Journal of the Operational Research Society (2007), 1–11

www.palgrave-journals.com/jors

Solving the capacitated vehicle routing problem using the ALGELECT electrostatic algorithm

J Faulin[1]* and A García del Valle[2]

[1]*Public University of Navarre, Spain; and* [2]*University of La Coruña, Spain*

The capacitated vehicle routing problem (CVRP) with a single depot is a classic routing problem with numerous real-world applications. This paper describes the design, modelling and computational aspects of ALGELECT (electrostatic algorithm), a new algorithm for the CVRP. After some general remarks about the origin of the algorithm and its parameters, a parameter tuning process is carried out in order to improve its efficiency. The algorithm is then explained in detail and its main characteristics are presented. Thus, ALGELECT develops good-quality solutions to the CVRP, in terms of the number of scheduled routes and the load ratio of the delivery vehicles. Finally, ALGELECT is used to find solutions for some Solomon's and Augerat's instances, which are then compared to solutions generated by other well-known methods.

Journal of the Operational Research Society advance online publication, 3 October 2007
doi:10.1057/palgrave.jors.2602502

Keywords: distribution; heuristics; logistics; transport; vehicle routing

Introduction

The ALGELECT electrostatic algorithm was designed to address the need to solve real problems in transportation companies or logistic carriers. First of all, the authors highlight the need for an algorithm to *optimize the number of routes* and the *load ratio of each vehicle*. The necessary characteristics can be outlined as follows:

> *Characteristic I: The solution must involve the smallest possible number of routes*. It can be proved that given a fixed number of nodes, demands included, there exists a minimum number of routes that covers them. Our routes will tend towards this minimum.
>
> *Characteristic II: The load ratio of each vehicle must be maximized*. The objective of this characteristic is to avoid scheduling with practically empty vehicles.

It is easy to see that the above characteristics are highly correlated, that is, once Characteristic I is satisfied it is easier to satisfy Characteristic II and vice versa. Nonetheless, the two characteristics are not equivalent. After a review of the main literature on VRP heuristics, we try to design a practical tool to solve real-world CVRPs. Our purpose is to design a new algorithm that satisfies the aforementioned *Characteristics I and II*.

The problem described in Table 1, and which we are about to solve is named the CVRP-2 problem. Its main characteristics are depicted in Bodin *et al* (1983). The fact that there are several goals involved in CVRP-2 does not imply that it is a multiobjective optimization problem. All goals in our case are derived from the minimization of distances, which is, in fact, the main objective. Our model seeks efficient routes for goods-delivery to or pick-up from a set of nodes. For this reason, the numeric quantities associated to the nodes will always be referred to as demands, whether they are actually demands or supplies. Clearly, CVRP-2 is, by definition, a capacitated vehicle routing problem (CVRP).

Literature review for the ALGELECT algorithm

The CVRP (and the CVRP-2 problem as a particular case) has been described as the most common problem in goods distribution companies in the food, hydrocarbon and retail sectors (Solomon, 1996; Toth and Vigo, 2002, Chapter 10). Moreover, there are several references describing different ways of studying the CVRP. The main source of references for the CVRP is the first part of Toth and Vigo's (2002) book, which offers a detailed analysis of the main types of algorithms to solve the CVRP: (a) branch-and-bound algorithms (Fischetti *et al*, 1994), (b) branch and cut algorithms (Augerat *et al*, 1999), (c) set-covering-based algorithms (Agarwal *et al*, 1989), (d) classical heuristics (Laporte *et al*, 2000) and (e) metaheuristics (Van Breedam, 2001). Other traditional papers on heuristic algorithms have been written by Gaskell (1967), Golden *et al* (1977) and Bodin and Berman (1979). This list of papers can be completed with Laporte and Nobert's (1987), which presents exact algorithms for routing problems.

Thus, the most updated reference is Toth and Vigo's (2002) book, which is a milestone in the VRP bibliography. This

*Correspondence: J Faulin, Department of Statistics and Operations Research, Public University of Navarre, Los Magnolios Building, 1st floor, Campus Arrosadia, Pamplona 31006, Spain.
E-mail: javier.faulin@unavarra.es

Table 1 Characteristics of CVRP-2

Characteristics of VRP	*Options for the CVRP-2*
Size of fleet	Multiple vehicles with capacity constraints
Type of fleet	Homogeneous or heterogeneous
Origin of vehicles	Single depot
Type of demand	Known deterministic demand
Location of demand	At each node
Type of network	Non-oriented. Distance-symmetrical
Maximum route time	No constraint
Activities	Deliveries only
Objectives	Minimize distances Minimize the number of vehicles Maximize vehicle load ratio

book is an excellent reference to find any specific paper about routing problems. Other good reviews about the VRP are the following: Golden and Assad (1991) and Freling *et al* (2001). The main aspects about the logistic implications of a good route scheduling are depicted in Solomon (1996).

Sometimes, knowledge of algorithms for solving TSPs (travelling salesman problems) is useful in order to improve the performance of the VRP methods. In fact, the electrostatic algorithm, which we are going to build, incorporates a TSP subroutine with the purpose of improving the initial method. Likewise, it is convenient to use the following references about the TSP: Lawler *et al* (1985), Raff and Punnen (1999) and Gutin and Punnen (2002). It will be easier to describe our electrostatic method having in mind the previous references.

Other algorithms that need to be taken into account in the construction of our method are the GRASP procedures (Feo and Resende, 1989, 1995), which are iterative randomized sampling techniques that provide a solution to the problem with each iteration. Each GRASP iteration usually can be split into two phases: the first develops an initial solution via an adaptive randomized greedy function, while the second designs a local search procedure for the current solution in the hope of achieving an improvement.

The use of metaheuristics in VRP became popular during the nineties. Two of the most important papers on the use of heuristics and metaheuristics are Gendreau *et al*'s (1994), which introduced the TabuRoute algorithm, and Laporte *et al* (2000), which includes a thorough discussion of classical and modern heuristics. Some ideas for the ALGELECT construction were taken from the metaheuristics described in the above references.

Some ideas for the design of a new procedure

Some experiences in vehicle scheduling support original ideas to build a new heuristic algorithm. Thus, we need to pursue the following objectives in order to construct an effective procedure for solving the CVRP-2:

(a) Knowing the group of demand nodes in the set of routes to be optimized, we will assess the quality of the new algorithm by the way in which it selects the nodes to build a specific route. This grouping criterion is essential for the accurate definition of the TSP. Initially, rather than designing a specific TSP algorithm, we will use one that has been widely tested for its efficiency.
(b) We will try to achieve a balance between the vehicle load ratios and the associated costs. The vehicle load ratios, calculated in percentages, will be taken into account in the development of the computer code to solve the CVRP-2.
(c) An attempt will be made to minimize the number of vehicles involved in the route schedule, in order to adjust the fleet size to the demand nodes. The achievement of this goal must be balanced against the fulfilment of objective (b).
(d) This algorithm must be characterized as an applied method with the intuitive perspective of the most common routing scenarios.
(e) The new algorithm should have a large enough number of parameters to allow model variability for easy adaptation to real cases.

All the above goals profile the *algorithm* that we call *electrostatic*, in accordance with its historical origin. In the next section, we present a detailed description of the philosophy of the ALGELECT procedure. This will be followed with a discussion of the particular values of the algorithm parameters.

Genesis and construction of the ALGELECT electrostatic algorithm

Considering the goals presented above and the mathematical laws in various models from the fields of Physics, Chemistry, Engineering and Economics, a pattern was finally found, the design guidelines of which can be fitted to the aforementioned goals (a)–(e). We were seeking to develop a model based on an existing pattern, already well studied in another scientific field, that might serve as a catalyst in the genesis of the new VRP algorithm. Using these ideas as a source of inspiration, we will formulate a method that will be called the ALGELECT electrostatic algorithm.

This algorithm is based on the imitation of the behaviour of two electrical standing charges. Electrostatics establishes that the forces of attraction or repulsion between two electrical standing charges are directly proportional to the values of the interacting charges and inversely proportional to the squared distance between them. Coulomb's law determines that this force will attract or repel depending on the sign of the charges:

$$F = K\frac{Qq}{d^2} \tag{1}$$

where F is the force of attraction (negative sign) or force of repulsion (positive sign); K the proportionality constant; Q the charge value on which we calculate the force of attraction or repulsion; q the charge value that interacts with the above and from which we calculate the force of attraction or repulsion; d the distance between two charges Q and q.

The interaction between two charges of the same sign will always be repulsive. Conversely, when the signs are different, the interaction will be attractive. We will use this electrostatic formula as a touchstone in the selection of nodes for the delivery vehicle supply routes. It should be noted that the CVRP-2 is characterized by a single depot. Nodes with the strongest electrostatic force of attraction (ie the greatest in *absolute value*) will be added to the route in the process of construction, provided there has been no violation of the capacity constraint of the current vehicle. Nevertheless, a route may conclude with an incomplete load if the load ratio exceeds a certain limit. Once a route has been built, we take the initial node to begin the configuration of the next. It is very important that these nodes should be named. We use the term *key node* to refer to the last node to be added to the current route. The *key node* is the benchmark node in the formation of a particular route. If the vehicles differ in capacity, they will be assigned to routes according to their size. Later, it will be possible to exchange nodes between routes in order to improve the total involved distance. This description of the ALGELECT algorithm appears analogous to the gravity models employed to build routes, similar to Gillett and Miller's (1974) sweep algorithm or to find optimum locations for public facilities (Watson-Gandy, 1972). Nevertheless, the use and definition of the *key node* concept implies that ALGELECT is a more complete and precise algorithm.

Solving the CVRP with the ALGELECT algorithm

The ALGELECT algorithm generates the individual VRP routes one by one, using one of the following criteria to add new nodes to the current route, once a *key node* has been selected: (a) either the *key node* exerts the maximum attraction on the new node or (b) the new node exerts the maximum attraction on the *key node*. This idea will be carefully explained later in the detailed description of the electrostatic algorithm. If the list of *candidate nodes* (nodes that do not exceed the residual capacity of the specific vehicle and have not yet been visited) to be added to the current route is void, that is, the remaining nodes are not feasible for that route, the algorithm generates a new route. If all nodes have been visited once, the algorithm stops and the procedure concludes. Therefore, a node is feasible for a specific route if and only if:

(i) It does not exceed the constraint placed on the maximum physical load of the vehicle (taking into account the current load).
(ii) Neither the current nor the previous routes have included that node.

We conceive the demands as negative electrostatic charges, where the charge value of the node j can be written as follows:

$$q_j = -demand_j \tag{2}$$

If the solution is not feasible we will assume a zero force of attraction and the node in question will be ignored. Therefore, the distance between nodes is the variable d in Equation (1) and the charge Q will take a value of *one* for all the nodes. K has the same effect on all the nodes so K will also be one, since it has no discriminatory effect.

Analogously with Equation (1), we will use the following formula to calculate the attraction or repulsion between node i and the current node j:

$$F_{ij} = K\frac{Qq_i}{d_{ij}^{\Phi_{1j}}} \tag{3}$$

where:

$$\Phi_{1j} = \Delta\left[round\left(\frac{1}{1-d_{1j}/[(1+\alpha)L]} - \chi\right)\right]^{\rho} \tag{4}$$

the parameters of which are depicted as follows:

round ()	Function that rounds to the nearest integer
L	Distance between the depot and the most distant node
d_{1j}	Distance from the current node j to the depot (the depot is the node indexed by 1)
d_{ij}	Distance between the current node j and the node i
Δ	Weight parameter for the distance between the current node j and the node i ($\Delta=2$ in Coulomb's law). It is also known as the *scale parameter*
χ	*Proximity parameter*
ρ	*Weight parameter* for the distance from the depot to the current node j
α	*Non-singularity parameter*

Clearly, $L, d_{1j} \geqslant 0$ since they represent distances. The n*on-singularity parameter* α is a number strictly greater than 0 and smaller than 1, which is introduced to avoid infinite non-real values in (4), when j is the most distant node (which implies that the distance d_{1j} coincides with L). The *proximity parameter* χ is also a number strictly greater than 0 and smaller than 1. Thus, the base of the exponent ρ is always positive, and formula (4) always makes sense. This assertion can be easily proved taking into account the inequalities $0 < \alpha < 1$, $L \geqslant d_{1j}$, and $0 < \chi < 1$.

The *scale parameter* Δ is generated using a uniform distribution in the interval (4, 12). Consequently, its mean is 8. The fact that Δ is simulated by means of a uniform distribution allows different solutions to be obtained when the algorithm is replicated several times. Using experimentation, the uniform distribution in (4, 12) yields positive benefits for the algorithm. Nonetheless, other suitable intervals, such as the interval (1, 10), were also possible.

The decision function F_{ij} will be minimized when selecting the nodes, thus benefiting those with higher demand and closer proximity to the current *key node*, since the feasible solutions have negative associated charges. An exception will be made with very low demand or very distant nodes, so that they do

Table 2 Values of the exponent Φ_{1j} according to the parameter χ and the distance quotient d_{1j}/L (values for the remaining parameters $\rho = 1.5$; $\Delta := 8$; $\alpha = 0.1$)

	d_{1j}/L									
χ	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
0.00	8.00	8.00	8.00	22.63	22.63	22.63	41.57	64.00	117.58	291.86
0.10	8.00	8.00	8.00	8.00	22.63	22.63	41.57	64.00	89.44	291.86
0.20	8.00	8.00	8.00	8.00	22.63	22.63	41.57	41.57	89.44	291.86
0.25	8.00	8.00	8.00	8.00	22.63	22.63	41.57	41.57	89.44	291.86
0.30	8.00	8.00	8.00	8.00	22.63	22.63	22.63	41.57	89.44	291.86
0.40	8.00	8.00	8.00	8.00	8.00	22.63	22.63	41.57	89.44	291.86
0.50	8.00	8.00	8.00	8.00	8.00	22.63	22.63	41.57	89.44	291.86
0.60	8.00	8.00	8.00	8.00	8.00	22.63	22.63	41.57	89.44	252.98
0.70	0.00	8.00	8.00	8.00	8.00	22.63	22.63	41.57	89.44	252.98
0.75	0.00	0.00	8.00	8.00	8.00	22.63	22.63	41.57	89.44	252.98
0.90	0.00	0.00	0.00	8.00	8.00	8.00	22.63	41.57	89.44	252.98
1.00	0.00	0.00	0.00	8.00	8.00	8.00	22.63	41.57	89.44	252.98

not appear isolated and give rise to less efficient routes. These ideas will be developed in the next section, where a detailed description of the algorithm is given.

Solving the TSPs generated by the electrostatic algorithm

The procedure described above gives us a group of routes selected by the electrostatic algorithm. They are in fact sets of nodes that can be improved according to the optimization criteria shown in Table 1. Therefore, we need to solve the associated TSP, taking into account the distance minimization criterion for each set or route. The electrostatic algorithm is not used to solve the TSPs. Instead, we implement Lin and Kernighan's (1973) algorithm, a well-known TSP solving method, in the selected groups of nodes generated by the electrostatic algorithm. Lin and Kernighan's algorithm was compared with other procedures presented in Lawler *et al* (1985) and Gutin and Punnen (2002). It maintains a balance between simplicity and effectiveness that makes it ideal for the solution of these TSPs.

Comments on the constructive process of the ALGELECT algorithm

The selection process of the nodes to be incorporated into the distribution routes is based on the discriminatory formula (3). It should be noted that making K and Q equal to 1 does not affect the selection of the remaining nodes. Only the charges associated with nodes still outside any route have a direct influence on the decision function F_{ij}. It is beyond doubt that the distance between nodes d_{ij} plays an important role in the formulation of the decision function. The minimization of the decision function therefore involves a trade-off in the node selection criteria. We are interested, on the one hand, in the nodes with the highest charge (ie the greatest demand) and on the other, in those nearest to any *key node*. The explanation of the exponent Φ_{1j} requires meticulous study. Let us recall that its form is depicted in formula (4).

Thus, the greater the distance weight exponent Φ_{1j}, the greater the apparent impact of distance on the node to route relationship. Clearly, distance should play a key role in the discovery of new nodes to be incorporated into the routes. Let us observe that the denominator of (3) is the mathematical expression that contains all the algorithmic complexity. During the development of the algorithm, the distance weight exponent Φ_{1j} has gone through many changes, all of which have been tested in practice to ensure that the current expression is appropriate for the convergence and speed of the electrostatic algorithm. Alternative formulations of the exponent that might also have been useful, and even more precise, were discarded because of their complexity. Since the role of the parameters in the distance weight exponent needs to be discussed, we should state the following:

(a) *Parameter* χ: It is a proximity, origin or reference parameter that enables us to establish various benchmarks, depending on the value of the quotient d_{1j}/L. It should be noted that an increase in parameter χ, causes a decrease of the distance weight exponent Φ_{1j}. Table 2 gives the values of the weight exponent Φ_{1j} in relation to parameter χ and the distance quotient d_{1j}/L.
(b) *Parameter* ρ: This parameter controls the remaining parameters in the exponent Φ_{1j}. Its main role is to check any increase in the weight exponent that might cause extreme bias in the decision function. Clearly, this parameter must be positive, that is $\rho > 0$.
(c) *Scale parameter* Δ: This represents the conventional interpretation of the exponent in Coulomb's law (eg: $\Phi_{1j} = 2$, if $\chi = 0$, $\rho = 1$, $\Delta = 2$, $\alpha = +\infty$). A direct increase in this parameter implies an increase in the importance of the distance between nodes as a decision rule. Let us note that a low *versus* high value for Δ makes the algorithm behave in a very different manner: the electrostatic algorithm would be a greedy algorithm for $\Phi_{1j} = 1$.

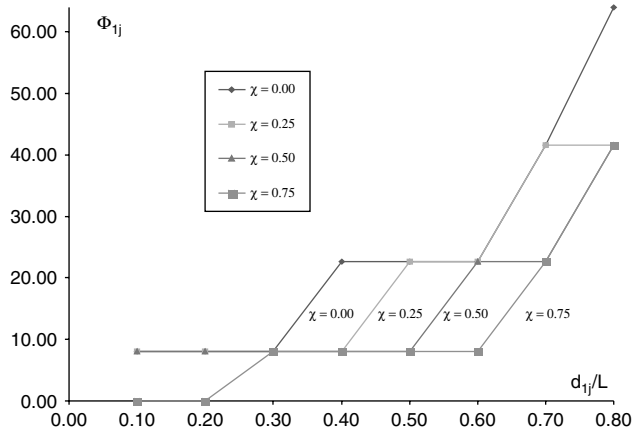

Figure 1 Variation of the weight exponent Φ_{1j} *versus* the parameter χ and the quotient d_{1j}/L.

Table 3 Recommended values for the parameters of the ALGELECT algorithm

χ	ρ	Δ	α
0.5	1.5	8	0.1

(d) *Parameter* α: This parameter is used to avoid singularities in the exponent Φ_{1j}, therefore its range of values is $0 < \alpha < 1$. In fact, it is enough with the constraint $\alpha > 0$.

Figure 1 shows the variation of the coefficient Φ_{1j} *versus* the parameter χ and the distance quotient d_{1j}/L, where the remaining parameter values are $\rho = 1.5$; $\Delta = 8$; $\alpha = 0.1$. We tested the values of the exponent Φ_{1j}, changing the following parameters as follows: χ=0.00, 0.25, 0.50 and 0.75 and d_{1j}/L ranges between 0.1 and 10. From the charts of some graphs, it is easy to verify that different graphs have certain points in common and are sometimes superimposed. Table 2 shows the numerical values of the graphs.

Hereafter, some default values can be assigned to the above-mentioned parameters. These values are the experimentation outcomes for a cluster of values with the defined parameters for the electrostatic algorithm. They are presented in Table 3. The code of the electrostatic algorithm is written in MATLAB.

Some mathematical aspects of the performance of the function F_{ij} and the exponent Φ_{1j}

The purpose of this section of the paper is to offer some explanation of the mathematical properties of the function F_{ij}. The proofs of these properties are straightforward, therefore omitted. We will offer some comments about the properties, instead.

Property 1. The function F_{ij} is not symmetric in its subindices, ie $F_{ij} \neq F_{ji}$.

This property is easy to verify. It is enough to consider the definition of the function:

$$F_{ij} = \frac{q_i}{d_{ij}^{\Phi_{1j}}}$$

The non-depicted parameters take the value 1 (if the force is attractive) and have no influence in the model. Node j is the *key node* and node i is the candidate node to be added to the current route. The lack of symmetry is due to: (i) charge q_i and (ii) weight exponent Φ_{1j}. When we interchange the indices i and j, the q_i and Φ_{1j} values do not change their roles in the formula. The consideration of Property 1 allows for a better understanding of the ALGELECT algorithm.

Property 2. The function F_{ij} is strictly increasing in parameter q_i and strictly decreasing in parameters d_{ij} and Φ_{1j}.

Property 3. The weight exponent Φ_{1j} is strictly increasing in parameters Δ and ρ; increasing in quotient d_{1j}/L and decreasing in parameters χ and α.

The nature of the increase properties for the function F_{ij} and the exponent Φ_{1j} allows for better parameter control in the electrostatic algorithm.

Property 4a. The relationship between the weight exponent Φ_{1j} and the couple $(\chi, d_{1j}/L)$ is an increasing step function, strongly dependent on the initial value of the parameter Δ (seed of the algorithm) and the parameter α. The step values of this function take the following mathematical form:

$$\{0, \Delta, 2^{3/2}\Delta, 3^{3/2}\Delta, 4^{3/2}\Delta, 5^{3/2}\Delta, 6^{3/2}\Delta, \ldots, n^{3/2}\Delta, \ldots\}$$
$$\text{with } n = 1, 2, 3, 4, 5, 6, \ldots$$

where Δ is the seed given by the uniform distribution of the scale parameter. The values for the remaining parameters were $\rho = 1.5$; $\alpha = 0.1$.

Property 4b. The set of values taken for the step function $\Phi_{1j}(\chi, d_{1j}/L)$ with parameters α, Δ, ρ belongs to the following set:

$$\{0, \Delta, 2^{\rho}\Delta, 3^{\rho}\Delta, 4^{\rho}\Delta, 5^{\rho}\Delta, 6^{\rho}\Delta, \ldots, n^{\rho}\Delta, \ldots\}$$
$$\text{with } n = 1, 2, 3, 4, 5, 6, \ldots$$

The description of the values that can be taken by the weight exponent Φ_{1j} allows the discernment of the appropriate behaviour of the electrostatic algorithm. The use of a '*round*' function to define the exponent enables us to simplify its mathematical structure while preserving its most valuable qualities. In this case, the values of the distance weight exponent for the selection of a specific node form a particular sequence, the general term of which is written above. Whether there appear many terms in this sequence or only a few depends on the value of the non-singularity parameter α. The

Table 4 Values of the normalized exponent Φ_{1j}/Δ *versus* the parameter χ and the distance quotient d_{1j}/L (values for the remaining parameters $\rho = 1.5$; $\alpha = 0.05$)

	d_{1j}/L									
χ	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
0.00	1.00	1.00	1.00	2.83	2.83	2.83	5.20	8.00	18.52	96.23
0.10	1.00	1.00	1.00	2.83	2.83	2.83	5.20	8.00	18.52	96.23
0.20	1.00	1.00	1.00	1.00	2.83	2.83	5.20	8.00	18.52	96.23
0.25	1.00	1.00	1.00	1.00	2.83	2.83	5.20	8.00	18.52	96.23
0.30	1.00	1.00	1.00	1.00	2.83	2.83	5.20	8.00	18.52	96.23
0.40	1.00	1.00	1.00	1.00	2.83	2.83	5.20	8.00	18.52	96.23
0.50	1.00	1.00	1.00	1.00	1.00	2.83	5.20	8.00	18.52	96.23
0.60	1.00	1.00	1.00	1.00	1.00	2.83	2.83	8.00	14.70	89.44
0.70	0.00	1.00	1.00	1.00	1.00	2.83	2.83	8.00	14.70	89.44
0.75	0.00	0.00	1.00	1.00	1.00	2.83	2.83	5.20	14.70	89.44
0.90	0.00	0.00	1.00	1.00	1.00	2.83	2.83	5.20	14.70	89.44
1.00	0.00	0.00	1.00	1.00	1.00	1.00	2.83	5.20	14.70	89.44

Table 5 Values of the normalized exponent Φ_{1j}/Δ *versus* the parameter χ and the distance quotient d_{1j}/L (values for the remaining parameters $\rho = 1.5$; $\alpha = 0.1$)

	d_{1j}/L									
χ	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
0.00	1.00	1.00	1.00	2.83	2.83	2.83	5.20	8.00	14.70	36.48
0.10	1.00	1.00	1.00	1.00	2.83	2.83	5.20	8.00	11.18	36.48
0.20	1.00	1.00	1.00	1.00	2.83	2.83	5.20	5.20	11.18	36.48
0.25	1.00	1.00	1.00	1.00	2.83	2.83	5.20	5.20	11.18	36.48
0.30	1.00	1.00	1.00	1.00	2.83	2.83	2.83	5.20	11.18	36.48
0.40	1.00	1.00	1.00	1.00	1.00	2.83	2.83	5.20	11.18	36.48
0.50	1.00	1.00	1.00	1.00	1.00	2.83	2.83	5.20	11.18	36.48
0.60	1.00	1.00	1.00	1.00	1.00	2.83	2.83	5.20	11.18	31.62
0.70	0.00	1.00	1.00	1.00	1.00	2.83	2.83	5.20	11.18	31.62
0.75	0.00	0.00	1.00	1.00	1.00	1.00	2.83	5.20	11.18	31.62
0.90	0.00	0.00	1.00	1.00	1.00	1.00	2.83	5.20	11.18	31.62
1.00	0.00	0.00	0.00	1.00	1.00	1.00	2.83	5.20	11.18	31.62

higher the value of α, the fewer the number of terms that appear in the associated table of exponents.

Property 5. The normalized weight exponent $\Phi_{1j}(\chi, d_{1j}/L)/\Delta$ (for $\rho = 1.5$; $\alpha = 0.05$) can be depicted in Table 4 (along with Table 2). The set of values taken by Φ_{1j}/Δ belongs to the set:

$$\{0, 1, 2^\rho, 3^\rho, 4^\rho, 5^\rho, 6^\rho, \ldots, n^\rho, \ldots\}$$
$$= \{0, 1, 2.83, 5.20, 8, 11.18, 14, 70, 18.52,$$
$$\ldots, 89.44, 96.23, \ldots\}$$

as can easily be verified from Table 4.

Tables 4 and 5 show the influence of parameters ρ and α in the exponent Φ_{1j}. The relationship between Φ_{1j} and ρ is obvious: tiny variations in ρ cause enormous changes in the exponent. The liaison between the exponent and α, however, is subtler. According to Property 3, there exists an inverse relationship between the exponent and the non-singularity parameter. If α takes any value in the interval (0, 0.1) the complexity in the variety of exponent values increases (it is enough to compare Table 2 with Tables 4–5). Best practice in this situation recommends that the value of the non-singularity parameter be at least 0.1, or even greater. The comparison of Tables 4 and 5 clearly shows the increase in complexity when parameter α decreases. These considerations suggest the best parameter choice to be that shown in Table 3, according to the stability of the exponent Φ_{1j}.

Detailed description of the electrostatic algorithm

The flow chart of the electrostatic algorithm is illustrated in Figure 2. We have made use of the following notation:

CMAX	maximum load of the vehicle for which the route is being scheduled
C	accumulated real load (assigned load to the vehicle that actually covers the route)
$c(i)$	demand of node i

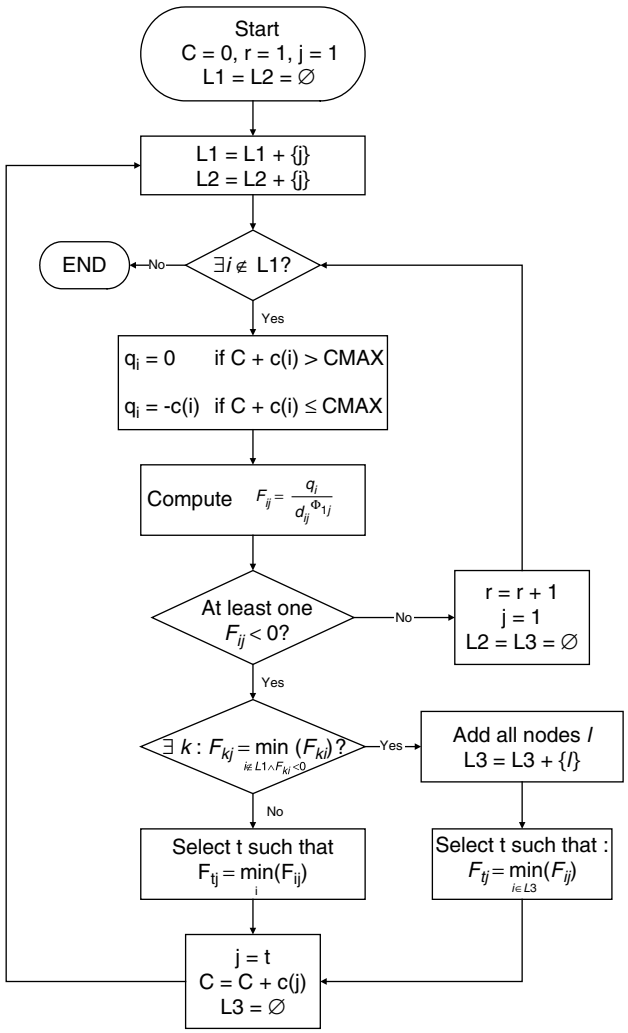

Figure 2 Flow chart for the ALGELECT electrostatic algorithm.

d_{ij} distance between nodes i and j. The depot is the node 1
r index of the route
j index of the *key node*
i index of nodes
$L1$ list of nodes already assigned to routes
$L2$ list of nodes of the route being planned
$L3$ list of nodes most strongly attracted by the *key node*, that is

$$L3 = \left\{ k \notin L1 / F_{kj} = \min_{i \notin L1 \wedge F_{ki<0}} (F_{ki}) \right\}$$

Thus, if a node k belongs to list $L3$ then the attraction from node k to the *key node j* is greater than the attraction from the node k to the remaining nodes in the list L1.

Initially, the *key node* is the depot for every route generated. Furthermore, each route begins and ends at the depot. The algorithm finishes when list L1 contains all the nodes (delivery points in the broad sense of the word).

The ALGELECT algorithm can be broken down into the following steps:

1. Begin the process making $C=0$, $r=1$, $j=1$ and emptying lists L1, L2 and L3.
2. Add the *key node j* (which is initially the depot) to lists L1 and L2.
3. If no node is free (ie not included in list L1) then the algorithm finishes. Otherwise, go to Step 4.
4. *Assignment of charges to nodes*. If the accumulated real load plus its demand exceeds the capacity of the vehicle, then this node is discarded, because its charge will have the same sign as that of the *key node* (always with the value $+1$). Therefore, the two nodes will repel each other. Otherwise, the charge of the current node is its demand with negative sign.
5. Calculate the forces F_{ij} between each couple of nodes not in list L1 while including the *key node*.
6. Verify if there are nodes that attract the *key node*. If none satisfies this property, then a new route has been obtained (and also the associated vehicle). The assignment $r=r+1$ is made (in order to find the next route), $j=1$ (the depot becomes the *key node*) and lists L2 and L3 are emptied; go to Step 3. Otherwise, go to the following step.
7. It is necessary to verify if there is any node that is subject to the maximum force of attraction from the *key node* (ie the key node attracts it with maximum force). If there is at least one node in list L3 (nodes that are subject to the maximum force of attraction from the *key node*), we must choose the node t in list L3 having the weakest force of attraction from node i to the *key node j* (maximum value in F_{ij}, because the forces of attraction are negative) and proceed to Step 8. If there is no such node (ie the list L3 is empty), one must choose the node t that is subject to the strongest force of attraction from the *key node* (minimum value for F_{ij}). Proceed to the next step.
8. Change the *key node* (which becomes t) making $j=t$, update the accumulated real load: $C=C+c(j)$ and empty list L3. Go to Step 2.

It is easy to understand that the algorithm adds points that are strongly attracted to one another, in order to minimize distances and achieve maximum used capacity from the vehicle that covers the scheduled route. It is important to highlight Step 7 because of its subtlety. At this stage, nodes non-assigned to previous routes and subject to the maximum force of attraction from the *key node* are checked. This prevents any nodes only weakly attracting the *key* node from being left unassigned to any route. This anomalous situation usually produces poor quality routes with isolated nodes. These poor quality routes imply long distance itineraries and low vehicle occupation levels.

Step 7 was the upshot of experimentation using plenty of instances and cases. The results obtained with the electrostatic algorithm were much better when Step 7 was implemented

than when it was omitted. The new variant of the electrostatic algorithm builds suitable routes for practical cases and essentially improves the solutions of the CVRP-2. The final routes involved shorter distances and therefore lower costs.

Computational results

We tested the electrostatic algorithm with some well-known instances, such as the *Extended Solomon Instances* (downloaded from http://branchandcut.org/VRP/data/, accessed 23 April 2007), the Augerat *et al* (1995) *Set A Instances* (downloaded from http://branchandcut.org/VRP/data/#A, accessed 27 January 2007), and the classical sets of *Solomon Instances* (downloaded from http://www.idsia.ch/luca/macs-vrptw/problems/welcome.htm, accessed 23 April 2007). We implemented the ALGELECT algorithm in some of the above-mentioned instances (Instance Sets #1, 2 and 3) and obtained the results given in Tables 6 and 7. We ignored the time windows associated to those instances in Tables 6 and 7.

We solved some of the *Extended Solomon Instances* (Set #1 Instances). We depicted only a few selected instances in order to summarize the problems that have been solved with ALGELECT. Each case takes the names R1_x_y, R2_x_y, C1_x_y, C2_x_y, or RC1_x_y and RC1_x_y, where 'y' is the number assigned to the problem and '100x' the number of nodes. Thus, for example, the case C1_8_3 corresponds to problem number 3 for a VRP with 800 nodes. As can be observed, a good level of occupation was achieved in Tables 6 and 7. Cases with up to 1000 nodes were solved: all the results are highly consistent in terms of the vehicle load level and minimal distance.

If we pay attention to the MATLAB report for the particular case c1_2_1, it is evident that the majority of the CPU time (around 370 s) is devoted to the resolution of the TSPs. Indeed, the first two functions used in the resolution of the TSPs take

Table 6 Miscellaneous cases solved by the ALGELECT electrostatic algorithm taken from the *Extended Solomon Instances* available at http://branchandcut.org/VRP/data/ ignoring the time windows, and compared to the traditional Clarke and Wright's (1964) algorithm. We used a PC Pentium 4 CPU 3.00 GHz with 960 MB of RAM

Set # 1 instances		*Extended Solomon instances*					
		ALGELECT Algorithm			*Clarke and Wright's Algorithm*		
Type	*Case*	*Distance*	*Number of vehicles*	*Vehicle load level*	*Distance*	*Number of vehicles*	*ALGELECT difference (%)*
R	**1_2_1**	3312	18	0.9758	3629	18	9.59
R	**1_4_1**	8118	36	0.9874	8471	36	4.35
R	**1_6_1**	17 244	54	0.9942	17 908	54	3.85
R	**1_8_1**	30 323	72	0.9882	31 190	72	2.86
R	**1_10_1**	45 830	92	0.9847	48 066	91	4.88
C	**1_2_1**	2798	18	0.9806	2819	18	0.73
C	**1_4_1**	7001	36	0.9986	7249	36	3.55
C	**1_6_1**	13 925	56	0.9839	14 299	55	2.68
C	**1_8_1**	24 606	72	0.9986	24 693	72	0.36
C	**1_10_1**	40 668	90	0.9967	41 073	90	1.00
RC	**1_2_1**	3287	19	0.9363	3412	18	3.81
RC	**1_4_1**	7970	37	0.9631	8538	36	7.12
RC	**1_6_1**	16 350	55	0.9891	16 937	55	3.59
RC	**1_8_1**	29 352	73	0.9854	31 040	73	5.75
RC	**1_10_1**	45 613	90	0.9901	46 818	90	2.64
R	**2_2_1**	1979	5	0.7026	1965	4	−0.71
R	**2_4_1**	4362	8	0.8886	4369	8	0.16
R	**2_6_1**	8634	13	0.8259	9543	16	10.53
R	**2_8_1**	13 662	15	0.9487	17 417	27	27.49
R	**2_10_1**	20 223	20	0.9059	28 116	39	39.03
C	**2_2_1**	1644	6	0.8976	1927	6	17.20
C	**2_4_1**	3831	11	0.9818	4067	11	6.16
C	**2_6_1**	7408	17	0.9697	8106	18	9.43
C	**2_8_1**	11 174	22	0.9877	13187	27	18.02
C	**2_10_1**	16 356	28	0.9770	20 959	38	28.15
RC	**2_2_1**	2015	6	0.5930	1833	4	−9.01
RC	**2_4_1**	3772	8	0.8909	4288	10	13.69
RC	**2_6_1**	7403	11	0.9891	8471	16	14.42
RC	**2_8_1**	12 242	15	0.9591	15 677	26	28.06
RC	**2_10_1**	17 784	18	0.9901	27 092	39	52.34

Table 7 Computational results of the implementation of the ALGELECT algorithm in 27 test cases from the Augerat *et al* (1995) *Set A* http://branchandcut.org/VRP/data/#A. The same webpage provides the best known solutions. We used a PC Pentium 4 CPU 3.00 GHz with 960 MB of RAM

Set #2 instances	*Augerat (1995) Set A*							
	ALGELECT solution			*Best known solution*				
Case	*Distance*	*Number of vehicles*	*Vehicle load level*	*Distance*	*Number of vehicles*	*Vehicle load level*	*Difference*	*Percentage difference (%)*
A-n32-k5	1032	5	0.8200	784	5	0.82	248	31.58
A-n33-k5	789	5	0.8920	661	5	0.89	128	19.32
A-n33-k5	834	7	0.7729	742	5	0.90	92	12.34
A-n34-k5	835	5	0.9200	778	5	0.92	57	7.36
A-n36-k5	908	5	0.8840	799	5	0.88	109	13.70
A-n37-k5	783	5	0.8140	669	5	0.81	114	17.10
A-n37-k6	1046	6	0.9500	949	6	0.95	97	10.20
A-n38-k5	861	6	0.8017	730	5	0.96	131	17.92
A-n39-k5	990	5	0.9500	822	5	0.95	168	20.43
A-n39-k6	861	6	0.8767	831	6	0.88	30	3.56
A-n44-k7	1028	6	0.9500	937	7	0.95	91	9.74
A-n45-k6	1040	7	0.8471	944	6	0.99	96	10.18
A-n45-k7	1258	7	0.9057	1146	7	0.9	112	9.77
A-n46-k7	1062	7	0.8614	914	7	0.86	148	16.24
A-n48-k7	1162	7	0.8943	1073	7	0.89	89	8.27
A-n53-k7	1191	7	0.9486	1010	7	0.95	181	17.89
A-n54-k7	1291	7	0.9557	1167	7	0.96	124	10.59
A-n55-k9	1191	9	0.9322	1073	9	0.93	118	10.98
A-n60-k9	1503	9	0.9211	1354	9	0.92	149	10.99
A-n61-k9	1181	10	0.8850	1034	9	0.98	147	14.21
A-n62-k8	1408	8	0.9163	1288	8	0.92	120	9.35
A-n63-k9	1745	9	0.9700	1616	9	0.97	129	8.00
A-n63-k10	1409	10	0.9320	1314	10	0.93	95	7.24
A-n64-k9	1521	9	0.9422	1401	9	0.94	120	8.58
A-n65-k9	1357	9	0.9744	1174	9	0.97	183	15.55
A-n69-k9	1389	9	0.9389	1159	9	0.94	230	19.81
A-n80-k10	1901	10	0.9420	1763	10	0.94	138	7.80

up 73.3% of this time. This percentage changes with the size of the problem: it diminishes as the dimension of the problem increases. Likewise, the greater the problem, the more CPU time it requires. A Pentium 4 CPU 3.00 GHz PC with 960 MB of RAM was used to perform all the calculations in the paper.

The Set #2 Instances are taken from Augerat *et al* (1995) *Set A*. The structure of the solved problems in this set is as follows: **A-nx-ky** implies that the problem has **x** nodes and **y** vehicles. The structure and best-known solutions for these problems can be found at http://branchandcut.org/VRP/data/#A. The results of the ALGELECT implementation on Set #2 Instances are shown in Table 7.

ALGELECT has been also tested on a classic set of *Solomon Instances* (Set #3 Instances) (Solomon, 1987) composed of six different problem types (C1, C2, R1, R2, RC1, RC2). According to Solomon (1987), each data set contains between eight and 12 100-node problems. The names of the six problem types should be interpreted as follows: Set C has clustered customers and time windows based on a known solution. In Set R customer location is generated uniformly randomly over a square. Set RC has a combination of randomly located and clustered customers. Set C nodes have narrow time windows and low vehicle capacity. Set R nodes have large time windows and high vehicle capacity. Therefore, the solutions to set R problems involve very few routes and significantly more customers per route. The results of the ALGELECT implementation on the classical test of *Solomon Instances* (Set #3 Instances) considering time windows are not good for the current ALGELECT version and will not be shown in this paper.

Comments about the implementation of the ALGELECT algorithm in the Set #1 instances (Table 6)

The behaviour of ALGELECT in the Set#1 Instances (*Extended Solomon Instances*) has been excellent in general terms: it has reached very high occupation levels. If we compare the electrostatic procedure with the traditional Clarke and Wright's (1964) savings algorithm (CWSA), we realize the quality of the solutions found by ALGELECT. We have solved 30 cases and ALGELECT has outperformed the Clarke and Wright's savings algorithm (CWSA) in all cases except two of them (R2_2_1 and RC2_2_1, which are cases of similar characteristics). It is true that the comparison with the

CWSA algorithm is, perhaps, not especially meaningful, but it is an initial benchmark which can be used as a touchstone to measure the algorithms' performance. An algorithm which clearly does not overcome the CWSA procedure cannot be considered as a good routing builder.

Nevertheless, ALGELECT has designed better solutions than those generated by the savings algorithm: on average, the electrostatic solutions outperform in a 10.39% in distance in the Set#1 Instances. Even so, it is necessary to pinpoint other details of the ALGELECT application to this case:

(a) ALGELECT outperforms the CWSA in 28 out of 30 instances.
(b) ALGELECT reaches a good occupation level in vehicles (94.20% on average).
(c) The number of vehicles used by ALGELECT is smaller in 10 cases, equal in 14 cases and greater in six cases. However, the difference in vehicles in favour of CWSA is always 1 vehicle, while that difference in favour of ALGELECT is much greater (up to a difference of 21 vehicles in the worst case). The average difference in vehicles is 2.73 in favour of ALGELECT.
(d) ALGELECT obtains its better results in the R and RC sets and as the number of nodes is greater (see cases R2_10_1 or RC2_10_1).

Therefore, it is clear the electrostatic algorithm behaves better than CWSA in the Set#1 Instances.

Comments about the implementation of the ALGELECT algorithm in the Set #2 instances (Table 7)

Knowing the results of the previous set of instances, we have chosen now the Augerat *et al* (1995) Set A http://branchandcut.org/VRP/data/#A as the Set #2 Instances. The *Augerat Set A* webpage also gives the best-known solution. The main results of the ALGELECT performance in this case are given in Table 7. The percentage difference given in the last column describes the percentage increase of the electrostatic solutions in relation to the best known ones. ALGELECT does not reach the best-known solution in any case. The best ALGELECT solution has been obtained in the A-n39-k6 case with a 3.56% of increase. The average increase of the electrostatic algorithm is 12.91% for the whole Set #2 Instances. On the other hand, the average vehicle occupation level for the ALGELECT solutions is 0.9036. The number of vehicles used by ALGELECT solutions and the best-known solutions are the same in 22 cases, while the best-known solutions employ one or two vehicles less in four cases and ALGELECT uses one vehicle less in one case.

Therefore, the ALGELECT behaviour with the Set #2 Instances, according to the solutions quality, has been appropriate for using those solutions in real cases. If we compare the ALGELECT performance in Instance Sets #1 and #2, we realize it clearly outperforms CWSA and it obtains solutions which are 10% above the best-known solutions for the Augerat *et al* (1995) *Set A*, roughly speaking.

Comments about the implementation of the ALGELECT algorithm in the Set #3 instances

Finally, we implemented the ALGELECT procedure taking into account the time windows of the traditional Solomon Instances, which can be found at http://www.idsia.ch/luca/macs-vrptw/problems/welcome.htm. We have adapted the electrostatic algorithm to the CVRPTW (Capacitated Vehicle Routing Problem with Time Windows) in a direct way, considering the time windows as a new constraint, similar to the capacity constraint. Therefore, in the route-building process, if a route does not fulfill the time windows constraint, its associated node is discarded and a new node is sought. We have compared the solutions generated by the ALGELECT algorithm with the best-known solutions. Nevertheless, ALGELECT performs less well in this set of instances than in the two previous ones. On average, the electrostatic algorithm worsens the best-known solutions by 84.04%, and its average vehicle load level is a mere 0.2853. There is an obvious need for improvement in the ALGELECT design for use with CVRPTW. This is clearly an open question to study in future works.

Conclusions and open questions

After the implementation of the ALGELECT algorithm in some cases taken from the *Extended Solomon Instances*, the Augerat *et al* (1995) *Set A Instances* and the *Solomon Instances* described in Tables 6 and 7, it is possible to assert the following:

(a) ALGELECT builds routes in a very intuitive way, reaching a good load level per vehicle. Hence, the number of routes, and therefore the number of vehicles, associated with this electrostatic procedure, is minimal. In some cases, ALGELECT reaches the optimum value for certain VRPs.
(b) The VRP solution CPU time when using ALGELECT depends on the size, node distribution and instance of the routing problem. Nevertheless, the running time of our algorithm in the shown cases ranges from 5 to 12 min. Most of this time is devoted to the resolution of the TSPs that appear after the ALGELECT grouping procedure.
(c) The good quality solutions obtained by the electrostatic algorithm in solving the CVRP do not depend on the problem size: the vehicle load level is around 90% even in the largest *Salomon's instances* with good outcomes in Augerat *et al* (1995) *Set A instances*.
(d) The direct application of the ALGELECT procedure to solve the CVRPTW does not generate good quality solutions.

In another order of consideration, the CVRP constitutes a broad field of research and study. It is open to continuous improvement. Specifically, there are many variants of the electrostatic algorithm developed in this paper and it can be adapted to many new situations. We are thus prompted to raise the following open questions:

(1) *To establish a time constraint* (*upper bound*) *on each delivery route*: This constraint is of great practical interest, because the drivers of the vehicles have to comply with local traffic legislations limiting their driving time. It is easy to implement this time constraint in the electrostatic algorithm due to special nature of the ALGELECT procedure.
(2) *To improve the time windows version of the ALGELECT algorithm*. We have carried out an initial version of the ALGELECT method with time windows. This version could be improved in future releases of the algorithm.
(3) *To assess the performance and properties of ALGELECT against changes in the parameter values*. We have employed ALGELECT with the values described in Table 3, but it would be interesting to find alternative values for the parameters that might improve on the outcomes obtained in this paper.
(4) Finally, the algorithm could be associated to a GIS in order to implement it in real time with real geographical information.

Acknowledgements— We would like to express our gratitude for the support given by Vicente Inglada and the Spanish Government Ministry of Transport (Ministerio de Fomento) to this research. Similarly, we would like to thank two anonymous referees for their valuable comments, which have significantly improved this paper.

References

Agarwal Y, Mathur K and Salkin HM (1989). A set-partitioning-based exact algorithm for the vehicle routing problem. *Networks* **19**: 731–749.

Augerat P, Belenger JM, Benavent E, Corberán A, Naddef D, Rinaldi G (1995). *Computational results with a branch and cut code for the capacitated vehicle routing problem*. Technical Report RR 949-M, University Joseph Fourier, Grenoble, France.

Augerat P, Belenger JM, Benavent E, Corberán A and Naddef D (1999). Separating capacity constraints in the CVRP using tabu search. *Eur J Opl Res* **106**: 546–557.

Bodin L and Berman L (1979). Routing and scheduling of school buses by computer. *Transport Sci* **13**: 113–129.

Bodin L, Golden B, Assad A and Ball M (1983). Routing and scheduling of vehicles and crews. The state of the art. *Comput Opns Res* **10**: 63–211.

Clarke G and Wright J (1964). Scheduling of vehicles from a central depot to a number of delivering points. *Opns Res* **12**: 568–581.

Feo TA and Resende MGC (1989). A probabilistic heuristic for a computationally difficult set covering problem. *Opns Res Lett* **8**: 67–71.

Feo TA and Resende MGC (1995). Greedy randomized adaptive search procedures. *J Glob Optim* **6**: 109–133.

Fischetti M, Toth P and Vigo D (1994). A branch-and-bound algorithm for the capacitated vehicle routing problem on direct graphs. *Opns Res* **42**: 846–859.

Freling R, Wagelmans APM and Pinto Paixão JM (2001). Models and algorithms for single-depot vehicle scheduling. *Transport Sci* **35**: 165–180.

Gaskell TJ (1967). Bases for the vehicle fleet scheduling. *Opl Res Quart* **18**: 281–295.

Gendreau M, Hertz A and Laporte G (1994). A tabu search heuristic for the vehicle routing problem. *Mngt Sci* **40**: 1276–1290.

Gillett BE and Miller LR (1974). A heuristic algorithm for the vehicle-dispatch problem. *Opns Res* **22**: 340–349.

Golden BL and Assad AA (eds) (1991). *Vehicle Routing: Methods and Studies*. Elsevier Science Publ. North Holland: Amsterdam.

Golden B, Magnanti T and Nguyen H (1977). Implementing vehicle routing algorithms. *Networks* **7**: 113–148.

Gutin G and Punnen AP (eds) (2002). *The Traveling Salesman Problem and Its Variations*. Kluwer Academic Publishers: Boston.

Laporte G and Nobert Y (1987). Exact algorithms for the vehicle routing problem. *Ann Discr Math* **31**: 147–184.

Laporte G, Gendreau M, Potvin JY and Semet F (2000). Classical and modern heuristics for the vehicle routing problem. *Int Trans Opl Res* **7**: 285–300.

Lawler EL, Lenstra JK, Rinnooy Kan AHG and Shmoys DB (eds) (1985). *The Traveling Salesman Problem*. John Wiley & Sons: Chichester, UK.

Lin S and Kernighan BW (1973). An effective heuristic algorithm for the traveling-salesman problem. *Opns Res* **11**: 498–516.

Raff SJ Punnen A (eds) (1999). The traveling salesman problem. *Comput Opns Res* 26: 293–441.

Solomon MM (1987). Algorithms for the vehicle routing and scheduling problem with time window constraints. *Opns Res* **35**: 254–265.

Solomon MM (1996). Logistics. In: Gass SI and Harris CM (eds). *Encyclopaedia of Operations Research and Management Science*. Kluwer Academic Publishers: London, pp 354–357.

Toth P and Vigo D (2002). *The Vehicle Routing Problem, SIAM Monographs on Discrete Mathematics and Applications*. SIAM—Society for Industrial and Applied Mathematics: Philadelphia.

Van Breedam A (2001). Comparing descent heuristics and metaheuristics for the vehicle routing problem. *Comput Opns Res* **28**: 289–315.

Watson-Gandy CDT (1972). A note on the centre of gravity in depot location. *Mngt Sci* **18**: B-478–B-481.

Received April 2003;
accepted June 2007 after three revisions