Journal of the Operational Research Society (2008) 59, 963–976

www.palgrave-journals.com/jors

A tabu search algorithm for the periodic vehicle routing problem with multiple vehicle trips and accessibility restrictions

F Alonso[1], MJ Alvarez[1] and JE Beasley[2]*

[1]*Universidad de Navarra, San Sebastián, Spain; and* [2]*Brunel University, Uxbridge, UK*

In this paper, we consider a periodic vehicle routing problem that includes, in addition to the classical constraints, the possibility of a vehicle doing more than one route per day, as long as the maximum daily operation time for the vehicle is not exceeded. In addition, some constraints relating to accessibility of the vehicles to the customers, in the sense that not every vehicle can visit every customer, must be observed. We refer to the problem we consider here as the site-dependent multi-trip periodic vehicle routing problem. An algorithm based on tabu search is presented for the problem and computational results presented on randomly generated test problems that are made publicly available. Our algorithm is also tested on a number of routing problems from the literature that constitute particular cases of the proposed problem. Specifically we consider the periodic vehicle routing problem; the site-dependent vehicle routing problem; the multi-trip vehicle routing problem; and the classical vehicle routing problem. Computational results for our tabu search algorithm on test problems taken from the literature for all of these problems are presented.

Journal of the Operational Research Society (2008) **59,** 963–976. doi:10.1057/palgrave.jors.2602405
Published online 18 April 2007

Keywords: tabu search; periodic vehicle routing problem; multiple use of vehicles; accessibility restrictions; site-dependencies; multi-trip

1. Introduction

A basic problem that arises in distribution systems is the well-known vehicle routing problem (VRP). Given a set of customers with known geographical locations and demands that must be satisfied by a homogeneous fleet of vehicles based at a central depot, the problem consists of designing a set of routes, one per vehicle, that minimizes the total distance travelled. Each route starts and finishes at the depot and each customer is visited exactly once. The total demand on a route must not exceed the capacity of the vehicle and the route may have limited length or duration.

In some distribution systems, planning is not on a daily basis, rather there is a periodicity in deliveries so that customers must be served from 1 to t times during a period of t days. When a customer has a delivery frequency smaller than the number of days in the period, the deliveries to that customer must be done using one of the feasible patterns of delivery days defined for that customer. This problem is known as the periodic vehicle routing problem (PVRP) and seeks to define the arrangement of the customer's deliveries in a given period at the same time as a set of routes, one per vehicle, for every day of the period are designed minimizing the total distance travelled. The service level required by every customer must be met, the total demand on every route in every day of the period must not exceed the vehicle capacity and the length or duration of the routes may be limited.

In some distribution systems not all vehicles can visit all customers. In other words, there exists a compatibility relationship between the customers and the vehicles. This problem is known in the literature as the site-dependent vehicle routing problem (SDVRP) and involves a fleet of heterogeneous vehicles. In such problems, in addition to the classic constraints for the VRP, customer/vehicle compatibility (accessibility) restrictions must be respected. The SDVRP was first presented by Nag *et al* (1988). They analysed several heuristics and presented computational results for problems involving 50, 75 and 100 customers with a different number of vehicle types and degrees of site dependencies. Chao *et al* (1999) developed a heuristic based on record-to-record improvement, similar to that developed for the PVRP (Chao *et al*, 1995). In both papers, they improved the best known solutions for PVRP and SDVRP test problems presented in the literature and developed new test problems for both problems.

The SDVRP can be viewed as a special case of the PVRP by relating vehicle types in the SDVRP to days of the period in the PVRP. Both problems can be characterized as multilevel routing problems or generalized assignment problems: at the

*Correspondence: JE Beasley, Department of Mathematical Sciences, Brunel University, Uxbridge UB8 3PH, UK.
E-mail: john.beasley@brunel.ac.uk

Table 1 Characteristics of different problems

Problem	*Multiple periods*	*Accessibility restrictions*	*Multiple vehicle trips*
SDMTPVRP	Yes	Yes	Yes
PVRP	Yes	No	No
SDVRP	No	Yes	No
MTVRP	No	No	Yes
VRP	No	No	No

first level, an allowable vehicle type (or pattern of delivery days) is selected for each customer and at the second level, a capacity and time constrained VRP is solved for each type of vehicle (or day of the period). Taking advantage of this characteristic, Cordeau and Laporte (2001) solved the SDVRP applying the same algorithm used to solve the PVRP (Cordeau *et al*, 1997). In these two papers, the best solutions they report on test problems taken from the literature show considerable improvement over the previous best known solutions for the PVRP and the SDVRP. In both cases, the authors present results on newly generated test problems.

The standard VRP definition implicitly assumes that each vehicle is used only once over the planning period. However, once the vehicle routes have been designed, it may be possible to assign several of them to the same vehicle such that the vehicle can feasibly operate them over the planning period. The problem of designing routes with multiple vehicle trips is known in the literature as the vehicle routing problem with multiple use of vehicles (VRPM) due to Taillard *et al* (1996) or the multi-trip vehicle routing problem (MTVRP) due to Brandão and Mercer (1997, 1998).

Taillard *et al* (1996) proposed a heuristic made up of three parts. It first generates a large set of good vehicle routes satisfying the VRP constraints. It then makes a selection of a subset of these routes using an enumerative algorithm. Finally, it assembles the selected routes into feasible working days using several applications of a bin packing heuristic. Brandão and Mercer (1998), drawing on their earlier work (Brandão and Mercer, 1997), presented a tabu search approach which they compared with the approach of Taillard *et al* (1996). Petch and Salhi (2003) presented a multi-phase constructive heuristic for the problem. Their approach involves a number of distinct features including: a savings-based approach to generate vehicle routes; use of bin-packing; local search heuristics to improve vehicle routes and a population approach. They presented a comparison of their approach with that of Taillard *et al* (1996) and Brandão and Mercer (1998).

In this paper, we consider a new VRP that includes the characteristics of the former ones, that is, a PVRP with multiple vehicle use and accessibility restrictions. The problem will be called the site-dependent multi-trip periodic vehicle routing problem (denoted by the abbreviation SDMTPVRP). The main characteristics of the presented problems are summarized in Table 1. We show that a tabu search heuristic developed previously by Cordeau *et al* (1997) for the PVRP can be modified to solve SDMTPVRP and its particular cases with encouraging results. We denote this modified tabu search algorithm **TS-ABB**. Table 1 clearly shows that the SDMTPVRP subsumes a significant number of other VRPs and so our algorithm, TS-ABB, can be applied to these as well. It is also worth mentioning here that Table 1 highlights a number of problems (eg the No/Yes/Yes choice) that have been little considered, or not considered at all, in the literature and again our algorithm can be applied to these problems as well.

In this paper, we first present our problem formulation for SDMTPVRP. An overview of our heuristic is then presented, followed by computational experiments and then conclusions. Note here that as in this paper we are considering all of the five different VRPs shown in Table 1 space limitations dictate that it is not possible to present a comprehensive literature review for these problems. The reader interested in a comprehensive overview of the work that has been done with regard to vehicle routing heuristics is referred to Cordeau *et al* (2002, 2005), Gendreau *et al* (2002), Laporte and Semet (2002) and Tarantilis *et al* (2005).

2. Notation and formulation

The SDMTPVRP is defined on a multigraph $G = (V, A)$, where $V = \{v_0, v_1, \ldots, v_n\}$ is the vertex set and $A = \{(v_i, v_j)^{k,r,l}\}$ is the arc set. Vertex v_0 represents the depot and vertices $V\backslash\{v_0\}$ correspond to the customers. Here, k, r and l represent the vehicle, the route of that vehicle and the delivery day, respectively. With each arc $(v_i, v_j)^{k,r,l}$ a cost is associated depending on the arc length and the vehicle that goes through it. In order to formulate the SDMTPVRP as a 0–1 linear programme, a_{hl} binary constants are defined equal to 1 if and only if the day l belongs to the delivery pattern h (zero otherwise). Also, binary variables x_{ij}^{krl} are defined equal to 1 if vehicle k in its route r on day l visits customer j immediately after customer i (zero otherwise), and y_{ih} equal to 1 if the delivery-day pattern $h \in C_i$ is assigned to customer i (zero otherwise), where C_i is the set of feasible delivery-day patterns for customer i.

Making use of the notation shown in Table 2, the set K_i of every feasible vehicle for customer i is created with every vehicle from the set K whose type belongs to the given set P_i of the feasible vehicle types for customer i. The task of generating these sets is accomplished in an initial step. Since the depot does not have any demand, we assume $q_0 = \mu_0 = \phi_0 = 0$; $K_0 = K$ is also assumed. The SDMTPVRP can be stated in the following way:

$$\text{minimize} \quad \sum_{i=0}^{n}\sum_{j=0}^{n}\sum_{k\in K_i\cap K_j}\sum_{r=1}^{w}\sum_{l=1}^{t} d_{ij}c_k x_{ij}^{krl} \tag{1}$$

$$\text{subject to} \quad \sum_{h\in C_i} y_{ih} = 1 \qquad (i = 1, \ldots, n) \tag{2}$$

Table 2 Notation used for the 0–1 linear program

n	Total number of customers
t	Number of days of the period
m	Number of available vehicles
w	Maximum number of daily routes per vehicle
C	Set of available delivery-day patterns
P	Set of available vehicle types
K	Set of available vehicles
$C_i \subseteq C$	Set of feasible delivery-day patterns for customer i
$P_i \subseteq P$	Set of feasible vehicle types for customer i
$K_i \subseteq K$	Set of feasible vehicles for customer i
q_i	Demand per service for customer i
Q_k	Capacity of vehicle k
D_k	Maximum operation time for vehicle k
c_k	Operation cost per unit of distance travelled for vehicle k
d_{ij}	Travel distance between customer i and customer j
t_{ij}	Travel time between customer i and customer j
ϕ_i	Unload time for customer i
μ_i	Load time at the depot for customer i

$$\sum_{j=0}^{n} \sum_{k \in K_i \cap K_j} \sum_{r=1}^{w} x_{ij}^{krl} - \sum_{h \in C_i} a_{hl} y_{ih} = 0 \quad (i = 1, \ldots, n;\ l = 1, \ldots, t) \tag{3}$$

$$\sum_{i=0}^{n} x_{ie}^{krl} - \sum_{j=0}^{n} x_{ej}^{krl} = 0 \quad (e = 1, \ldots, n;\ k \in K_e;\ r = 1, \ldots, w;\ l = 1, \ldots, t) \tag{4}$$

$$\sum_{j=0}^{n} x_{0j}^{krl} \leqslant 1 \quad (k = 1, \ldots, m;\ r = 1, \ldots, w;\ l = 1, \ldots, t) \tag{5}$$

$$\sum_{i=0}^{n} \sum_{j=0}^{n} q_i x_{ij}^{krl} \leqslant Q_k \quad (k = 1, \ldots, m;\ r = 1, \ldots, w;\ l = 1, \ldots, t) \tag{6}$$

$$\sum_{i=0}^{n} \sum_{j=0}^{n} \sum_{r=1}^{w} (\mu_i + t_{ij} + \phi_i) x_{ij}^{krl} \leqslant D_k \quad (k = 1, \ldots, m;\ l = 1, \ldots, t) \tag{7}$$

$$\sum_{i, v_i \in S} \sum_{j, v_j \in S} x_{ij}^{krl} \leqslant |S| - 1 \quad (k = 1, \ldots, m;\ r = 1, \ldots, w;\ l = 1, \ldots, t;\ \forall S \subseteq V \backslash \{0\};\ |S| \geqslant 2) \tag{8}$$

$$x_{ij}^{krl} \in \{0, 1\} \quad (i = 0, \ldots, n;\ j = 0, \ldots, n;\ k = 1, \ldots, m;\ r = 1, \ldots, w;\ l = 1, \ldots, t) \tag{9}$$

$$y_{ih} \in \{0, 1\} \quad (i = 0, \ldots, n;\ \forall h \in C_i) \tag{10}$$

Constraints (2) mean that one feasible delivery pattern must be assigned to each customer while constraints (3) guarantee that each customer is visited only on the days corresponding to the assigned pattern. The role of constraints (4) is to ensure that when a vehicle arrives at a customer on a given day it also leaves that customer on the same day on the same vehicle on the same route. Constraints (5) ensure that each vehicle leaves the depot (if at all) only once per route. Limits on vehicle capacity and operation time are imposed through constraints (6) and (7). Finally, constraints (8) are standard subtour elimination constraints.

Note that if we use $K_i = K$ for each customer i, that is, there are no accessibility constraints, and $w = 1$, that is, only one route per vehicle per day are allowed, then this formulation reduces to the formulation proposed by Cordeau *et al* (1997) for the PVRP.

3. Overview of the algorithm

Our tabu search algorithm for the SDMTPVRP is based on that developed by Cordeau *et al* (1997) for the PVRP and solves a generalization of that problem. Note here that the Cordeau *et al* (1997) algorithm itself shares many features with the earlier tabu search algorithm developed for the VRP by Gendreau *et al* (1994). Although the structure of the iterative process within our algorithm is similar to that in Cordeau *et al* (1997), there are key differences with respect to:

- the definition of the attributes of a solution;
- the construction of the neighbourhood of a solution;
- the evaluation of the objective function and
- the construction of the initial solution.

Moreover, as Table 1 indicates, our algorithm for solving SDMTPVRP deals with a more general problem than the algorithm for the PVRP presented by Cordeau *et al* (1997).

As for previous algorithms, Cordeau *et al* (1997) and Gendreau *et al* (1994), our tabu search algorithm uses the GENIUS heuristic, developed by Gendreau *et al* (1992) for the insertion and removal of customers from routes. We first present our heuristic to generate the initial solution from which start the search. This heuristic shares some features with that presented in Cordeau *et al* (1997) for the same task, but in this case several steps have been added in order to cope with the accessibility constraints and multiple vehicle trips. We then explain the general functioning of the search procedure, followed by a complete description of our tabu search algorithm.

3.1. Construction of an initial solution

For the construction of an initial solution, s^0, from which to start the search procedure, we apply the following heuristic:

1. If we have Euclidean coordinates for the customers, sort them in ascending order of the angle between the line that connects the customer with the depot and the horizontal. Otherwise, use an arbitrary order.
2. Assign to each customer i a delivery-day pattern randomly chosen from those in the set C_i.
3. Let j be a customer randomly chosen from those nearer to the depot.
4. Let r_{kl} be the route for vehicle k that is currently being filled on day l. We initialize $r_{kl} := 1$ for each vehicle and each day.

5. For the sequence of customers $i = j, j+1, \ldots, n, 1, \ldots, j-1$, for each day l in which customer i has delivery assigned follow these steps:
   (a) Set k equal to the last vehicle in ascending order of capacity within the set of feasible vehicles for customer i.
   (b) Compute, using the GENIUS heuristic, the insertion cost of customer i in route r_{kl} of vehicle k on day l. The key parameter for this heuristic is p, the number of neighbours of each customer that we consider when we evaluate potential insertion positions.
   (c) If the insertion results in a violation of the total operation time of the vehicle on that day:
      - If $k > 1$, with 1 being the first vehicle in ascending order of capacity in the list of feasible vehicles for the customer, set $k := k - 1$, so that k becomes the next vehicle in descending order in the former list. Go to Step (b).
      - Otherwise go to Step (e).
   (d) If the insertion results in a violation of the capacity of vehicle k on route r_{kl} on day l:
      - If $r_{kl} < w$, set $r_{kl} := r_{kl} + 1$. Go to Step (b).
      - Otherwise, if $k > 1$, set $k := k - 1$. Go to Step (b).
   (e) Insert customer i in route r_{kl} of vehicle k on day l using the GENIUS heuristic.

For each customer, each set of feasible vehicles is sorted in ascending order of capacity. At Step 5 each customer is inserted in routes on the days of its assigned delivery-day pattern. On each day, we try to insert the customer in routes of vehicles in its set of feasible vehicles, starting with the vehicle of maximum capacity. At Step 5(c) if the insertion violates the total operation time of the vehicle, then we try to insert the customer in the next vehicle in the decreasing ordered list of feasible vehicles for that customer. At Step 5(d) if the insertion violates the capacity of the current vehicle route then we try to insert it in another available route for the same vehicle if possible or we try to insert it in the next vehicle in the decreasing ordered list of feasible vehicles.

Note that this initial solution heuristic cannot guarantee to produce a feasible solution. If we cannot find a feasible insertion a customer ends up inserted into the last assigned route of its feasible vehicle with smallest capacity (Step 5e).

3.2. Search procedure

An efficient way of handling side constraints in a local search process for VRPs is through the use of two objectives (Cordeau *et al*, 2002). The first, $c(s)$, computes the routing cost of solution s. The second, $f(s)$, is the sum of $c(s)$ and two penalization terms associated with violations of each side constraint, one proportional to the total over-capacity of the routes and the other proportional to the over-time in the total operation time of the vehicles:

$$f(s) = c(s) + \alpha q(s) + \beta d(s) \tag{11}$$

$$c(s) = \sum_{i=0}^{n}\sum_{j=0}^{n}\sum_{k\in K_i\cap K_j}\sum_{r=1}^{w}\sum_{l=1}^{t} d_{ij}c_k x_{ij}^{krl} \tag{12}$$

$$q(s) = \sum_{l=1}^{t}\sum_{k=1}^{m}\sum_{r=1}^{w}\left[\left(\sum_{i=0}^{n}\sum_{j=0}^{n} q_i x_{ij}^{krl}\right) - Q_k\right]^{+} \tag{13}$$

$$d(s) = \sum_{l=1}^{t}\sum_{k=1}^{m}\left[\left(\sum_{i=0}^{n}\sum_{j=0}^{n}\sum_{r=1}^{w}(\mu_i + t_{ij} + \phi_i)x_{ij}^{krl}\right) - D_k\right]^{+} \tag{14}$$

where α and β are positive self-adjusting penalty parameters and $[z]^+ = \max(0, z)$. Initially, α and β are set equal to 1 and are periodically increased or decreased throughout the search according to whether previous solutions were feasible or infeasible. According to Cordeau *et al* (2002), this way of proceeding means that the search will probably evolve through a mix of feasible and infeasible solutions, thus reducing the probability of becoming trapped in a local minimum. They also point to another algorithmic advantage of this device, that the search can operate with relatively simple moves, such as removing a customer from its current route and inserting it into a different route.

Our heuristic algorithm, TS-ABB, for the SDMTPVRP is based on the tabu search technique and makes use of the notation shown in Table 3. Our algorithm starts with an initial

Table 3 Notation used for the description of the tabu search algorithm

$B(s)$	Attribute set of the solution s
$c(s)$	Cost of the solution s
$q(s)$	Over-capacity of the solution s
$d(s)$	Over-time of the solution s
$f(s)$	Objective function of the solution s
$g(s)$	Penalised cost function of the solution s
$M(s)$	Subset of $N(s)$
$N(s)$	Neighbourhood of the solution s
p	Neighbourhood size for the GENIUS heuristic
R	Set of routes associated with solution s
$s, \bar{s}, s'$	Solutions
s^*	Best solution found
α	Penalizing factor for the over-capacity
β	Penalizing factor for the over-time
γ	Factor used to adjust the intensity of the diversification
δ	Parameter used to update α and β
η	Total number of iterations allowed
θ	Number of iterations during which an attribute is tabu
λ	Iteration counter
ρ_{ikrl}	Number of times the attribute (i, k, r, l) has been added to the actual solution
σ_{ikrl}	Aspiration level of the attribute (i, k, r, l)
τ_{ikrl}	Last iteration for which the attribute (i, k, r, l) remains tabu

solution found as explained above, sets it as the current solution and keeps moving from the current solution s to a neighbouring solution $s' \in N(s)$ until a chosen number of iterations (η) has elapsed. $N(s)$ is the neighbourhood of a solution s and is composed of all the solutions that can be reached by performing one of the following changes to the solution s:

1. Remove a customer i from a route r on day l and insert it into another route r' on the same day using the GENIUS heuristic. The insertion can be made into a route r' belonging to:
   (a) the same vehicle k,
   (b) another vehicle k' whose type belongs to the set K_i.

   We refer to this as a type I move.
2. Replace the pattern of deliveries h for customer i with another pattern $h' \in C_i$. Customer i is removed from each route on the days in pattern h and is inserted in that route for each day of the new pattern h' that minimizes, using the GENIUS heuristic, the increase in $f(s)$. For any days common to both patterns the customer is not moved. We refer to this as a type II move. Note here that ideally we would like $f(s)$ to decrease, see Equation (11), but since this cannot be guaranteed we use the smallest increase in $f(s)$ in deciding the insertion.

For each customer i there will be $|K_i|w - 1$ type I moves, since the customer can (potentially) be added into any route for any vehicle on the same day, there being $|K_i|$ such vehicles and w such routes from which we must exclude the current vehicle route. For the same customer there will be $|C_i| - 1$ type II moves, as there are $|C_i|$ delivery-day patterns for this customer from which we must exclude the current delivery-day pattern.

Each solution s has an attribute set $B(s) = \{(i, k, r, l)$: customer i is visited by vehicle k in its route r on day $l\}$ associated with it. In this way, the transition from a solution s to a solution $s' \in N(s)$ can be viewed as changes in the attribute set $B(s)$. Note here though that this attribute set only specifies which customers are visited by which vehicle on which route on which day. The actual order that customers are visited on any particular vehicle route is decided via the GENIUS heuristic of Gendreau *et al* (1992).

When a customer i is removed from route r of vehicle k on day l, inserting it again into the same route is prohibited, declared tabu, for θ iterations. We denote by τ_{ikrl} the number of the last iteration for which the attribute (i, k, r, l) remains tabu.

We denote by σ_{ikrl} the aspiration level of the attribute (i, k, r, l), defined as the cost of the best feasible solution found with that attribute. At the beginning, the aspiration level of an attribute is set equal to the cost of the initial solution $c(s^0)$ if the attribute belongs to the attribute set of that solution and the solution is feasible. It is set equal to ∞ otherwise. Every time a feasible solution s is found during the search procedure, the aspiration level of each of its attributes is updated to $\min\{\sigma_{ikrl}, c(s)\}$.

In order to avoid the risk of cycling, a subset $M(s) \subseteq N(s)$ is created with all those solutions $\bar{s} \in N(s)$ such that at least one of the attributes that must be changed in s to obtain $\bar{s}$ is not tabu or $c(\bar{s})$ is less than the aspiration level of this attribute.

In order to diversify the search, for all those solutions $\bar{s} \in M(s)$ that lead to an increase in the value of $f(s)$, the cost function $f(\bar{s})$ is penalized, giving more weight to those attributes that have been more frequently added to the current solution. The penalty term is proportional to the product of $c(s)$, ρ_{ikrl} and $\sqrt{nmwt}$, where ρ_{ikrl} is the number of iterations for which the attribute (i, k, r, l) has been added to the current solution. The $\sqrt{nmwt}$ factor is used to compensate for problem size and was first proposed by Taillard (1993) in the context of the VRP and was later used by Cordeau *et al* (1997) in their tabu search algorithm for the PVRP. A constant factor γ is used to adjust the intensity of the diversification. Limited computational experience has indicated that these diversification factors have beneficial effects in our algorithm with respect to the quality of results obtained.

At each iteration, the values of α and β, penalty parameters for the side constraints in the objective function, are updated by a factor $(1 + \delta) > 1$. If the current solution is feasible with respect to vehicle capacity the value of α is divided by $(1+\delta)$; otherwise it is multiplied by $(1 + \delta)$. β is adjusted in a similar manner.

3.3. Tabu search algorithm

The algorithm starts with an initial solution s^0, is initialized with the parameter vector $(\delta, \gamma, \theta, \eta, p)$, and returns the best solution found s^*, if any. We assume that $c(s^*) = \infty$ if no feasible solution has yet been found. The steps followed by the algorithm are:

1. If s^0 is feasible, set $s^* := s^0$. Set $\alpha := 1$ and $\beta := 1$.
2. For each attribute (i, k, r, l):
   (a) Set $\rho_{ikrl} := 0$ and $\tau_{ikrl} := 0$.
   (b) If $(i, k, r, l) \in B(s^0)$ and s^0 is feasible, set $\sigma_{ikrl} := c(s^0)$; otherwise, set $\sigma_{ikrl} := \infty$.
3. Set $s := s^0$.
4. For $\lambda = 1, \ldots, \eta$,
   (a) Set $M(s) := \emptyset$.
   (b) For each $\bar{s} \in N(s)$, set $M(s) := M(s) \cup \bar{s}$ if there exists $(i, k, r, l) \in B(\bar{s}) \backslash B(s)$ such that:
      - $\tau_{ikrl} < \lambda$; or
      - $\bar{s}$ is feasible and $c(\bar{s}) < \sigma_{ikrl}$.
   (c) For each $\bar{s} \in M(s)$,
      - If $f(\bar{s}) \geqslant f(s)$, set $g(\bar{s}) := f(\bar{s}) + \gamma c(s)\sqrt{nmwt}$ $\sum_{(i,k,r,l)\in B(\bar{s})\backslash B(s)} \rho_{ikrl}/\lambda$.
      - Otherwise, set $g(\bar{s}) := f(\bar{s})$.

(d) Identify the solution $s' \in M(s)$ such that $g(s') = \min\{g(\bar{s}), \bar{s} \in M(s)\}$. We now move from the current solution s to this neighbouring solution s' by suitable adjustment of the routes associated with s.

(e) Let R represent the set of routes associated with the solution s and:

- For each attribute $(i, k, r, l) \in B(s) \backslash B(s')$,
  - Adjust R by removing customer i from route r of vehicle k on day l using the GENIUS heuristic.
  - Set $\tau_{ikrl} := \lambda + \theta$.
- For each attribute $(i, k, r, l) \in B(s') \backslash B(s)$,
  - Adjust R by inserting customer i into route r of vehicle k on day l using the GENIUS heuristic.
  - Set $\rho_{ikrl} := \rho_{ikrl} + 1$.

(f) Let s represent the solution associated with R. If s is feasible:

- Set $s^* := s$ if $c(s) < c(s^*)$.
- For each $(i, k, r, l) \in B(s)$, set $\sigma_{ikrl} := \min\{\sigma_{ikrl}, c(s)\}$.

(g) If $q(s) = 0$, set $\alpha := \alpha/(1 + \delta)$; otherwise set $\alpha := \alpha(1 + \delta)$

(h) If $d(s) = 0$, set $\beta := \beta/(1 + \delta)$; otherwise set $\beta := \beta(1 + \delta)$

Steps 1–3 constitute the initialization process of the search procedure while Step 4 constitutes the iterative process. Within this iterative process, Steps (a)–(d) involve the construction of a subset of the neighbourhood of the current solution from which a good neighbouring solution to move to is selected. Step (e) corresponds to the removal and insertion of customers in appropriate routes guided by the attribute set differences between the current solution and its chosen neighbour. Note here that, because of the way the neighbourhood is defined, any neighbour automatically satisfies the constraints relating to delivery-day patterns. At Step (f) if a feasible solution has been reached attribute aspiration levels are updated, as is the best solution found. At Steps (g) and (h) the penalizing factors for the over-capacity and over-time are also updated depending on the new current solution. Note that we apply adaptive penalties here (unlike some previous work, eg Taillard *et al* (1996) which used constant penalty values).

An interesting feature of our algorithm is that, considering practical constraints, it allows the possibility of including, in a very easy way, constraints such as:

- prevent a specific vehicle visiting a specific customer, even if the vehicle belongs to a feasible vehicle type for that customer;
- force a specific vehicle to visit a specific customer.

This is because the algorithm uses the set K_i of feasible vehicles for each customer i (created from the set P_i of feasible vehicle types), so the constraints referred to above can be dealt with by amending K_i appropriately.

Note here that the above algorithm reduces to that in Cordeau *et al* (1997) when both $|P|$, the number of different vehicle types, and w, the number of routes per vehicle are one. Nevertheless, it should be stressed here that our algorithm is capable of dealing with a much more general problem than the algorithm of Cordeau *et al* (1997), namely any problem with $|P| > 1$ and/or $w > 1$. Moreover, as discussed below, the parameter settings for the standard algorithms in the two cases are different.

3.4. Computational complexity for TS-AAB

A complexity analysis for TS-ABB is necessarily based on the complexity of each iteration. In each iteration, the most time-consuming process is that of calculating the cost of the solutions in the new neighbourhood, because in that process procedures for inserting a customer into a route, or removing it from a route, are used. These procedures are taken from the GENIUS heuristic of Gendreau *et al* (1992).

As discussed above, for each customer i on a particular route on a particular day there will be $|K_i|w - 1$ type I moves (involving a move for the customer between routes on the same day). If e_i is the total number of visits in the period for customer i then the total number of type I moves is $\sum_{i=1}^{n}(|K_i|w - 1)e_i$. Each of these moves requires one insertion using GENIUS.

As discussed above, for each customer i there will be $|C_i| - 1$ type II moves (involving a change in the delivery-day pattern). Each type II move involves $|K_i|we_i$ insertions of the customer into routes, as on each day of the new delivery-day pattern insertion must be examined in each route of each feasible vehicle in order to ascertain the insertion that minimises the increase in $f(s)$. The total number of GENIUS insertions is therefore $\sum_{i=1}^{n}|K_i|we_i(|C_i| - 1)$.

With respect to the computational complexity of removing customers from routes, this need not concern us here as this has the same computational complexity, $O(p^4)$ where p is the size of the neighbourhood in GENIUS, as the GENIUS insertion procedure and the number of insertions evaluated dominates the number of removals evaluated.

Hence the overall computational complexity of each iteration is $O(p^4)(\sum_{i=1}^{n}(|K_i|w - 1)e_i + \sum_{i=1}^{n}|K_i|we_i(|C_i| - 1)) = O(p^4 w \sum_{i=1}^{n}|K_i||C_i|e_i)$. It can be seen from this expression that the value of p, the size of the neighbourhood in GENIUS, has a major influence on the computation time (per iteration). Computation time also depends on the number of feasible vehicles per customer $|K_i|$, the maximum number of daily routes per vehicle w, the number of feasible delivery patterns per customer $|C_i|$ and the total number of visits in the period e_i. These last two are often related one to another: the lower the total number of visits in the period, the greater the number of feasible delivery patterns, and vice versa.

4. Computational experiments

4.1. Experiments on randomly generated test problems

Since, to the best of our knowledge, this paper is the first time SDMTPVRP has appeared in the literature, there are no test problems for the problem with which to test the tabu search algorithm TS-ABB we have developed, nor previous results to compare with. Because of this we randomly generated our own test problems with different sets of characteristics. In this section, we explain how these test problems were generated, how they were used to tune the five parameters associated with our algorithm and the results obtained with the algorithm on these test problems.

4.1.1. Characteristics of the test problems We generated 10 test problems with periods of 2, 4, 5 and 6 days. The number of services per period (delivery frequencies) in each test problem depend on the days of the period and were {1, 2} for 2 days, {1, 2, 4} for 4 days, {1, 2, 5} for 5 days and {1, 2, 3, 6} for 6 days. Given a delivery frequency and the length of the period, a set of available delivery-day patterns can be easily generated.

There are test problems with 2, 3 and 5 different vehicle types. In our formulation and algorithm, the vehicle type is used to determine whether (or not) a vehicle can be used to deliver to a customer. As such vehicles belonging to the same type need not have exactly the same characteristics for capacity, total daily operation time and operation cost. In the generated test problems, vehicles of the same type have the same operation cost and total daily operation time but may have different capacities.

The locations of the customers were randomly generated using a procedure presented by Cordeau *et al* (1997). This procedure generates locations grouped around a certain number of centres. The demand q_i for each customer is established randomly and independently according to a discrete uniform distribution in [1, 25]. The unload time at the customer, and load time at the depot, is taken to be proportional to customer demand. The travel time of the route is taken to be proportional to the travel distance.

The assignment of delivery frequencies to the customers in the test problems is made randomly, that is, for a given period, each delivery frequency available is assigned to a specified percentage of customers. The assignment of sets of feasible vehicle types is made in the same way. For the set of customers with a given delivery frequency, each set of feasible vehicle types is assigned to a percentage of these customers.

Table 6 shows the main characteristics of the ten randomly generated test problems. Figure 1 and Tables 4 and 5 present a more detailed explanation of the characteristics of test problem 4 as an example. This test problem involves 150 customers who have to be delivered to over a 4-day period using four vehicles that are of three different types. It can be seen from Table 4 that of these 150 customers 25 have delivery frequency 1 and can only be visited by a vehicle of type 1. By contrast 10 customers have delivery frequency 2 (ie must be visited two times over the 4-day period) and can only be visited by vehicles of types 2 or 3. The characteristics of the vehicles are shown in Table 5 so that, for example, vehicle 4 is of type 3, has a load capacity of 175 units, a maximum operation time of 1800 time units and an operation cost per unit of distance travelled of 3. Note how vehicles 1 and 2, although of the same type, have different load capacities. All SDMTPVRP data sets used in this paper are publicly available from OR-Library (Beasley, 1990) http://people.brunel.ac.uk/~mastjjb/jeb/orlib/sdmtpvrpinfo.html.

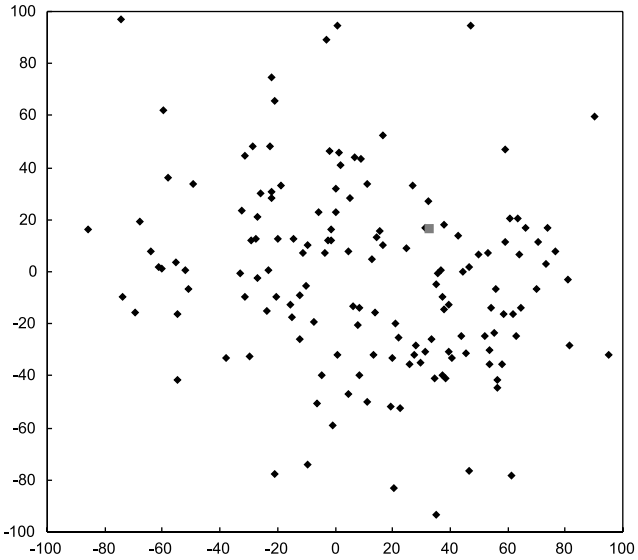

Figure 1 Location of customers in test problem 4.

Table 4 Number of customers for each delivery frequency and vehicle types set in test problem 4

	Vehicle-type Set		
Delivery Frequency	{*1*}	{*1*, *2*}	{*2*,*3*}
1	25	25	30
2	10	15	10
4	15	10	10

Table 5 Characteristics of the vehicles in test problem 4

Vehicle k	*Type*	Q_k	D_k	c_k
1	1	50	1800	1
2	1	60	1800	1
3	2	125	1800	2
4	3	175	1800	3

4.1.2. Sensitivity analysis Our algorithm involves five parameters: (γ, δ, θ, η, p) (see Table 3). A series of experiments have been conducted in order to define values for these parameters for the 'standard algorithm'. It is important to note

Table 6 Results on SDMTPVRP randomly generated test problems

I	n	t	$\|P\|$	m	*Best known*	*Best*	T	%
1	50	2	2	2	2690.41	2695.96	0.78	0.21
2	75	4	2	2	6245.17	6245.17	1.35	0.00
3	100	4	2	3	7208.69	7305.18	1.69	1.34
4	150	4	3	4	12499.03	12774.28	2.23	2.20
5	200	4	3	5	15116.86	15589.62	3.44	3.13
6	288	5	3	6	20437.29	20658.37	8.27	1.08
7	350	5	3	8	32592.80	33126.81	8.43	1.64
8	500	5	5	10	101418.52	102329.36	8.15	0.90
9	750	6	5	13	130635.46	131334.72	12.90	0.54
10	1000	6	5	13	118703.09	121479.08	19.74	2.34
Average							6.70	1.34

here that once these parameter values have been defined then the same set of parameter values are used in *every* run of the algorithm on *every* test problem reported below (so we make no attempt to refine parameter values for the particular special cases of the SDMTPVRP considered below).

We start with the set of values proposed by Cordeau *et al* (1997) in their algorithm for the classic PVRP. These values are: γ=0.015, δ=0.5, θ=7.5 $\log_{10}(n)$, η=15 000 and p=3. We have performed a sequential analysis, parameter by parameter, maintaining the values of all the parameters except one, whose value is changed within a predefined range. Parameter γ has been studied in the range [0.001, 0.25], parameter δ in the range [0.05, 10] and parameter θ in the range [7, 27]. The other two parameters are studied jointly because the execution time of the algorithm increases with the increase in both of them. Seven alternatives for the pair of values (η, p) have been tried.

Three complete series of executions have been carried out on the randomly generated test problems 3, 4, 5, 7 and 8. In each one of the series, all the executions with different sets of parameters values use the same initial solution. From these experiments, the best values for the parameters are the following: γ=0.02, δ=2, θ=7.5 $\log_{10}(n)$, η=15 000 and p=3. In the case of parameter θ, our algorithm is quite insensitive to variations in this parameter in the given range of values, so we opted to maintain the value proposed by Cordeau *et al* (1997). In the case of the pair of values (η, p) we obtain the same best values as the previous authors.

4.1.3. Results on randomly generated test problems Table 6 presents the results obtained on each test problem of the SDMTPVRP that was generated. Column *Best Known* shows the cost of the best known solution, which was either identified during parameter sensitivity analysis for those test problems used in that task, or by performing several runs using different parameter settings for the other test problems. Column *Best* shows the best solution found with the algorithm in its standard version (ie using the parameter values decided as discussed above). Column % shows the percentage deviation between the solution with the standard algorithm and the best known solution, that is 100(TS-ABB solution−Best known solution)/Best known solution. Column T is the computation time in minutes on a DELL Dimension 8200 (1.6 GHz, 256 MB RAM) personal computer.

The average percentage deviation between the solution with the standard algorithm and the best known solution is just 1.34%. The computation time required is reasonable, even for the test problems of large size.

It is clear from Table 1 that SDMTPVRP is the most general of all the problems shown in that table. Hence our TS-ABB algorithm for SDMTPVRP can be used to solve *any* of the other VRPs shown there. Below we show computationally how our algorithm performs on such problems.

4.2. Results on PVRP test problems

When there exists only one type of vehicle (so no accessibility restrictions) and each vehicle is only allowed to do one route per day, the SDMTPVRP reduces to the classic PVRP. We have tested our TS-ABB algorithm with the test problems for the PVRP presented in Cordeau *et al* (1997).

Table 7 presents the results obtained by Cordeau *et al* (1997) and those obtained with our algorithm. Column *Best Known* shows the best known solution as taken from Cordeau *et al* (1997), while columns *Best* show the best solution found with the Cordeau *et al* (1997) algorithm and our TS-ABB algorithm in their standard versions. Emphasized in italics for TS-ABB are those solutions better than, or equal to, the previously best known solution. Columns % show the percentage deviation between the solution with the standard algorithm and the best known solution. Column $T^{(1)}$ is the computation time in minutes on a Sun Sparcstation 10 and $T^{(2)}$ is the computation time in minutes on a DELL Dimension 8200 (1.6 GHz, 256 MB RAM) personal computer. Utilizing Dongarra (2006, http://www.netlib.org/benchmark/performance.ps), we have made an *approximate* estimate of the time that Cordeau *et al* (1997) would have required had their algorithm been run on our computer, and this is also shown in Table 7.

Table 7 Results on PVRP test problems in Cordeau *et al* (1997)

Test problem						*Cordeau et al (1997)*				*TS-AAB*		
I	*n*	*m*	*t*	*D*	*Q*	*Best known*	*Best*	$T^{(1)}$	%	*Best*	$T^{(2)}$	%
a	48	2	4	500	200	2209.02	2234.23	3.86	1.14	*2209.02*	0.85	*0.00*
b	96	4	4	480	195	3799.28	3836.49	9.79	0.98	3812.31	1.71	0.34
c	144	6	4	460	190	5218.13	5277.62	18.62	1.14	5277.46	2.95	1.14
d	192	8	4	440	185	6012.79	6072.67	28.24	1.00	*6002.45*	4.32	*−0.17*
e	240	10	4	420	180	6769.80	6769.80	36.62	0.00	*6745.57*	5.04	*−0.36*
f	288	12	4	400	175	8422.64	8462.37	50.97	0.47	*8399.65*	6.94	*−0.27*
g	72	3	6	500	200	4997.41	5000.90	9.63	0.07	*4996.75*	1.65	*−0.01*
h	144	6	6	475	190	7094.52	7183.39	24.05	1.25	7160.84	3.62	0.93
i	216	9	6	450	180	10370.45	10507.34	45.65	1.32	*10348.88*	6.19	*−0.21*
j	288	12	6	425	170	13370.04	13629.25	71.56	1.94	*13321.04*	8.21	*−0.37*
Average								29.90	0.93		4.15	0.10
Approximate average time, DELL Dimension 8200								0.87				

It is clear from Table 7 that our TS-ABB algorithm dominates the algorithm of Cordeau *et al* (1997) in terms of solution quality, as for all 10 problems considered our algorithm finds a better solution than the algorithm of Cordeau *et al* (1997). Computationally, as best as we can judge from the approximate nature of the comparison made, our algorithm is on average slower than the algorithm of Cordeau *et al* (1997). However our algorithm does not require an excessive amount of time, even for the larger problems, and it improves upon the best known solution for six of the ten problems in Table 7.

4.3. Results on SDVRP test problems

The algorithm has been tested on the test problems presented in Cordeau and Laporte (2001) where there are different types of vehicles, accessibility restrictions and every vehicle does only one route per day. The planning period is one day, hence it is a VRP.

The way in which Cordeau and Laporte (2001) solve the problem is by using the relation between the structure of the SDVRP and the structure of the PVRP, relating the types of vehicles in the SDVRP with the days of the period in the PVRP and the feasible types of vehicles for each customer in the SDVRP to the feasible patterns of delivery days for each customer in the PVRP. In this way they can apply, without modification, the algorithm developed for the PVRP in Cordeau *et al* (1997) to solve the SDVRP with results that improve upon those obtained for the same problem by Chao *et al* (1999).

The results obtained by Cordeau and Laporte (2001) and those obtained with our TS-ABB algorithm are detailed in Table 8. Column *Best Known* shows the best known solution that appeared in Cordeau and Laporte (2001), while columns *Best* show the best solution found with the Cordeau and Laporte (2001) algorithm and our TS-ABB algorithm in their standard versions. Emphasized in italics for TS-ABB are those solutions better than, or equal to, the previously best known solution. Columns % show the percentage deviation between the solution with the standard algorithm and the best known solution. Column $T^{(1)}$ is the computation time in minutes on a Sun Ultra 2 (300 MHz) workstation and $T^{(2)}$ is the computation time in minutes on a DELL Dimension 8200 (1.6 GHz, 256 Mb RAM) personal computer.

It is clear from Table 8 that, on average, our TS-ABB algorithm performs better than the algorithm of Cordeau and Laporte (2001). For nine of the 12 problems in Table 8 our algorithm produces a solution that is better than the solution produced by the algorithm of Cordeau and Laporte (2001). Computationally, as best as we can judge from the approximate nature of the comparison made, our algorithm is on average faster than the algorithm of Cordeau and Laporte (2001). Computationally our algorithm does not require an excessive amount of time, even for the larger problems, and it improves upon the best known solution for six of the twelve problems in Table 8.

4.4. Results on MTVRP test problems

The algorithm was tested on the test problems for the MTVRP proposed by Taillard *et al* (1996). These test problems are defined in the following way: based on the same graph, demands and vehicle capacities as problems 1–5 and 11–12 in Christofides *et al* (1979), several test problems are generated for the MTVRP with different values for the number of vehicles, m, and the maximum travel time or maximum route length, T (taking into account that the travel time is proportional to the route length and the proportionality constant is one). For each value of m, two different values of T are used: $T_1 = [1.05z^*/m]$ and $T_2 = [1.10z^*/m]$, where $[x]$ is the value of x rounded to the nearest integer and z^* is the value of the best solution for the classic VRP without specifying the number of vehicles as found by the heuristic algorithm of Rochat and Taillard (1996). When the maximum travel time is exceeded and no feasible solution can be found, the overtime is penalised by a factor of two.

Table 8 Results on SDVRP test problems in Cordeau and Laporte (2001)

Test problem						*Cordeau and Laporte (2001)*				*TS-AAB*		
I	*n*	*m*	$\|P\|$	*D*	*Q*	*Best known*	*Best*	$T^{(1)}$	%	*Best*	$T^{(2)}$	%
1	48	1	4	500	A	1384.15	1385.47	1.92	0.10	1416.28	0.43	2.32
2	96	2	4	500	A	2320.97	2322.08	4.27	0.05	2410.29	1.19	3.85
3	144	3	4	500	A	2623.31	2626.97	7.12	0.14	*2544.03*	2.18	*−3.02*
4	192	4	4	500	A	3500.79	3575.79	10.10	2.14	*3374.09*	2.98	*−3.62*
5	240	5	4	500	A	4479.34	4479.34	13.18	0.00	*4352.84*	3.98	*−2.82*
6	288	6	4	500	A	4546.79	4583.46	15.84	0.81	4609.52	4.98	1.38
7	72	1	6	500	B	1955.11	1990.68	3.39	1.82	*1924.74*	0.78	*−1.55*
8	144	2	6	500	B	3082.32	3103.40	7.31	0.68	*2930.64*	1.89	*−4.92*
9	216	3	6	500	B	3664.22	3732.06	13.47	1.85	3690.22	3.65	0.71
10	288	4	6	500	B	4739.43	4782.17	16.03	0.90	4773.54	5.07	0.72
11	1008	21	4	500	A	13227.96	13479.90	115.69	1.90	*13139.93*	24.35	*−0.67*
12	720	10	6	500	B	9621.99	9938.57	69.32	3.29	9661.61	16.01	0.41
Average								23.14	1.14		5.62	*−0.60*
Approximate average time, DELL Dimension 8200								11.82				

In terms of vehicle capacity Q a problem labelled A has vehicle capacities {100, 150, 200, 250} while a problem labelled B has vehicle capacities {125, 150, 175, 200, 225, 250}.

The purpose of using T_1 and T_2 is that for MTVRP problems it is difficult to generate problems to which a heuristic algorithm can find feasible solutions and increasing the value of T (the limit on vehicle travel time/distance), from T_1 to T_2, may be necessary in order to enable the algorithm to produce feasible solutions.

There are three papers in the literature that present computational experience with these test problems: Taillard *et al* (1996), Brandão and Mercer (1998) and Petch and Salhi (2003) and we consider each of these papers in turn.

4.4.1. MTVRP comparison: Taillard et al (1996) Taillard *et al* (1996) only presented detailed results for those test problems for which their algorithm did not find a feasible solution using T_1. Table 9 shows the comparison of the solutions obtained with their algorithm and the solutions obtained with our TS-ABB algorithm. Columns *Length* show the total length (time) for all the routes of the best solution found, while columns $c(s)$ show the cost of that solution including the penalization for the overtime. If *Length* and $c(s)$ are equal then the solution is feasible. Columns *Ratio* show the length (time) of the longest route divided by the value of T_1, so a value over one indicates that the longest route (and hence also the entire solution) is infeasible. Columns Difference (%) show the percentage difference between the Taillard *et al* (1996) result and our result, that is 100 (Taillard *et al* result−TS-ABB result)/TS-ABB result. Column T is the computation time in minutes on a DELL Dimension 8200 (1.6 GHz, 256 MB RAM) personal computer.

The results in Table 9 are presented so as to be exactly comparable to the results presented in Taillard *et al* (1996). A key point to realise about their results however is that, as the differences between *Length* and $c(s)$ show, all of their results correspond to infeasible solutions. By contrast we do find a feasible solution for test problem C3 with $m = 5$. It is clear from the columns headed *Ratio* that our TS-ABB algorithm reduces the degree of infeasibility associated with the longest route although, on average, our penalised solutions are just under one-half of 1% more costly than the solutions produced by the Taillard *et al* (1996) algorithm. Moreover note here that, although interpreting the results given in Taillard *et al* (1996) is not easy, our best interpretation is that the results for the Taillard *et al* (1996) algorithm, taken directly from Taillard *et al* (1996) and shown in Table 9, are the best over six runs of their algorithm (five independent runs followed by a sixth run using routes from the first five runs). By contrast our algorithm was only run once.

With respect to computation time Taillard *et al* (1996) do not give individual computation times for the test problems shown in Table 9. However, from the information they do present, and utilizing Dongarra (2006), we have made an *approximate* estimate that their average computation time over all 12 test problems would have been 7.20 min on our DELL Dimension 8200. Bearing in mind the nature of the approximation this is effectively equivalent to our computation time of 7.42 minutes as shown in Table 9.

Note here that we have only presented in Table 9 the results for T_1 for the test problems shown. This is because our algorithm finds feasible solutions for *all* of the 12 test problems shown in Table 9 when using T_2. By contrast the results presented in Taillard *et al* (1996) show that their algorithm only finds feasible solutions using T_2 for three of these 12 test problems.

In summary, therefore here it seems reasonable to conclude that:

- our TS-ABB algorithm is much better able to find feasible solutions for MTVRP probems than the algorithm of Taillard *et al* (1996)

Table 9 Results on MTVRP test problems in Taillard *et al* (1996)

Test problem				*Taillard et al (1996)*			*TS-AAB*				*Difference (%)*			*Petch and Salhi (2003)*	
Problem	*m*	z^*	T_1	*Length*	*c(s)*	*Ratio*	*Length*	*c(s)*	*Ratio*	*T*	*Length*	*c(s)*	*Ratio*	*Ratio*	*Difference (%)*
C1	3	524.61	184	533.00	579.48	1.115	554.37	569.54	1.041	2.66	4.01	*−1.72*	*−6.64*	1.026	1.46
	4		138	546.29	565.27	1.027	547.10	564.10	1.027	2.70	0.15	*−0.21*	*0.00*	1.085	*−5.35*
C2	6	835.26	146	841.60	857.19	1.032	851.59	868.99	1.052	3.60	1.19	1.38	1.94	1.019	3.24
	7		125	843.60	878.29	1.073	855.58	878.05	1.050	3.75	1.42	*−0.03*	*−2.14*	1.064	*−1.32*
C3	5	826.14	173	829.50	853.23	1.062	836.01	836.01	0.983	7.38	0.78	*−2.02*	*−7.44*	1.052	*−6.56*
	6		145	842.85	861.18	1.032	839.58	840.03	1.002	7.90	*−0.39*	*−2.46*	*−2.91*	1.001	0.10
C4	7	1028.42	154	1042.39	1074.16	1.033	1053.39	1071.99	1.041	10.91	1.06	*−0.20*	0.77	1.072	*−2.89*
	8		135	1049.02	1088.03	1.075	1062.53	1083.36	1.038	11.79	1.29	*−0.43*	*−3.44*	1.058	*−1.89*
C5	10	1291.44	136	1316.00	1331.09	1.024	1350.52	1364.42	1.045	14.90	2.62	2.50	2.05	1.064	*−1.79*
C11	4	1042.11	274	1042.11	1055.07	1.020	1097.05	1118.70	1.038	8.79	5.27	6.03	1.76	1.052	*−1.33*
C12	5	819.56	172	819.56	836.80	1.050	825.65	838.07	1.035	7.00	0.74	0.15	*−1.43*	1.000	3.50
	6		143	819.56	845.48	1.064	819.97	866.54	1.113	7.62	0.05	2.49	4.61	1.029	8.16
Average										7.42	1.52	0.46	*−1.07*		*−0.39*

- our TS-ABB algorithm requires approximately the same computation time as the algorithm of Taillard *et al* (1996)
- our TS-ABB algorithm, when both it and Taillard *et al* (1996) are unable to find a feasible solution, produces solutions that have a slightly higher total length and cost, but for which the degree of infeasibility associated with the longest route is smaller

4.4.2. MTVRP comparison: Brandão and Mercer (1998). Brandão and Mercer (1998) reported that their algorithm for MTVRP, when applied to the problems of Taillard *et al* (1996), also always produced feasible solutions using T_2 (as did TS-ABB). Table 10 shows the results reported by them using T_1, together with the results found by our TS-ABB algorithm, for all of the problems shown in Table 9 plus two extra problems (C5 with $m = 9$ and C12 with $m = 4$). For both TS-ABB and Taillard *et al* (1996), a feasible solution to problem C12 with $m = 4$ of length 819.56 is found. However, Brandão and Mercer (1998) state that for this problem their solution of length 819.56 is infeasible. We have no clear explanation for this discrepancy other than to note that we agree with the results of Taillard *et al* (1996).

From Table 10 it appears that, on average, with respect to solution quality our TS-ABB algorithm dominates the algorithm of Brandão and Mercer (1998) as the difference percentages are all negative. In other words, on average our algorithm produces shorter routes that are less infeasible.

With respect to computation time Brandão and Mercer (1998) do not give individual computation times for the test problems shown in Table 10. However, from the information they do present, and utilizing Dongarra (2006), we have made an *approximate* estimate that their average computation time over all 14 test problems would have been 0.84 min on our DELL Dimension 8200. This is clearly less than our computation time of 7.80 min as shown in Table 10, but the time required by TS-ABB does not seem unreasonable for results that dominate those of Brandão and Mercer (1998).

4.4.3. MTVRP comparison: Petch and Salhi (2003). The last two columns in Tables 9 and 10 give the results for Petch and Salhi (2003) on these test problems. The last column (labelled Difference (%)) gives the percentage difference between the ratio as found by TS-ABB and as found by Petch and Salhi (2003). Note here that in their paper Petch and Salhi (2003) do not present any information as to route length or cost for individual test problems.

With respect to these tables it can be seen that both TS-ABB and Petch and Salhi (2003) fail to find a feasible solution for the majority of problems. For one test problem (C12 with $m = 4$) both algorithms do find a feasible solution; there is one test problem where TS-ABB finds a feasible solution but Petch and Salhi (2003) do not (problem C3 with $m = 5$); and one test problem where TS-ABB fails to find a feasible solution but Petch and Salhi (2003) do (problem C12 with $m = 5$, since Petch and Salhi (2003) have a ratio value of one for that test problem, indicating that to within rounding errors, the longest route is feasible). On average, it is clear that our algorithm outperforms that of Petch and Salhi (2003) as the longest routes produced by TS-ABB have a lower degree of infeasibility.

With respect to computation time, Petch and Salhi (2003) do not give individual computation times for the test problems shown in Tables 9 and 10. However from the information they do present, and utilising Dongarra (2006), we have made an *approximate* estimate of the average time that their algorithm would take on our DELL Dimension 8200. Our estimate is 6.83 min per problem for Table 9 and 7.42 min per problem for Table 10. Bearing in mind the nature of the approximation these times are effectively equivalent to the times seen for TS-ABB in those tables.

4.5. Results on VRP test problems

Finally, our TS-ABB algorithm has been tested on the 14 test problems for the VRP in Christofides *et al* (1979). Between

Table 10 Results on MTVRP test problems in Brandão and Mercer (1998)

Test problem				*Brandão and Mercer (1998)*			*TS-AAB*				*Difference (%)*			*Petch and Salhi (2003)*	
Problem	*m*	z^*	T_1	*Length*	$c(s)$	*Ratio*	*Length*	$c(s)$	*Ratio*	*T*	*Length*	$c(s)$	*Ratio*	*Ratio*	*Difference (%)*
C1	3	524.61	184	556.34	575.73	1.041	554.37	569.54	1.041	2.66	*−0.35*	*−1.08*	*0.00*	1.026	1.46
	4		138	547.10	564.07	1.027	547.10	564.10	1.027	2.70	*0.00*	0.01	*0.00*	1.085	*−5.35*
C2	6	835.26	146	858.04	867.02	1.031	851.59	868.99	1.052	3.60	*−0.75*	0.23	2.04	1.019	3.24
	7		125	870.07	896.57	1.088	855.58	878.05	1.050	3.75	*−1.67*	*−2.07*	*−3.49*	1.064	*−1.32*
C3	5	826.14	173	845.89	845.89	0.999	836.01	836.01	0.983	7.38	*−1.17*	*−1.17*	*−1.60*	1.052	*−6.56*
	6		145	837.82	838.74	1.003	839.58	840.03	1.002	7.90	0.21	0.15	*−0.10*	1.001	0.10
C4	7	1028.42	154	1063.95	1102.63	1.071	1053.39	1071.99	1.041	10.91	*−0.99*	*−2.78*	*−2.80*	1.072	*−2.89*
	8		135	1069.54	1085.94	1.031	1062.53	1083.36	1.038	11.79	*−0.66*	*−0.24*	0.68	1.058	*−1.89*
C5	9	1291.44	136	1329.28	1359.75	1.056	1327.63	1330.96	1.011	14.10	*−0.12*	*−2.12*	*−4.26*	1.024	*−1.27*
	10		136	1336.92	1359.75	1.051	1350.52	1364.42	1.045	14.90	1.02	0.34	*−0.57*	1.064	*−1.79*
C11	4	1042.11	274	1069.24	1075.35	1.011	1097.05	1118.70	1.038	8.79	2.60	4.03	2.67	1.052	*−1.33*
C12	4	819.56	215	819.56	825.94	1.012	819.56	819.56	0.991	6.06	*0.00*	*−0.77*	*−2.08*	0.991	*0.00*
	5		172	828.94	841.14	1.036	825.65	838.07	1.035	7.00	*−0.40*	*−0.36*	*−0.10*	1.000	3.50
	6		143	819.56	847.85	1.072	819.97	866.54	1.113	7.62	0.05	2.20	3.82	1.029	8.16
Average										7.80	*−0.16*	*−0.26*	*−0.41*		*−0.42*

them, these include test problems with both capacity and duration constraints and test problems with only capacity constraints. These test problems have been extensively used to evaluate algorithms for the VRP.

The best known solutions for these test problems (from Cordeau *et al*, 2002) and those obtained with our TS-ABB algorithm are detailed in Table 11. Column δ shows the service time for every customer in those test problems that have route duration time restrictions. Column *Best* shows the best solution found with our TS-ABB algorithm in its standard version. Column % shows the percentage deviation between the solution with the standard algorithm and the best known solution. Emphasized in italics are those solutions obtained with our algorithm whose cost is equal to the best known solution. Column *T* is the computation time in minutes on a DELL Dimension 8200 (1.6 GHz, 256 MB RAM) personal computer.

At first sight we might conclude from Table 11 that our TS-ABB algorithm, despite not being specifically designed for the classical VRP problem, produces good quality results, with an average percentage deviation from the best known solution of under 1%. However, such a conclusion underestimates the vast amount of work that has been done in terms of the VRP, and the quality of heuristics that are available for the problem.

The recent paper of Tarantilis *et al* (2005) lists a significant number of heuristic algorithms for the VRP that have been presented in the literature, indeed that paper gives detailed results (for the same test problems as in Table 11) for 19 different heuristics. As discussed in Tarantilis *et al* (2005), it is often difficult to make a computational time comparison between such a large number of heuristics, for example due to factors such as the many different computers used and the manner in which authors set parameters associated with their heuristics. In their paper, Tarantilis *et al* (2005) report percentage deviation from best known solutions, but make no attempt to convert the differing computational times reported by other authors into a common base (eg by using Dongarra, 2006).

In terms of evaluating how our algorithm TS-ABB does in comparison with these heuristics then, rather than repeat here detailed results already reported in Tarantilis *et al* (2005), we will state that of the 19 heuristics reported in Tarantilis *et al* (2005) the average percentage deviation of 0.86% for TS-ABB (as shown in Table 11) is better than (or equal to) the performance of six of these heuristics, worse than 13 of these heuristics. In other words in terms of average percentage deviation 13 of the 19 heuristics considered in Tarantilis *et al* (2005) dominate TS-ABB.

Other authors have also reported comparative results (average percentage deviation) for a number of heuristics for the VRP:

- Berger and Barkaoui (2003): seven heuristics, five of which dominate TS-ABB;
- Cordeau *et al* (2005): nine heuristics, all of which dominate TS-ABB;
- Wassan (2006): five heuristics, all of which dominate TS-ABB.

Although there is some overlap between these papers in terms of the heuristics considered it is clear that TS-ABB is not competitive with state of the art heuristics for the VRP.

4.6. Remarks on the experiments

The maximum number of daily routes per vehicle, w, is a fixed parameter, with the same value for every vehicle within a test problem and depends on the characteristics of that problem. Obviously w depends on the relationship between vehicle

Table 11 Results on VRP test problems

Test problem						*Best known*	*TS-AAB*		
I	*n*	*m*	*Q*	*D*	δ		*Best*	*T*	*%*
C1	50	5	160	∞	0	524.61	*524.61*	0.79	*0.00*
C2	75	10	140	∞	0	835.26	845.32	1.22	1.20
C3	100	8	200	∞	0	826.14	828.56	1.94	0.29
C4	150	12	200	∞	0	1028.42	1039.77	3.30	1.10
C5	199	17	200	∞	0	1291.45	1317.65	4.34	2.03
C6	50	6	160	200	10	555.43	*555.43*	0.76	*0.00*
C7	75	11	140	160	10	909.68	*909.68*	1.16	*0.00*
C8	100	9	200	230	10	865.94	*865.94*	1.89	*0.00*
C9	150	14	200	200	10	1162.55	1175.06	3.06	1.08
C10	199	18	200	200	10	1395.85	1420.09	4.23	1.74
C11	120	7	200	∞	0	1042.11	1075.06	2.94	3.16
C12	100	10	200	∞	0	819.56	819.97	1.78	0.05
C13	120	11	200	720	50	1541.14	1560.74	2.16	1.27
C14	100	11	200	1040	90	866.37	867.76	1.86	0.16
Average								2.25	0.86

capacities and customer demands (this determines the number of customers on the routes); on the dispersion of the customers (which determines the length of the routes); on the times required for loading, travelling and unloading; and on the maximum daily operation time for the vehicles. In practical situations, w is easily determined by experimentation with the algorithm on various historical test problems. In this paper, we used $w = 5$ for all the random generated test problems for the SDMTPVRP, $w = 3$ for the test problems for the MTVRP taken from the literature and $w = 1$ for all the test problems for the other problems: PVRP, SDVRP and VRP.

Our TS-ABB algorithm does not minimize the number of vehicles in the solution, so it can happen that on any day, two vehicles of the same type, with the same characteristics, are used when the routes assigned to them could be consolidated into only one of the two vehicles. If the opportunity cost of the vehicles must be taken into account (because a vehicle that is not used for distribution can be used in another activity for example) then this opportunity cost should be included in the objective function (Equation (11)) used in our algorithm.

5. Conclusions

In this paper, we presented and discussed the periodic VRP with multiple use of vehicles and accessibility restrictions (SDMTPVRP) that constitutes a generalization of some classical VRPs: the PVRP, the SDVRP and the MTVRP. We proposed for its solution a tabu search heuristic algorithm based on a previously developed algorithm for the PVRP.

Our heuristic has been tested on a set of randomly generated SDMTPVRP test problems. Results were of reasonable quality, as was computation time, even for test problems involving up to 1000 customers. These test problems have been made publicly available for use by future workers.

Our algorithm is also capable of solving other VRPs, specifically: PVRP, SDVRP, MTVRP and VRP. Computational experiments were carried out on test problems taken from the literature with encouraging results:

- For the PVRP our algorithm produced results that in terms of solution quality were better (for all 10 test problems examined) than those produced in a previous paper (Cordeau *et al*, 1997). For six of these test problems, we improved on the best known solution, but at the expense of a longer (albeit reasonable) computation time.
- For the SDVRP our algorithm produced results that in terms of both solution quality and computation time were better than those produced in a previous paper (Cordeau and Laporte, 2001). For nine of 12 test problems, the solution given by our algorithm was better than the solution found in Cordeau and Laporte (2001), and we improved on the best known solution for six test problems.
- For the MTVRP our algorithm with respect to the algorithm of Taillard *et al* (1996):
  - is much better able to find feasible solutions;
  - requires approximately the same computation time;
  - when both algorithms are unable to find a feasible solution, produces solutions that have a slightly higher total route length and cost, but for which the degree of infeasibility associated with the longest route is smaller.
- For the MTVRP our algorithm with respect to the algorithm of Brandão and Mercer (1998):
  - on average produces better quality results, not only with respect to total route length and cost, but also with respect to the degree of infeasibility of the longest route;

  - these better quality results are achieved at the expense of a longer (albeit not unreasonable) computation time.
- For the MTVRP our algorithm with respect to the algorithm of Petch and Salhi (2003):
  - on average produces solutions for which the degree of infeasibility associated with the longest route is smaller;
  - requires approximately the same computation time.
- For the VRP our algorithm produced results with an average percentage deviation from the best known solution of under 1%. However, our algorithm is not competitive with many existing heuristics for the VRP.

Computation times were, in all cases, reasonable.

References

Beasley JE (1990). OR-Library: distributing test problems by electronic mail. *J Opl Res Soc* **41**: 1069–1072.

Berger J and Barkaoui M (2003). A new hybrid genetic algorithm for the capacitated vehicle routing problem. *J Opl Res Soc* **54**: 1254–1262.

Brandão J and Mercer A (1997). A tabu search algorithm for the multi-trip vehicle routing and scheduling problem. *Eur J Opl Res* **100**: 180–191.

Brandão JCS and Mercer A (1998). The multi-trip vehicle routing problem. *J Opl Res Soc* **49**: 799–805.

Chao I, Golden B and Wasil E (1995). An improved heuristic for the period vehicle routing problem. *Networks* **26**: 25–44.

Chao I-M, Golden B and Wasil E (1999). A computational study of a new heuristic for the site-dependent vehicle routing problem. *INFOR* **37**: 319–336.

Christofides N, Mingozzi A and Toth P (1979). The vehicle routing problem. In: Christofides N, Mingozzi A, Toth P and Sandi C (eds). *Combinatorial Optimization*. Wiley: Chichester, pp 315–338.

Cordeau J-F and Laporte G (2001). A tabu search algorithm for the site dependent vehicle routing problem with time windows. *INFOR* **39**: 292–298.

Cordeau J-F, Gendreau M, Hertz A, Laporte G and Sormany J-S (2005). New heuristics for the vehicle routing problem. In: Langevin A and Riopel D (eds). *Logistics Systems: Design and Optimization*. Springer: New York, pp 279–297.

Cordeau J-F, Gendreau M and Laporte G (1997). A tabu search heuristic for periodic and multi-depot vehicle routing problems. *Networks* **30**: 105–119.

Cordeau J-F, Gendreau M, Laporte G, Potvin J-Y and Semet F (2002). A guide to vehicle routing heuristics. *J Opl Res Soc* **53**: 512–522.

Dongarra JJ (2006). Performance of various computers using standard linear equations software. http://www.netlib.org/benchmark/performance.ps.

Gendreau M, Hertz A and Laporte G (1992). New insertion and postoptimization procedures for the traveling salesman problem. *Opns Res* **40**: 1086–1094.

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

Gendreau M, Laporte G and Potvin J-Y (2002). Metaheuristics for the capacitated VRP. In: Toth P and Vigo D (eds). *The Vehicle Routing Problem*. Society for Industrial and Applied Mathematics: Philadelphia, USA, pp 129–154.

Laporte G and Semet F (2002). Classical heuristics for the capacitated VRP. In: Toth P and Vigo D (eds). *The Vehicle Routing Problem*. Society for Industrial and Applied Mathematics: Philadelphia, USA, pp 109–128.

Nag B, Golden BL and Assad A (1988). Vehicle routing with site dependencies. In: Golden BL and Assad A (eds). *Vehicle Routing: Methods and Studies*, Studies in Management Science and Systems, Volume 16. North-Holland: Amsterdam, The Netherlands, pp 149–159.

Petch RJ and Salhi S (2003). A multi-phase constructive heuristic for the vehicle routing problem with multiple trips. *Discrete Appl Math* **133**: 69–92.

Rochat Y and Taillard E (1996). Probabilistic diversification and intensification in local search for vehicle routing. *J Heur* **1**: 147–167.

Taillard ED (1993). Parallel iterative search methods for vehicle routing problems. *Networks* **23**: 661–673.

Taillard ED, Laporte G and Gendreau M (1996). Vehicle routeing with multiple use of vehicles. *J Opl Res Soc* **47**: 1065–1070.

Tarantilis CD, Ioannou G and Prastacos G (2005). Advanced vehicle routing algorithms for complex operations management problems. *J Food Eng* **70**: 455–471.

Wassan NA (2006). A reactive tabu search for the vehicle routing problem. *J Opl Res Soc* **57**: 111–116.

Received March 2006;
accepted January 2007 after two revisions