lambda_i \tag{17}$$

where $\bar{c}_l = c_l + \lambda_{i_l} + \lambda_{j_l}$, i_l and j_l are the two terminal vertices of arc l, $\lambda_0 = 0$, and where w_l is written for ξ_l, ξ_l^0 and ξ_l^1, respectively, for the three problems.

For a given value of y and λ, let $V(\lambda, y)$ be the optimal solution value of problem P defined by (17), (10), (11), (12) and (16a), let $V^0(\lambda, y)$ be the optimal solution value of problem P_0 defined by (17), (13) and (16b) and let $V^1(\lambda, y)$ be the optimal solution value of problem P_1 defined by (17), (14) and (16c). A lower bound to the M-TSP is therefore:

$$H = \max_{M_1 \le y \le M} \{\max_{\lambda \ge 0} [V(\lambda, y) + V^0(\lambda, y) + V^1(\lambda, y)]\}. \tag{18}$$

The above bound is clearly also valid for the VRP. Moreover, the additional constraints in the VRP can be used to derive a better value of M_1 than the value $M-1$ applicable to the M-TSP, thus increasing the bound further. A value of M_1 for the VRP can be obtained as follows. Let the customers be ordered in decreasing order of the q_i. M_1 single customer routes can at most supply a quantity $\sum_{i=1}^{M_1} q_i$ and the remaining vehicles must, therefore, be able to supply the remaining quantity, i.e.

$$(M - M_1)Q \ge \sum_{i=M_1+1}^{N} q_i. \tag{19a}$$

Similarly, each customer x_i contributes an amount of at least $t_i = u_i + \frac{1}{2}(c_{ii_1} + c_{ii_2})$ to the "cost" of a route, where x_{i_1} is the "nearest" and x_{i_2} the second "nearest" customer to customer x_i. Thus, if the customers are ordered in descending order of the t_i we obtain an expression similar to (19a) as:

$$(M - M_1)T \ge \sum_{i=M_1+1}^{N} t_i. \tag{19b}$$

we can now choose M_1 as the largest value which satisfies both inequalities (19a) and (19b).

3.1.1. Computation of $V(\lambda, y)$

For a given λ and y, problem P is a k-DCT, $(k = 2M - y)$; the problem's solution is T_k^* and has value $V(\lambda, y)$. T_k^* can be computed very easily by calculating the shortest spanning tree (SST) of G [3], and noting the degree d of vertex x_0. If $d = k$ tree T_k^* is derived, if $d < k$ a positive penalty μ is placed on vertex x_0 and the costs c_{0j} are replaced by $c_{0j} - \mu$, and if $d > k$ a negative penalty μ is placed on vertex x_0 and the costs c_{0j} replaced by $c_{0j} - \mu$. In the last two cases the SST is recomputed and the procedure repeated until $d = k$.

It is easy to show that the above procedure is finite. Let us assume that initially $d < k$. For every arc $(x_0, x_i) \notin T_k^*$ compute Δ_i as:

$$\Delta_i = \bar{c}_{0i} - \max_{l \in E_i} [\bar{c}_l] \tag{20}$$

where E_i is the set of arcs of T_k^* on the path from x_0 to x_i, and $\bar{c}_{ij}$ and $\bar{c}_l$ are the arc costs modified by the lagrangean penalties as defined in the last section.

Let $l(i)$ be the arc which produces the above maximum. If μ is then chosen as:

$$\mu = \min_i \left[\Delta_i \mid (x_0, x_i) \notin T_k^* \right], \tag{21}$$

each iteration described above will introduce one extra arc, namely arc (x_0, x_{i^*}) and remove arc $l(i^*)$, where i^* is the value i which minimizes (21).

A similar proof of finiteness can be given when $d > k$.

3.1.2. Computation of $V^0(\lambda, y)$

The solution to problem P_0 involves choosing the y smallest arc costs $\bar{c}_{0i}$, $i = 1, \ldots, N$ to obtain the value $V^0(\lambda, y)$.

However, a better choice of the set S_0 of y arcs can be obtained as follows: Let us associate with each route r in the M-TSP solution two costs $\bar{c}_1^r$ and $\bar{c}_2^r$, these being the costs of the two arcs joining route r to x_0. We will assume $\bar{c}_2^r \geq \bar{c}_1^r$ and note that for one-customer routes $\bar{c}_1^r = \bar{c}_2^r$ since both costs refer to the same arc. In addition, let us order the routes in ascending lexicographical order of the vector $(\bar{c}_2^r, \bar{c}_1^r)$ and renumber them so that route r refers to the rth route in this order. Since for each of the first y routes we can choose for elements of S_0 the longest of the two arcs linking each route to the depot, it is clear that the cost of the set S_0 of arcs can be as high as: $\sum_{r=1}^{y} \bar{c}_2^r$.

Now let all the costs $\bar{c}_{0i}$ of arcs adjacent to the depot be ordered in ascending order with the first M_1 costs in the ordered list repeated once so that they appear twice in the ordered list. Let $h(p)$ correspond to the cost in the pth position of this list. It is then quite clear that for any feasible solution to the M-TSP we have $\bar{c}_2^1 \geq h(2)$, $\bar{c}_2^2 \geq h(4)$, $\bar{c}_2^3 \geq h(6)$, etc., and in general $\bar{c}_2^r \geq h(2r)$ for any $r = 1, \ldots, M$. Hence:

$$L^0(\lambda, y) = \sum_{r=1}^{y} h(2r) \leq \sum_{r=1}^{y} \bar{c}_2^r \tag{22}$$

is a lower bound on the cost of the arcs in set S_0.

$L^0(\lambda, y)$ computed from (22) can be used instead of $V^0(\lambda, y)$ in the computation of the bound from equation (18). This is so because the constraint:

$$\sum_{l \in A_0} \bar{c}_l \xi_l^0 \geq L^0(\lambda, y)$$

could have been added to the M-TSP formulation given by equations (9)–(16). This constraint is redundant for the M-TSP but is no longer redundant in the problem P_0 resulting from the Lagrangean relaxation of the M-TSP. (Indeed, a number of other constraints redundant for the M-TSP but not so for one of the resulting problems P, P_0 or P_1 could have been added to improve on the optimal values V, V^0 and V^1 of these problems.)

3.1.3. *Computation of* $V^1(\lambda, y)$

Let $\bar{c}_i[1]$ be the cost of the shortest arc incident at customer x_i excluding arcs from the depot. Let the quantities $\bar{c}_i[1]$ be ordered in increasing order and let the customers be renumbered so that customer x_i refers to the ith customer in the ordered list. We then have:

$$V^1(\lambda, y) = \sum_{i=1}^{M-y} \bar{c}_i[1].$$

3.1.4. *Calculation of the bound* (*LB*0)

In this section we give a procedure for the calculation of the lower bound from the k-DCT—referred to hereafter as LB0.

Step 0 (initialization). Set the best lower bound $z^*_{\mathrm{L}} = 0$. Let z^*_{U} be the value of the best solution so far. Set $y = M_1$.

Step 1 (initialization). Set KOUNT = 1; $\lambda_i = 0$, $i = 0, \ldots, N$ and $k = 2M - y$.

Step 2 (shortest k-DCT). Calculate the shortest spanning k-DCT T^*_k as mentioned above.

Step 3 (additional arcs). Compute $L^0(\lambda, y)$ and $V^1(\lambda, y)$ and let S_0 and S_1 be the sets of arcs which produced the values of $L^0(\lambda, y)$ and $V^1(\lambda, y)$.

Step 4 (graph G). Form the graph $G = (X, T^*_k \cup S_0 \cup S_1)$ where T^*_k is used to imply the set of arcs in the tree T^*_k. Let z_{L} = sum of the costs of all arcs in G. z_{L} is a lower bound on the value of the solution to the VRP. If $z^*_{\mathrm{L}} < z_{\mathrm{L}}$ set $z^*_{\mathrm{L}} = z_{\mathrm{L}}$. If $z^*_{\mathrm{L}} \geq z^*_{\mathrm{U}}$ stop (go to back-tracking step in main algorithm), else if $z^*_{\mathrm{L}} < z^*_{\mathrm{U}}$ go to Step 5. If $z^*_{\mathrm{L}} \geq z_{\mathrm{L}}$ and KOUNT = maximum number of iterations allowed go to step 7. Else KOUNT = KOUNT + 1, go to Step 5.

Step 5 (penalties). If the degree d_i of vertex x_i with respect to graph G is 2 ($i = 1, \ldots, N$) and $d_0 = 2M$, stop. (z^*_{L} is the best lower bound that can be obtained by this procedure.) Otherwise compute penalties:

$$\lambda_i = \lambda_i + \alpha \frac{(z^*_{\mathrm{U}} - z_{\mathrm{L}})}{\left[\sum_{j=0}^{N} (d_j - \beta_j)^2\right]^{1/2}} (d_i - \beta_i), \quad i = 0, \ldots, N$$

where $\beta_i = 2$ if $i \neq 0$ and $\beta_i = 2M$ for $i = 0$ and α is a constant.

Step 6 (cost matrix $[c_{ij}]$). Modify the cost matrix $[c_{ij}]$ as:

$$c_{ij} = c_{ij} + \lambda_i + \lambda_j.$$

Go to Step 2.

Step 7 (update y). If $y = M$ stop, else set $y = y + 1$; go to Step 1.

3.2. Minimum q-routes

Houck et al. [17] introduced n-paths for the TSP: these are special cases of dynamic programming relaxation, procedures for obtaining bounds to combinatorial problems and are described in [7]. Here we use such procedures to develop bounds for the VRP.

Let W be the set of all possible loads (quantities) that could exist on any vehicle route, i.e.

$$W = \left\{ q \;\middle|\; \sum_{i=1}^{N} q_i \xi_i = q \leq Q, \text{ for some } \xi, \xi_i \in \{0, 1\} \right\}.$$

Let the elements of W be ordered in ascending order and let $w = |W|$. We will denote by $q(l)$ the value of the lth element of W and by $\pi(q)$ that l^* so that $q(l^*) = q$.

The total load on a path $\Phi = (x_0, x_{i_1}, x_{i_2}, \ldots, x_{i_k})$ is defined as $\sum_{x_i \in \Phi - \{x_0\}} q_i$. (Note that Φ is not necessarily a simple path.)

Let $f_l(x_i)$ be the cost of the least cost path from x_0 to $x_i \neq x_0$ with total load $q(l)$. Fig. 2 shows a possible such path called a *q-path*. A q-path with the additional arc (x_i, x_0) is called a terminal q-route and has cost $f'_l(x_i) = f_l(x_i) + c_{i0}$.

Let the solution to the VRP consist of the M routes:

$$R_1 = (x_0, x_{i_1}, \ldots, x_{i_{p1}}, x_0) \quad \text{with load } q(l_1);$$

$$R_2 = (x_0, x_{i_2}, \ldots, x_{i_{p2}}, x_0) \quad \text{with load } q(l_2);$$

$$R_M = (x_0, x_{i_M}, \ldots, x_{i_{pM}}, x_0) \quad \text{with load } q(l_M).$$

If the cost of a route R_k is denoted by $C(R_k)$ then

$$f'_{l_k}(x_{i_{pk}}) \equiv f_{l_k}(x_{i_{pk}}) + c_{i_{pk}'0} \leq C(R_k), \qquad k = 1, \ldots, M.$$

The path corresponding to $f_l(x_i)$ is not necessarily simple but it is not easy to impose the condition that no vertex is visited by the path more than once. On the other hand it is quite easy to impose the restriction that the path should not contain loops formed by three consecutive vertices. (For example, the path in Fig. 2 contains the loop x_k, x_r, x_k.) With this restriction imposed, a better lower bound can be calculated.

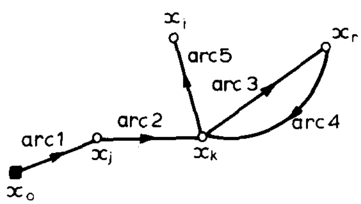

Fig. 2.

3.2.1. q-paths with no loops

Let $p_l(x_i)$ be the vertex just prior to x_i on the path corresponding to $f_l(x_i)$.

Let $\phi_l(x_i)$ be the least cost path from the depot to x_i with load $q(l)$ and with $\bar{\pi}_l(x_i) \neq p_l(x_i)$, where $\bar{\pi}_l(x_i)$ is the vertex just prior to x_i on the path corresponding to $\phi_l(x_i)$.

Fig. 3 shows two possible paths corresponding to $f_l(x_i)$ and $\phi_l(x_i)$.

For a given value of l, let $g(x_j, x_i)$ be the cost of the least cost path from x_0 to x_i with x_j just prior to x_i and without loops.

Then:

$$g(x_j, x_i) = \begin{cases} f_{l'}(x_j) + c_{j,i}, & \text{if } p_{l'}(x_j) \neq x_i \\ \phi_{l'}(x_j) + c_{j,i}, & \text{otherwise} \end{cases} \tag{23}$$

where l' is such that $q(l') = q(l) - q_i$.

Given the function g computed from (23), functions f and ϕ can now be computed for the given l as follows:

$$\left.\begin{aligned} f_l(x_i) &= \min_{x_j} [g(x_j, x_i)], \\ p_l(x_i) &= x_j^* \end{aligned}\right\} \tag{24a}$$

where x_j^* is the value of x_j corresponding to the above minimum;

$$\left.\begin{aligned} \phi_l(x_i) &= \min_{x_k \neq p_l(x_i)} [g(x_k, x_i)], \\ \bar{\pi}_l(x_i) &= x_k^* \end{aligned}\right\} \tag{24b}$$

where x_k^* is the value of x_k corresponding to the above minimum.

From the above expression it is clear that the path corresponding to $f_l(x_i)$ has no end loops.

The initialization of the functions f, ϕ, p and $\bar{\pi}$ is as follows:

$$f_l(x_i) = \phi_l(x_i) = \infty \quad \text{for } l \text{ such that } q(l) \neq q_i,$$

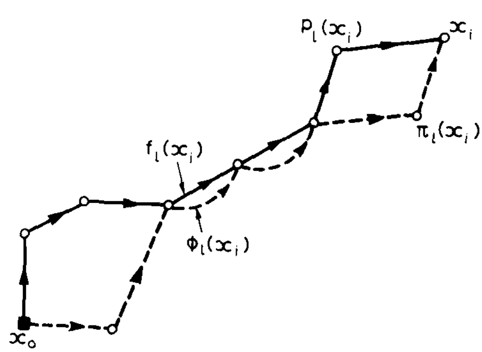

Fig. 3.

$$\left.\begin{array}{l} f_l(x_i) = c_{0,i}; \quad p_l(x_i) = x_0 \\ \phi_l(x_i) = \infty \end{array}\right\} \quad \text{for } l \text{ such that } q(l) = q_i.$$

3.2.2. *Through q-routes*

Let $\psi_l(x_i)$ be the value of the least cost route, without loops, starting from the depot, *passing through* x_i and finishing back at the depot with a total load $q(l)$. Such a route will be called a *through q-route.* $\psi_l(x_i)$ must be composed of either the two best q-paths to x_i whose total loads add up to $q'(l) \equiv q(l) + q_i$, or a best path and a second-best path to x_i whose total loads add up to $q'(l)$.

$\psi_l(x_i)$ can then be computed as follows:

$$\psi_l(x_i) = \underset{\substack{q_i \le q \le \frac{1}{2}q'(l) \\ q \in W}}{\operatorname{Min}} \begin{bmatrix} f_{\pi(q)}(x_i) + f_{\pi(q'(l)-q)}(x_i) \\ \text{if } p_{\pi(q)}(x_i) \neq p_{\pi(q'(l)-q)}(x_i), \\ \\ \operatorname{Min}\,[f_{\pi(q)}(x_i) + \phi_{\pi(q'(l)-q)}(x_i);\ \phi_{\pi(q)}(x_i) + f_{\pi(q'(l)-q)}(x_i)] \\ \text{if } p_{\pi(q)}(x_i) = p_{\pi(q'(l)-q)}(x_i). \end{bmatrix}$$

We note here that the computational effort involved in computing the q-paths is linearly related to w. Thus, if w is large this operation can be quite time consuming.

3.3. *Calculation of bounds from the q-routes*

Let the total number of feasible single routes possible in the VRP be indexed $r = 1, \ldots, \hat{r}$. Let the index set of customers in route r be M_r, the cost of the route be d_r and the total load of the route be $K_r = \sum_{i \in M_r} q_i$. Let N_i be the index set of routes visiting customer x_i.

$$\text{Let } y_r = \begin{cases} 1 & \text{if route } r \text{ is in the optimal VRP solution,} \\ 0 & \text{otherwise.} \end{cases}$$

The VRP is then:

$$\operatorname{Min} \sum_{r=1}^{\hat{r}} d_r y_r, \tag{25}$$

$$\text{s.t.} \quad \sum_{r \in N_i} y_r = 1, \quad i = 1, \ldots, N, \tag{26}$$

$$\sum_{r=1}^{\hat{r}} y_r = M, \tag{27}$$

$$y_r \in \{0, 1\}. \tag{28}$$

3.3.1. *Bound* LB1

Let us renumber the set of routes $r = 1, \dots, \hat{r}$, by partitioning the set into N blocks, where block i ($i = 1, \dots, N$) is associated with the index i of the last customer x_i on the routes in the block. Within a block the routes are indexed by $s = 1, \dots, \hat{r}_i$.

$$\text{Let } \xi_{is} = \begin{cases} 1 & \text{if route } s \text{ of block } i \text{ is in the solution,} \\ 0 & \text{otherwise.} \end{cases}$$

The above formulation (equations (25)–(28)) becomes:

$$\text{Min} \quad \sum_{i=1}^{N} \sum_{s=1}^{\hat{r}_i} d_{is} \xi_{is}, \tag{25'}$$

$$\text{s.t.} \quad \sum_{(i,s) \in \bar{N}_j} \xi_{is} = 1, \quad j = 1, \dots, N, \tag{26'}$$

$$\sum_{i=1}^{N} \sum_{s=1}^{\hat{r}_i} \xi_{is} = M, \tag{27'}$$

$$\xi_{is} \in \{0, 1\}. \tag{28'}$$

where the ordered pair (i, s) is used for the label of the route s of block i, and $\bar{N}_j$ is the set of ordered pairs (i, s) identifying routes containing customer x_j. d_{is} is the cost of route (i, s).

The problem defined by (25′) to (28′) can be relaxed by replacing constraints (26′) with the following set of weaker constraints:

$$\sum_{s=1}^{\hat{r}_i} \xi_{is} \le 1, \quad i = 1, \dots, N, \tag{26'a}$$

$$\sum_{i=1}^{N} \sum_{s=1}^{\hat{r}_i} K_{is} \xi_{is} = Q_T \tag{26'b}$$

where K_{is} is the load of route (i, s) and $Q_T = \sum_{i=1}^{N} q_i$.

The problem defined by (25′), (26′a), (26′b), (27′) and (28′) can now be reduced by observing that for a given i and a given load q on a route, route (i, s^*) dominates all other routes (i, s), $s = 1, \dots, \hat{r}_i$, $s \ne s^*$, if

$$d_{is^*} \le d_{is} \quad \text{and} \quad K_{is} = K_{is^*} = q.$$

Thus, for each i and each $q \in W$ only one route need be considered. Let us call this route (i, l), with $l = \pi(q)$. There are now w routes to consider for each i.

The relaxed problem becomes:

$$\text{Min} \quad \sum_{i=1}^{N} \sum_{l=1}^{w} d_{il} \xi_{il}, \tag{29}$$

$$\text{s.t.} \quad \sum_{l=1}^{w} \xi_{il} \leq 1, \quad i = 1, \dots, N, \tag{30}$$

$$\sum_{i=1}^{N} \sum_{l=1}^{w} q(l) \cdot \xi_{il} = Q_T, \tag{31}$$

$$\sum_{i=1}^{N} \sum_{l=1}^{w} \xi_{il} = M, \tag{32}$$

$$\xi_{il} \in \{0, 1\}. \tag{33}$$

A lower bound on the value d_{il} (i.e. the cost of the minimum cost route with total load $q(l)$ and having x_i as the last customer visited) is the cost $m_{il} \equiv f_l'(x_i)$ of the least cost q-route with $q = q(l)$. Hence a bound LB1 can be obtained as the value of the solution of problem B1 defined by the objective function:

$$\sum_{i=1}^{N} \sum_{l=1}^{w} m_{il} \xi_{il} \tag{29'}$$

subject to constraints (30)–(33).

Problem B1 can be solved conveniently by dynamic programming as follows.

Let $h_i(b, a)$ be the optimum solution to problem B1 with the right-hand side of (31) replaced by b and the right-hand side of (32) replaced by a, and with $\xi_{jl} = 0$ for $j > i$, $l = 1, \dots, w$. The function $h_i(b, a)$ can be computed recursively from

$$h_i(b, a) = \mathrm{Min}\left[h_{i-1}(b, a), \mathop{\mathrm{Min}}_{q \in W}\{h_{i-1}(b-q, a-1) + m_{i, \pi(q)}\} \right] \tag{34}$$

for: $a = 2, \dots, M$; $i = a, \dots, N - M + a$; $Q_T - (M-a) \cdot Q \leq b \leq \min[a \cdot Q, Q_T]$. The function is initialized as:

$$h_i(b, 1) = \begin{cases} m_{i,\pi(b)} & \text{if } b \in W, \\ \infty & \text{if } b \notin W. \end{cases}$$

The lower bound to the VRP is then

$$\mathrm{LB1} = h_N(Q_T, M). \tag{35}$$

3.3.2. *Bound* LB2

We again consider the VRP formulation given by equations (25)–(28).

Let us substitute y_r by:

$$y_r = \frac{1}{K_r} \sum_{i \in M_r} \zeta_{ir} q_i. \tag{36}$$

The formulation of the VRP given by equations (25)–(28) now becomes

$$\mathrm{Min} \quad \sum_{r=1}^{\hat{r}} \frac{d_r}{K_r} \sum_{i \in M_r} \zeta_{ir} q_i, \tag{37}$$

$$\text{s.t.} \quad \sum_{r \in N_i} \zeta_{ir} = 1, \quad i = 1, \dots, N, \tag{38}$$

$$\zeta_{ir} = \frac{1}{K_r} \sum_{j \in M_r} \zeta_{jr} q_j, \quad i \in M_r,\ r = 1, \dots, \hat{r}, \tag{39}$$

$$\sum_{r=1}^{\hat{r}} \sum_{i \in M_r} \frac{q_i}{K_r} \zeta_{ir} = M, \tag{40}$$

$$\zeta_{ir} \in \{0, 1\}. \tag{41}$$

Constraints (39) ensure that $\zeta_{ir} = 1$ if and only if $\zeta_{jr} = 1\ \forall j \in M_r$ and hence $y_r = 1$. Thus, constraints (38) correspond to constraints (26).

Let the above problem be relaxed by (i) removing constraints (39) and (ii) by replacing set M_r for route r by the complete set $I = \{1, \dots, N\}$. The resulting relaxed problem can be somewhat strengthened by adding the constraint

$$\sum_{r=1}^{\hat{r}} \sum_{i=1}^{N} q_i \zeta_{ir} = Q_T, \tag{42}$$

which was redundant for the formulation given by equations (37)–(41) but which is no longer redundant for the new relaxed problem.

In the relaxed problem defined by equations (37), (38), (40), (41) and (42) (with M_r replaced by I), only one route need by considered for each customer x_i and for each possible value of load q on the route ($q \in W$). This is clear from the fact that if two routes r_1 and r_2 both contain customer x_i and have loads $K_{r_1} = K_{r_2} = q$, then if $d_{r_1} \leq d_{r_2}$, route r_1 dominates route r_2 in the relaxed problem. Let us call the undominated route (i, l) with $l = \pi(q)$. We will denote the cost of this route by d_{il}. There are now w routes to consider for each i. The relaxed problem now becomes

$$\text{Min} \quad \sum_{i=1}^{N} \sum_{l=1}^{w} \bar{d}_{il} \zeta_{il}, \tag{43}$$

$$\text{s.t.} \quad \sum_{l=1}^{w} \zeta_{il} = 1, \quad i = 1, \dots, N, \tag{44}$$

$$\sum_{i=1}^{N} \sum_{l=1}^{w} \frac{q_i}{q(l)} \zeta_{il} = M, \tag{45}$$

$$\sum_{i=1}^{N} \sum_{l=1}^{w} q_i \zeta_{il} = Q_T, \tag{46}$$

$$\zeta_{il} \in \{0, 1\}, \tag{47}$$

where $\bar{d}_{il} = d_{il} q_i / q(l)$.

Note that if no route passing through i with load $q(l)$ exists, then $d_{il} = \infty$. $d_{il} / q(l)$ represents the marginal cost of supplying customer i, on a route with load $q(l)$, with a unit quantity and hence $\bar{d}_{il}$ is the "cost contribution" of customer i.

It is quite apparent that the cost $\psi_l(x_i)$ of the minimum cost through q-route passing through customer x_i and having load $q(l)$ is a lower bound on d_{il}. Thus, the solution of problem B2 defined by the objective function

$$\text{Min} \quad \sum_{i=1}^{N} \sum_{l=1}^{w} b_{il} \zeta_{il} \tag{48}$$

and constraints (44)–(47), where $b_{il} = \psi_l(x_i) \cdot q_i / q(l)$, is a lower bound to the VRP. b_{il} is a lower bound on $\bar{d}_{il}$ and is obtained by relaxing the restrictions that in a feasible solution the degree of every vertex is 2.

In Section 3.2 we gave a procedure for computing $\psi_l(x_i)$ for every $i = 1, \dots, N$ and every $l = 1, \dots, w$.

3.3.3. *The computation of bound* LB2

A simple bound LB2 can be computed by ignoring constraints (45) and (46) and minimizing (48) subject to only (44) and (47). The resulting bound is:

$$\text{LB2} = \sum_{i=1}^{N} \min_{l=1,\dots,w} [b_{il}]. \tag{49}$$

A better bound LB2′ can be derived by considering a Lagrangean relaxation of constraints (45) and (46) in the usual way.

3.3.4. *Penalty procedures for improving the bounds*

The solution corresponding to bound LB1 can be obtained by backtracking using recursions (34), (24a) and (24b). This solution represents a graph, such as the one shown in Fig. 4, which shows three q-routes and since some of these routes are not simple and the routes are not necessarily pairwise vertex disjoint the resulting graph G in Fig. 4 contains vertices (customers) with degrees with respect to G greater than 2 and some customers with degree 0.

Let us now place penalties λ_i $(i = 1, \dots, N)$ on the vertices x_i computed from an expression similar to that used for the M-DCT bound, i.e.

$$\lambda_i = \lambda_i + \alpha \cdot \frac{z_{\mathrm{U}}^* - z_{\mathrm{L}}}{\left[\sum_{j=1}^{N} (d_j - 2)^2\right]^{1/2}} (d_i - 2) \tag{50}$$

and modify the cost matrix $[c_{ij}]$ as: $c_{ij} = c_{ij} + \lambda_i + \lambda_j$.

The functions f, p, ϕ and $\bar{\pi}$ can now be computed for the new matrix $[c_{ij}]$, etc. At the end of the tth iteration:

$$\text{LB1} = h_N(Q_T, M) - 2 \sum_{i=1}^{N} \lambda_i \tag{51}$$

where $h_N(Q_T, M)$, computed from recursion (34) with respect to the modified matrix c_{ij}, is a bound to the VRP.

For the case of bound LB2, we can find—by backtracking—the q-route G_i

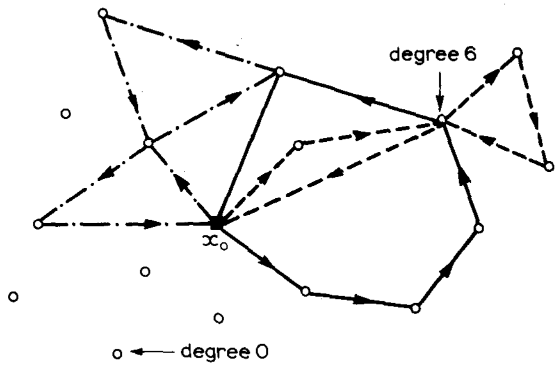

Fig. 4.

corresponding to $\psi_{l_i}(x_i)$, where l_i is the value of l producing the minimum in the expression

$$\operatorname*{Min}_{l=1,\dots,w}\left[\frac{\psi_l(x_i)}{q(l)}\right].$$

Let δ_k^i be the degree of x_k with respect to G_i. Compute

$$d_k=\sum_{i=1}^{N}\delta_k^i q_i/q(l_i).$$

The degrees d_k should all be equal to 2 in any feasible solution to the VRP.

An iteration of the ascent procedure now involves computing the λ_i from expression (50), modifying the costs c_{ij}, recomputing f, p, ϕ, $\bar{\pi}$ and ψ and so on. At the end of the tth iteration

$$\mathrm{LB2}=\sum_{i=1}^{N}\min_{l=1,\dots,w}\left[\frac{\psi_l(x_i)}{q(l)}\right]\cdot q_i-2\sum_{i=1}^{N}\lambda_i \tag{52}$$

is the bound to the VRP.

3.4. Computational aspects of bound calculations

The value of bound LB0 obtained from the minimum DCT as a function of the number of penalty iterations in the ascent procedure is given in Fig. 5. Also shown in Fig. 5 is the bound LB2 obtained from the q-routes. Fig. 5. refers to a 21-customer VRP that has previously appeared in the literature as test problem 3 in reference [9], but without the constraints on maximum route distance, and with customer delivery times assumed zero. The value of the optimum solution to this problem is 374.3 units of length involving 4 vehicles.

As a means of comparison, the solution to the unconstrained 4-TSP is of value 315.

In the branch and bound procedure it is clear that backtracking can occur as

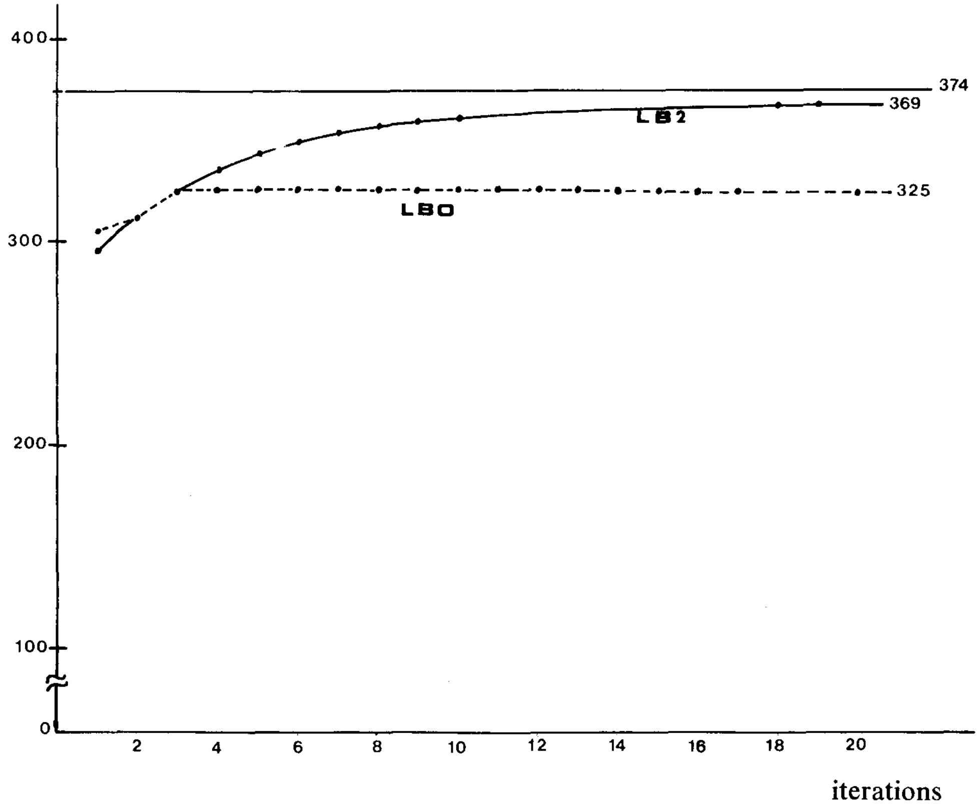

Fig. 5.

soon as the bound at some iteration exceeds z_U^* (the value of the best solution found so far). In order to avoid unnecessary computations during the iterations we use a fitted polynomial describing the value of the bound as a function of the iteration number to predict the expected number of iterations necessary for the bound to exceed z_U^*. If this number is very high or if the final expected value of the bound is below z_U^*, the ascent iterations are terminated and branching takes place.

Let T be the maximum number of iterations allowed and let:

$$\eta = \frac{\mathrm{LB}(T)}{\mathrm{LB}(1)},$$

where $\mathrm{LB}(t)$ is the value of the bound after iteration (t). Let η be computed for the initial node of the branch and bound tree. If, at some arbitrary node v of the branch and bound tree $\eta \mathrm{LB}^v(1) < z_U^*$ (where $\mathrm{LB}^v(1)$ is the bound after the first iteration at node v) then no further iterations are made to try to improve the bound.

If $\eta \mathrm{LB}^v(1) \geq z_U^*$ a block of t_1 iterations (fixed a priori) is made, a polynomial is fitted to the values $\mathrm{LB}^v(t)$, $t = 1, \ldots, t_1$ and that value of t^* for which $\mathrm{LB}^v(t^*) = z_U^*$ is calculated. If $t^* \geq T$ the ascent is abandoned. If $t^* < T$ another block of t_1 iterations is performed, a new polynomial is fitted to $\mathrm{LB}^v(t)$, $t = 1, \ldots, t_1$, etc.

4. Branching strategies

The computational results shown in Section 7 were derived by using two possible branching strategies as follows.

4.1. *Branch on an arc* (*Rule* BB1)

With this type of branching, an arc (x_i, x_j) is chosen for branching at a node of the search tree in order to extend a partially completed route $(x_0, x_k, \ldots, x_i)$. The alternative branching is to reject arc (x_i, x_j) as a possible extension of the route.

4.2. *Branch on a route* (*Rule* BB2)

With this type of branching a node of the search tree corresponds to a single feasible route S_j. The state at this node (say at stage h) is represented by the ordered list:

$$L = \{S_{j_1}(x_{i_1}), S_{j_2}(x_{i_2}), \ldots, S_{j_h}(x_{i_h})\},$$

where $S_{j_r}(x_{i_r})$ is a feasible single route (the j_rth) which includes a specified customer x_{i_r} and other customers which are not already included in the previous routes $S_{j_1}(x_{i_1}), \ldots, S_{j_{r-1}}(x_{i_{r-1}})$. In this section we will use the word route to imply the unordered set of customers forming the route. The set $\pi(x_{i_r})$, say, of all feasible routes $S_j(x_{i_r})$ passing through a customer x_{i_r}, is generated by forming all possible sets of customers (including customer x_{i_r}) so that the total demand of customers within a set is less than or equal to Q. The cost of the route through the jth such set $S_j(x_{i_r})$ is then computed by solving a corresponding TSP. The state represented by list L is shown diagrammatically in Fig. 6 where

$$F_h = X - \sum_{r=1}^{h} S_{j_r}(x_{i_r})$$

is used for the set of "free" (i.e. as yet unrouted) customers following stage h.

Once the bottom of the tree is reached, say at stage M when $F_M = \emptyset$, the list L contains a solution to the VRP consisting of M routes.

A forward branching from some stage h involves the choice of a customer $x_{i_{h+1}} \in F_h$ and the generation of a list $\pi(x_{i_{h+1}})$ of all feasible single routes passing through this customer. It is quite apparent that the smaller the number of branching possibilities at any stage h the more efficient the tree search, and it is therefore obvious that $x_{i_{h+1}}$ should be chosen from F_h so as to make the list π

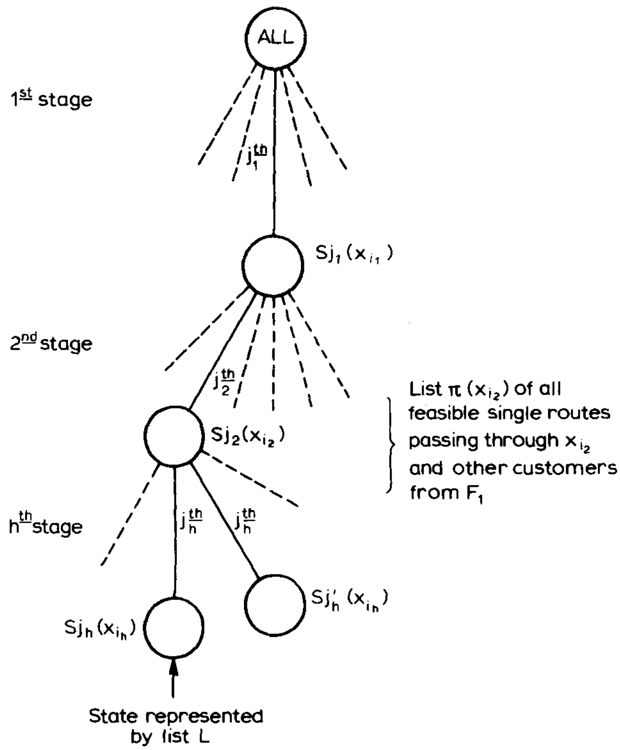

Fig. 6.

$(x_{i_{h+1}})$ as short as possible. This would tend to be the case if $x_{i_{h+1}}$ is chosen to be an "isolated" customer far from the depot, or if it has a large demand $q_{i_{h+1}}$.

It is computationally expensive simply to generate all routes passing through x_{i_h} directly and then eliminate infeasible ones by imposing various tests. It is, therefore, desirable to incorporate some of these reduction tests into the route-generating process and thus eliminate at an early stage routes that would have otherwise been found to be infeasible at a later time. Some of these tests are facilitated by the considerations in the next section.

5. Reduction and dominance tests

Consider the matrix $B = [b_{il}]$ introduced during the calculation of bound LB2. Let l_i^* be the value of l for which

$$b_{il_i^*} = \min_l [b_{il}].$$

Reduction R1. If for some i and l:

$$\mathrm{LB2} - b_{il_i^*} + b_{il} \geq z_U^* \tag{53}$$

then element b_{il} of matrix B can be set to ∞, where LB2 is given by equation (52).

Reduction R2. Let us suppose that in some solution S, customer x_k is supplied by a route with load $q(l)$. A lower bound on the value of S is then:

$$\sum_{i \neq k} b_{il_i^*} + \psi_l(x_k) - V - 2\sum_i \lambda_i$$

where V is the maximum contribution to bound LB2 of a set of customers Y whose total demand is $q(l) - q_k$, i.e.

$$V = \operatorname*{Max}_{Y \subset X - \{x_k\}} \left[\sum_{x_i \in Y} b_{il_i^*} \,\Big|\, \sum_{x_i \in Y} q_i = q(l) - q_k \right]. \tag{54}$$

An upper bound $\hat{V}$ on V can be computed by ignoring the integrality restrictions in expression (24) above in exactly the same way as the Dantzig bound is calculated for the knapsack problem. If

$$\sum_{i \neq k} b_{il_i^*} + \psi_l(x_k) - \hat{V} - 2\sum_i \lambda_i \geq z_U^*$$

then element b_{kl} of matrix B can be set to ∞.

5.1. Inclusion and exclusion of customer pairs

5.1.1. Exclusion

We will now assume that B has been reduced by applying reductions R1 and R2 above. Let us consider two customers x_k and x_j. If these two customers are on the same route (say R) in the optimal solution, then the cost $C(R)$ of this route is

$$C(R) \geq c_{0k} + c_{kj} + c_{j0}$$

and the maximum load on route R must be $\leq Q$. Hence, if x_k and x_j are on the same route,

$$V = \sum_{i \neq k, j} b_{il_i^*} + (c_{0k} + c_{kj} + c_{j0}) \cdot (q_j + q_k)/Q - 2\sum_i \lambda_i$$

is a lower bound on the optimal solution. Thus, if $V \geq z_U^*$ it follows that x_k and x_j cannot be on the same route. We will call $[x_k, x_j]$ a prohibited pair of customers and set element (k, j) of a matrix $P = [p_{kj}]$ to -1.

5.1.2. Inclusion

Let us again consider two customers x_{α_1} and x_{α_2} and form the reduced VRP without these two customers. The lower bound on the reduced VRP is at least $\sum_{i \neq \alpha_1, \alpha_2} b_{il_i^*} - 2\sum_{i \neq \alpha_1, \alpha_2} \lambda_i$, because if the route corresponding to $\psi_{l_i^*}(x_i)$ contains customers x_{α_1} or x_{α_2} or both, then the route not containing these customers must have greater or equal value. The addition of customers x_{α_1} and x_{α_2} to the reduced VRP will increase the cost of the VRP solution by at least

$$V = \sum_{i=1,2} \min_{\beta_i, \gamma_i} [c_{\alpha_i\beta_i} + c_{\alpha_i\gamma_i} - c_{\beta_i\gamma_i} \mid q_{\alpha_i} + q_{\beta_i} + q_{\gamma_i} \leq Q].$$

If arc $(x_{\alpha_1}, x_{\alpha 2})$ is excluded from the solution, then the addition of x_{α_1} and x_{α_2} to the reduced VRP will increase the cost of the solution by an amount V' given by the same expression as above but with $x_{\beta_1}, x_{\gamma_1} \neq x_{\alpha_2}$ and $x_{\beta_2}, x_{\gamma_2} \neq x_{\alpha_1}$.

Hence a lower bound to the VRP with arc $(x_{\alpha_1}, x_{\alpha_2})$ excluded is

$$\sum_{i \neq \alpha_1, \alpha_2} b_{il_i^*} - 2 \sum_{i=1}^{N} \lambda_i + V',$$

and if this bound is greater than or equal to z_U^* it follows that arc $(x_{\alpha_1}, x_{\alpha_2})$ must be in the solution in which case we set element $p_{\alpha_1\alpha_2} = 1$.

All elements of matrix P not set to ± 1 by the exclusion and inclusion tests are assumed to be set to 0.

Matrix P can be used to reduce the number of branchings in the tree search algorithms for both the type of branching based on the arcs (BB1) and the branching based on a route (BB2).

5.2. Feasibility and dominance

We will now describe some simple feasibility and dominance tests that could be used to eliminate nodes of the branch and bound tree.

(i) When using branching scheme BB1 the representations of state include a partially finished route. If this route can be improved by a 3-optimal local optimization procedure [18] the corresponding node can be rejected.

(ii) When using branching scheme BB2 let $S_{j_h^\alpha}(x_{i_h})$ and $S_{j_h^\beta}(x_{i_h})$ be two nodes (routes) emanating from the same vertex at level $(h-1)$.

Let UB (F_h^α) be an upper bound and LB (F_h^α) be a lower bound on the total cost needed to supply the customers in F_h^α,

$$C[S_{j_h^\beta}(x_{i_h})] + \mathrm{UB}\,(F_h^\beta) \leq C[S_{j_h^\alpha}(x_{i_h})] + \mathrm{LB}\,(F_h^\alpha),$$

where $C(S)$ is the cost of the single feasible route S, then node $S_{j_h^\beta}(x_{i_h})$ dominates node $S_{j_h^\alpha}(x_{i_h})$ and the latter can be removed from the list $\pi(x_{i_h})$.

Both for the branching scheme BB1 (at a step when a route currently being formed is completed) and for scheme BB2:

(iii) If it could be shown that the remaining free vehicles are not capable (e.g. because of insufficient capacity) of supplying the unrouted customers, the node can be rejected.

(iv) If for two nodes α and β, the cost of the routes represented by α is less than or equal to the cost of the routes represented by β and the VRP remaining at node α is the same problem or a subproblem of the VRP remaining at node β, then β is dominated by α, and node β can be rejected.

6. Example

We will illustrate the procedures described in the earlier sections by means of an example involving $N = 10$ customers and $M = 4$ vehicles. The distance matrix $[c_{ij}]$ and the customer demands are given below; 0 refers to the depot. The vehicle capacity is 24 units.

Distance matrix C_{ij}

	0	1	2	3	4	5	6	7	8	9	10
0	—	24.1	27.6	17.2	23.3	11.1	16.0	7.0	20.2	9.8	22.0
1	24.1	—	21.2	9.2	16.1	26.0	28.1	31.2	38.0	33.0	46.2
2	27.6	21.2	—	15.5	35.3	20.2	40.4	32.7	28.0	37.2	45.8
3	17.2	9.2	15.5	—	20.0	17.0	26.1	24.0	28.8	26.9	39.0
4	23.3	16.1	35.3	20.0	—	31.3	17.0	29.2	42.7	28.1	42.4
5	11.1	26.0	20.2	17.0	31.3	—	27.1	13.4	12.0	18.6	25.6
6	16.0	28.1	40.4	26.1	17.0	27.1	—	18.0	35.5	14.4	27.6
7	7.0	31.2	32.7	24.0	29.2	13.4	18.0	—	18.0	5.3	15.0
8	20.2	38.0	28.0	28.8	42.7	12.0	35.5	18.0	—	23.0	22.6
9	9.8	33.0	37.2	26.9	28.1	18.6	14.4	5.3	23.0	—	14.3
10	22.0	46.2	45.8	39.0	42.4	25.6	27.6	15.0	22.6	14.3	—

Demands

$i =$	1	2	3	4	5	6	7	8	9	10
$q_i =$	1	5	6	12	13	13	3	9	21	10

(A) We will first use a tree search procedure using branching rule BB1 with bound LB0.

A heuristically obtained initial feasible solution gives an upper bound of 225.0. The lower bound LB0 computed for the root node of the branch and bound tree is 211.0, computed after 30 penalty iterations. The complete tree is shown in Fig. 7. The number next to a node is the bound LB0 corresponding to that node; the number in each node is the number of the customer to be visited next on the emerging route. A bar above the customer number represents negation (i.e. that customer is not the next one to be visited). Fig. 7 represents a depth first procedure and the tree is generated by "branching to the left" whenever possible. Thus, the first solution found during the search is at the 23rd generated node (marked with an asterisk) and another 26 nodes are generated to prove that the solution is optimal. The optimal solution is:

route 1: 0–9–7–0
route 2: 0–5–8–0
route 3: 0–4–1–3–2–0
route 4: 0–6–10–0

and the optimal value is 222.7.

(B) We will now illustrate the use of bound LB2 on the same example. In attempting to calculate a lower bound for the route node of the branch and bound tree using expression (22), with penalties calculated according to expression (20), and allowing a maximum of 20 penalty iterations, a bound of 222.7 was derived at the 7th iteration. The routes $\psi_l(x_i)$ which produced this bound from equation (22) represented a feasible solution to the VRP, i.e.

$\psi_{l^*}(9)$ and $\psi_{l^*}(7)$ corresponded to route 1 above;
$\psi_{l^*}(5)$ and $\psi_{l^*}(8)$ corresponded to route 2 above;
$\psi_{l^*}(6)$ and $\psi_{l^*}(10)$ corresponded to route 3 above;
$\psi_{l^*}(4)$, $\psi_{l^*}(1)$, $\psi_{l^*}(3)$ and $\psi_{l^*}(2)$ corresponded to route 4 above.

The procedure is therefore terminated having obtained the optimum solution with no branching of the branch and bound tree being necessary.

(C) We will now illustrate the use of bound LB0 with the branching rule of Section 4.2. The complete tree search is shown in Fig. 8. At the first level three nodes are generated, each node representing a feasible route through a chosen customer (customer 9 for level 1). (The starting upper bound is again assumed to be 225.0.) Following a search where the branching is from the node with the lowest computed bound at every stage, the tree shown in Fig. 8 is generated. In this figure the bound is shown next to the node and the customers forming the corresponding route are shown inside the node. Obviously, with four routes, there are only four levels to this tree having a total of 22 nodes.

7. Computational results

In this section we investigate the computational performance of the three algorithms illustrated in the example, also including the dominance tests of Section 5. Ten problems are used for the tests ranging from $N = 10$ to $N = 25$

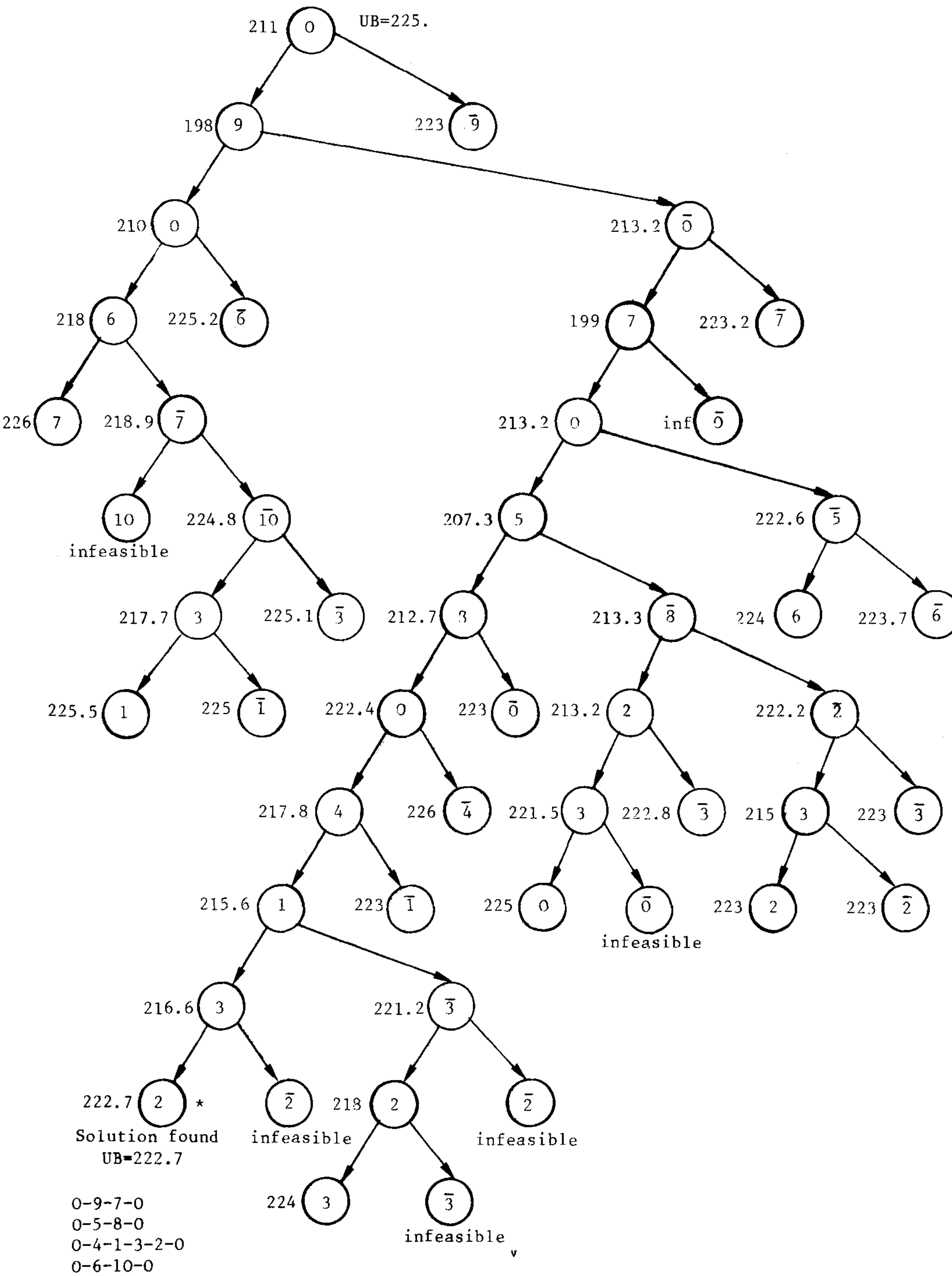

Fig. 7. Branch and bound tree for the example using bound LBO and branching rule BB1.

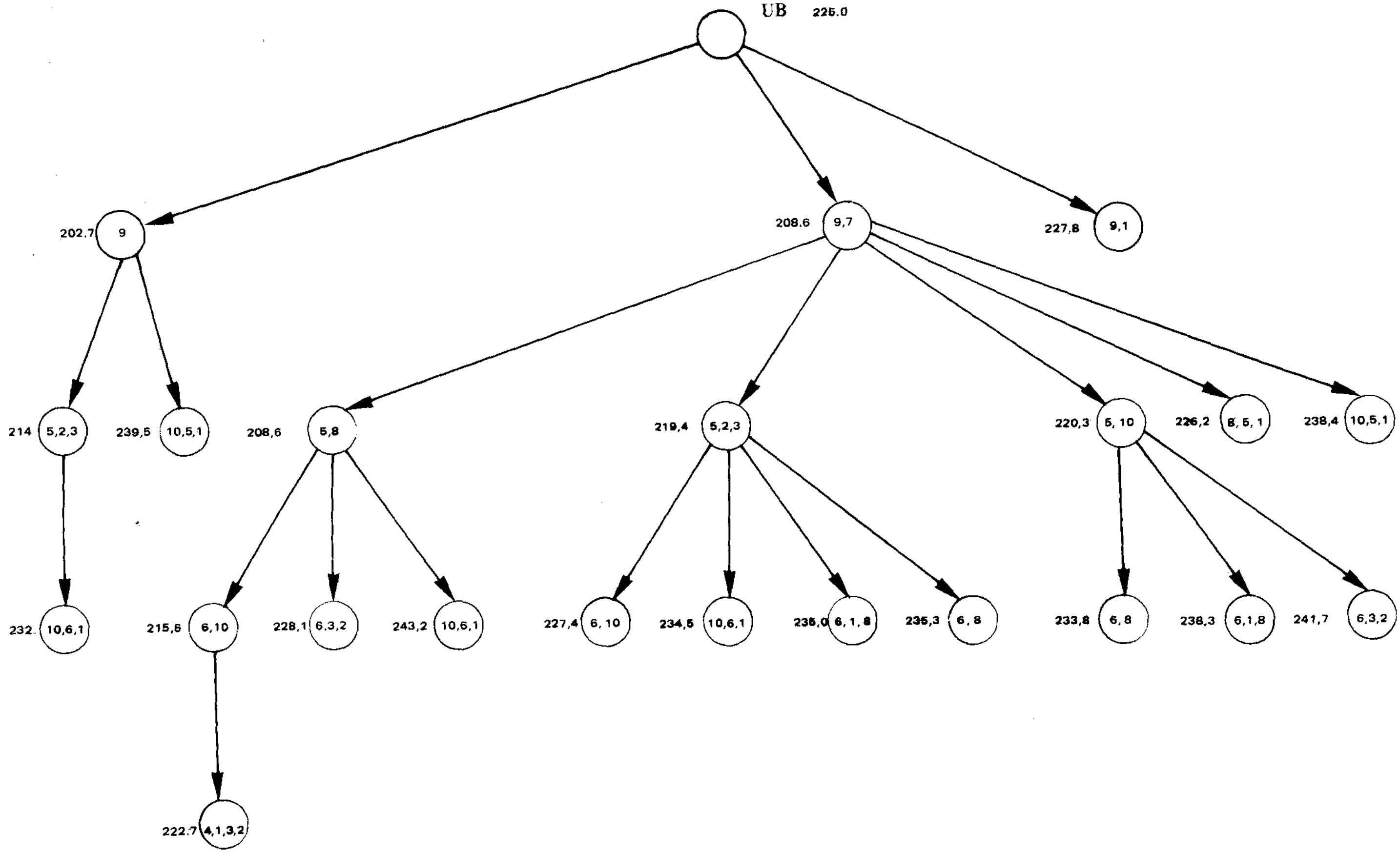

Fig. 8. Branch and bound tree for the example using bound LB0 and branching rule of Section 4.2.

customers. Some of these problems are from the literature; others are derived directly from these. The ten test problems are shown in Table 1.

The depot in problems 6 to 10 in Table 1 is at the same location as the depot in the 50-customer problem in [9]. In all problems no time constraints are considered.

Tables 2 and 3 give the computational performance of the three algorithms. Algorithm 1 is the tree search with branching rule BB1 and bound LB0; algorithm 2 is the same as algorithm 1 with bound LB2; algorithm 3 is the tree search with the branching rule of BB2 and with bound LB0.

Table 2 shows the values of the optimal solution to the problems, together with the values of bounds LB0 and LB2 for the root node. Also shown are the values of the solutions obtained by the use of vehicle routing heuristics, namely the best of the "2-phase" and "tree" heuristics of [6].

Table 3 shows the computing times and total number of nodes in the branch and bound tree for algorithms 1, 2, and 3. Also shown are the computing times required to run both of the abovementioned heuristics. All computing times shown in Table 3 are seconds on the CDC 7600 using the FTN compiler. All codes are in FORTRAN 4.

Although the heuristic used has always obtained an optimal solution for every problem, it should be noted that this is mainly due to the fact that the problems are of small size and relatively free of constraints. It should also be noted from Table 2 that bound LB2 is on average within 2.4% of the optimum solution value and on no occasion is the bound worse than 6.9%. This would suggest that on many practical occasions, a currently available solution to the VRP may be

Table 1
Test problems

Problem	N	M	Q	Details
1	10	4	24	Given as an example
2	12	4	6000	See [9]
3	12	3	6500	Other data as for problem 2
4	21	4	6000	See [9]
5	21	6	4000	Other data as for problem 4
6	15	5	55	Customers are the first 15 of the 50-customer problem in [9]
7	15	3	90	Other data as for problem 6
8	20	6	58	Customers are those numbered 11 to 30 in the 50-customer problem in [9]
9	20	4	85	Other data as for problem 8
10	25	8	48	Customers are those numbered 16 to 30 in the 50-customer problem in [9]

guaranteed (by using the bound) to be close enough to optimal not to warrant the continuation of the search for an improved solution.

Table 2
Computational results: values

Problem	Solution value		Initial lower bounds	
	Optimal	Heuristic	LB0	LB2
1	222.7	222.7	211.0	222.7
2	290.0	290.0	228.6	269.8
3	244.0	244.0	225.1	240.3
4	374.3	374.3	325.4	369.1
5	494.7	494.7	400.0	474.0
6	334.1	334.1	298.1	321.4
7	277.9	277.9	252.1	265.5
8	429.9	429.9	381.2	429.7
9	357.6	357.6	260.0	346.4
10	606.3	606.3	488.9	602.9

Table 3
Computational results: times (seconds on the CDC 7600 using the FTN compiler) and total number of nodes

Problem	Algorithm 1		Algorithm 2		Algorithm 3		Heuristic
	time	nodes	time	nodes	time	nodes	time
1	0.2	49	0.1	1	0.1	22	0.1
2	40.0	3389	28.2	1208	0.9	73	0.1
3	7.2	70	3.3	30	0.7	7	0.1
4	*		28.1	158	*		0.4
5	*		244.0	4026	*		0.5
6	54.0	3336	11.6	194	4.0	227	0.3
7	33.7	2148	60.9	498	26.6	1651	0.2
8	*		8.1	6	75.8	4628	0.4
9	*		139.2	886	*		0.4
10	*		118.6	1718	*		0.6

* Time limit 250 sec.

Acknowledgment

The authors would like to thank SOGESTA, Urbino (Italy) for supporting this work. They would also like to thank two anonymous referees for their suggestions that have led to several improvements and also Dr. John Beasley for undertaking some of the computational tests.

References

[1] M.L. Balinski and R.E. Quandt, "On an integer program for a delivery problem", *Operations Research* 12 (1964) 300.

[2] R. Bellman, "On a routing problem", *Quarterly Journal of Applied Mathematics* (1958) 87.

[3] N. Christofides, *Graph theory, an algorithmic approach* (Academic Press, London, 1975).

[4] N. Christofides, "The vehicle routing problem", *Revue Française d'Automatique, Informatique et Recherche Opérationnelle* 10 (1976) 55.

[5] N. Christofides, "The travelling salesman problem", in: N. Christofides, A. Mingozzi, P. Toth and C. Sandi, eds., *Combinatorial optimization* (John Wiley and Sons, London, 1979).

[6] N. Christofides, A. Mingozzi and P. Toth, "The vehicle routing problem", in: N. Christofides, A. Mingozzi, P. Toth and C. Sandi, eds., *Combinatorial optimization* (John Wiley and Sons, London, 1979).

[7] N. Christofides, A. Mingozzi and P. Toth, "State-space relaxations for combinatorial problems", Imperial College internal report IC, OR, 79, 09, July 1979.

[8] C. Clarke and J.Q. Wright, "Scheduling of vehicles from a central depot to a number of delivery points", *Operations Research* 12 (1964) 568.

[9] S. Eilon, C. Watson-Gandy and N. Christofides, *Distribution management, mathematical modelling and practical analysis* (Griffin, London, 1971).

[10] T.J. Gaskell, "Bases for vehicle fleet scheduling", *Operational Research Quarterly* 18 (1967) 281.

[11] B.E. Gillet and L.R. Miller, "A heuristic algorithm for the vehicle dispatch problem", *Operations Research* 22 (1974) 340.

[12] B.L. Golden, "Vehicle routing problems: formulations and heuristic solution techniques", ORC technical report 113, Massachusetts Institute of Technology (August 1975).

[13] B.L. Golden, "Recent developments in vehicle routing", Presented at the Bicentennial Conference on Mathematical Programming (November 1976).

[14] R. Hays, "The delivery problem", Carnegie Institute of Technology Management Science Research, report 106 (1967).

[15] M. Held and R. Karp, "The TSP and minimum spanning trees I", *Operations Research* 18 (1970) 1138.

[16] M. Held and R. Karp, "The TSP and minimum spanning trees II", *Mathematical Programming* 1 (1971) 6–25.

[17] D. Houck, J.C. Picard, M. Queyranne and R.R. Vemuganti, "The travelling salesman problem and shortest n-paths", University of Maryland (1977).

[18] S. Lin and B.W. Kernighan, "An effective heuristic algorithm for the TSP", *Operations Research* 21 (1973) 498.

[19] C. Miller, A.W. Tucker and R.A. Zemlin, "Integer programming formulation of the travelling salesman problem", *Journal of the Association for Computing Machinery* 7 (1960).

[20] J.F. Pierce, "A two-stage approach to the solution of vehicle dispatching problems", Presented at the 17th T.I.M.S. International Conference London (1970).

[21] F.A. Tillman and H. Cochran, "A heuristic approach for solving the delivery problem", *Journal of Industrial Engineering* 19 (1969) 354.