European Journal of Operational Research 231 (2013) 43–56

Contents lists available at SciVerse ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Discrete Optimization

A tabu search for Time-dependent Multi-zone Multi-trip Vehicle Routing Problem with Time Windows

Phuong Khanh Nguyen [a,b], Teodor Gabriel Crainic [b,c,*], Michel Toulouse [b,d]

[a] Dept. d'informatique et de Recherche Opérationnelle, Université de Montréal, Canada
[b] Centre Interuniversitaire de Recherche sur les Réseaux d'Entreprise, la Logistique et le Transport (CIRRELT), Canada
[c] Dept. de management et technologie, ESG, U.Q.A.M, Canada
[d] Dept. of Computer Science, Oklahoma State University, United States

ARTICLE INFO

Article history:
Received 13 August 2012
Accepted 15 May 2013
Available online 30 May 2013

Keywords:
Multi-trip vehicle Routing with Time Windows
Synchronization
Time-dependent demand
Tabu search

ABSTRACT

We propose a tabu search meta-heuristic for the Time-dependent Multi-zone Multi-trip Vehicle Routing Problem with Time Windows. Two types of neighborhoods, corresponding to the two sets of decisions of the problem, together with a strategy controlling the selection of the neighborhood type for particular phases of the search, provide the means to set up and combine exploration and exploitation capabilities for the search. A diversification strategy, guided by an elite solution set and a frequency-based memory, is also used to drive the search to potentially unexplored good regions and, hopefully, enhance the solution quality. Extensive numerical experiments and comparisons with the literature show that the proposed tabu search yields very high quality solutions, improving those currently published.

1. Introduction

The basic Vehicle Routing Problem with Time Windows (VRPTW) aims to design least cost routes from a single depot to a set of geographically distributed customers, while satisfying time window constraints at customers and the capacity of vehicles. In this paper, we consider the *Time-dependent Multi-zone Multi-trip Vehicle Routing Problem with Time Windows* (*TMZT-VRPTW*), which is an extension of the VRPTW involving both designing and assigning routes to vehicles within time synchronization restrictions.

In the TMZT-VRPTW setting, a homogeneous fleet of vehicles operates out of a single garage to deliver customer-specific loads, available at particular facilities during particular operating time intervals. Deliveries at customers must be performed according to hard time windows. Vehicles must synchronize their arrivals at facilities with the respective operating time periods, that is, time windows at facilities are hard and vehicles are not permitted to arrive in advance and wait. Particular waiting stations may be used by the vehicles to wait for the next appointment. A vehicle route thus leaves the garage to visit a first facility within its operating time periods and load freight, proceeds to deliver it to customers within their time windows, and then moves to its next appointment to a facility, possibly stopping to wait for the appropriate time at a waiting station. The route continues until either there are no more loads to deliver or its cost becomes noncompetitive compared to other routes. The vehicle returns to the garage in both cases. The goal of the TMZT-VRPTW is to determine the set of routes, and assign them to particular vehicles, providing timely customer service and synchronized arrival at facilities for loading freight, minimizing the total cost made up of the (variable) costs of operating vehicles and the (fixed) costs of using them. The "time dependency" characterizing the problem setting follows from the time stamps of the origin-to-destination demand, indicating the time interval when the load is available at the origin facility and the delivery time window at the customer. This is different from most time-dependent vehicle routing problem contributions in the literature where travel costs or travel times are considered to vary with time.

The TMZT-VRPTW is encountered in several settings, in particular in the context of planning the operations of two-tiered City Logistics systems (Crainic et al., 2009). In such systems, the first tier involves large-capacity vehicles delivering freight from the city distribution centers (CDCs) located on the outskirts of the city to intermediary facilities, called satellites, where it is transferred to smaller-capacity vehicles performing the satellite-to-customer delivery routes. Given the concerns regarding the impact of freight transport on the city living conditions (e.g., congestion and environment), as well as the locations of most satellites within or close to the city center, very short waiting times are allowed, most transfer operations being performed according to transdock practices, without intermediate storage.

* Corresponding author at: Dept. de management et technologie, ESG, U.Q.A.M, Canada. Tel.: +1 514 343 7143; fax: +1 514 343 7121.
E-mail addresses: TeodorGabriel.Crainic@cirrelt.ca, doru.crainic@gmail.com (T.G. Crainic).

0377-2217/$ - see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.ejor.2013.05.026

The arrival of first-tier vehicles at a given time period define the set of customers to be serviced, and the time required to unload and transfer the freight thus defining the availability period during which second-tier vehicles must arrive at the satellite and load. Second-tier vehicles must therefore synchronize their arrivals at satellites with these availability periods. After loading the planned freight, each second-tier vehicle undertakes a trip servicing the customers assigned to it. Once the last customer is serviced, the vehicle moves empty either directly to a satellite for its next trip, to a waiting station (when available) to wait for its next appointment at a satellite, or to the garage to end the current work assignment. The TMZT-VRPTW corresponds to the planning of the activities of second-tier vehicles.

To our knowledge, Crainic et al. (2009) were the first (and only) to propose a method to address the TMZT-VRPTW. The authors proposed a decomposition approach that first addressed the VRPTW subproblems representing the delivery to the customers associated with each combination of satellite and availability period. The vehicle trips resulting from the solution of the subproblems were then put together into multi-trip routes by solving a minimum cost network flow problem. These two sets of decisions, (1) how to service customers associated to given facility-availability period combinations and (2) how to combine the resulting trips into vehicle-specific multi-trip routes abiding by the synchronization requirements at facilities, are not independent, however. Combining them into one formulation and solution method should yield better results. The objective of this paper is to take up on this challenge and present a meta-heuristic that addresses the two decisions simultaneously, in a comprehensive and efficient way.

We thus introduce the first tabu search for the TMZT-VRPTW, integrating multiple neighborhoods grouped into two classes to address the two sets of decisions identified above. A first set of neighborhoods and moves work on the construction of the multiple-trip vehicle routes by modifying the facilities and availability periods a given vehicle visits. A second set aims to improve the routing of vehicles between two such visits by working on the customer to route/vehicle assignments. The former perturb significantly the solution and thus favor exploration of the search space, while the latter applied to each vehicle trip exploit good assignments. Hence, dynamically adjusting their utilization during the search provides the proposed algorithm with desired exploration and exploitation capabilities.

The proposed algorithm starts by freely exploring the search space made up of feasible and unfeasible solutions. As the search advances, one lowers the probability of selecting neighborhoods modifying the facility-to-vehicle route assignments, thus limiting the size of the search region and giving routing moves more time to optimize routes. Of course, customer time windows and synchronization requirements constrain these decisions and moves. A diversification strategy guided by an elite set of solutions and a frequency-based memory is called upon when the search begins to stagnate. Creating new working solutions from the elite set helps to capitalize on the best solution attributes obtained so far. On the other hand, employing a frequency-based memory to perturb new working solutions provides a certain level of diversity to the search.

The main contributions of the paper are the following: (1) a new formulation for the TMZT-VRPTW, which is the source to define neighborhoods in the proposed tabu search; (2) the neighborhood structure and the dynamic strategy used to control the selection of neighborhoods; and (3) a new tabu search meta-heuristic outperforming the available method (Crainic et al., 2012) with new best-known solutions on all instances and an improvement in the solution quality by 4.42% on average.

The remainder of the paper is organized as follows. Section 2 contains a detailed problem description. The problem formulation is then provided in Section 3. Section 4 reviews the literature. The details of the proposed methodology are described in Section 5. Computational results are then reported and analyzed in Section 6, while conclusions and future works are considered in Section 7.

2. Problem description

The time-dependency characterizing demand in the TMZT-VRPTW setting translates into two phenomena. The first concerns facilities, which become available for work at particular time periods only with a set of loads destined to specific customers. A given facility may be available at several periods during the planning period considered, with a different set of loads at each occurrence. To model this time dependency, we define ***supply points*** as particular combinations of facilities and availability time periods. A supply point is then characterized by a set of loads to be delivered to particular customers, and by a no-wait hard time window, meaning that vehicles cannot arrive before the beginning of the time window and wait for the opening of the facility, nor after the end of the time window by paying a penalty. The second phenomena concerns customers, which may receive several loads, from different facilities and time periods. We model this time dependency by identifying each particular load as a ***customer demand***, characterized by the supply point where it is available for delivery, the customer it must be delivered to, and the particular time window for the delivery at the customer.

Synchronization at supply points requires that vehicles arrive at supply points at appointed times. Consequently, a direct move that gets the vehicle to a supply point sooner than the appointed time is forbidden. In this case, the vehicle may go to a location, which we call *waiting station* (e.g., a parking lot), and wait there in order to get to its next supply point just before the appointed time. Otherwise, if there is no waiting station available, the vehicle goes to the garage to finish its route.

The TMZT-VRPTW can then be described as follows. There is a garage, or main depot, g, a set of waiting stations $w \in \mathcal{W}$, a set of supply points $s \in \mathcal{S}$, and a set of customer-demand nodes $d \in \mathcal{D}$ (i.e., one node for each customer demand). We assume that there is a limited allowable waiting time, defined by η, at each supply point. Each supply point $s \in \mathcal{S}$ has a no-wait, hard opening time window $[t(s) - \eta, t(s)]$, specifying the earliest and latest times the vehicle may be at s, respectively, a vehicle loading time $\delta(s)$, and a set of customer-demand nodes $D_s \in \mathcal{D}$ making up its *service zone*. Each customer-demand node $d \in D_s$ has a volume q_d to be delivered, a service time $\delta(d)$ to unload freight from the vehicle, and a time window $[e_d, l_d]$, where e_d is the earliest time service may begin and l_d is the latest time. It is assumed all customer-demand volumes are less than the capacity of the vehicle (otherwise, the classical technique of duplicating customer demands such that each conforms to this requirement is applied in a processing phase).

The TMZT-VRPTW can be seen as the problem of determining a set of routes made up of a sequence of supply-point visits, each followed by a trip servicing customer loads in the zone of the respective supply point, and of assigning each route to one vehicle. The objective is to minimize the total cost, which is made up of the fixed cost of using the vehicles and the routing costs of servicing customer demands and moving between supply points, while the following conditions are satisfied:

1. Every vehicle starts and ends its route at the main depot g;
2. Every vehicle servicing customer-demand nodes in D_s must reach the supply point $s \in \mathcal{S}$ within its time window, i.e., it must not arrive sooner than $(t(s) - \eta)$ and no later than

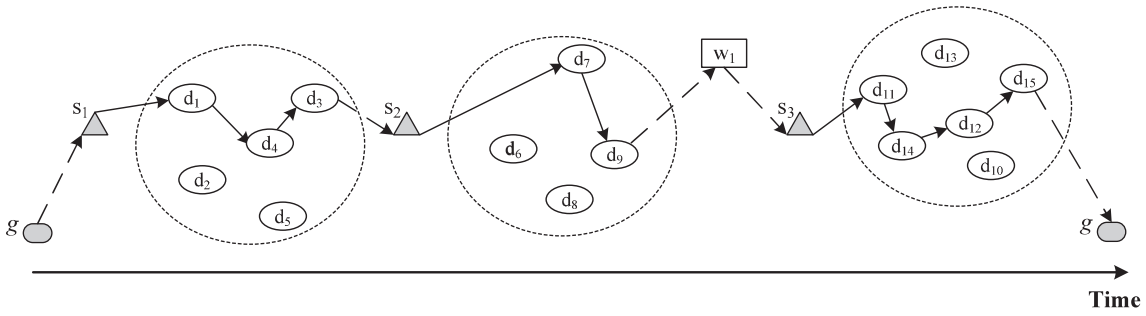

Fig. 1. A three-leg work assignment illustration.

$t(s)$; When needed, the vehicle may wait at a waiting station $w \in \mathcal{W}$ before moving to s; Once at s, the vehicle starts loading at time $t(s)$ and continues loading for a time $\delta(s)$, after which it leaves s to service the assigned customer-demand nodes in D_s. After performing a route within zone D_s, the vehicle may move to another supply point for the next trip or go to the main depot g to complete its route;

3. Every customer-demand node $d \in \cup_{s\in\mathcal{S}} D_s$ is visited by exactly one vehicle within its time window (these are hard).

3. Model formulation

The TMZT-VRPTW is defined on a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{A})$, with vertex set $\mathcal{V} = g \cup \mathcal{S} \cup \mathcal{D} \cup \mathcal{W}$, where g is the main depot, $\mathcal{S}$ is the set of supply points, $\mathcal{D} = \{\cup D_s : s \in \mathcal{S}\}$ is the customer-demand node set, $\mathcal{W}$ is the set of waiting stations, and the arc set $\mathcal{A} = \{(g,s) : s \in \mathcal{S}\} \cup \{(s,d) : s \in \mathcal{S}, d \in D_s\} \cup \{(d,j) : d \in \mathcal{D}, j \in g \cup \mathcal{W}\} \cup \{(i,j) : i,j \in D_s, s \in \mathcal{S}\} \cup \{(d,s') : d \in D_s, s, s' \in \mathcal{S}, t(s) < t(s')\} \cup \{(w,s) : w \in \mathcal{W}, s \in \mathcal{S}\}$. Hence, the set $\mathcal{A}$ does not include arcs representing direct travel:

- From the main depot g to any customer-demand node or waiting station;
- From any customer-demand node to its supply point or to supply points with opening times earlier than that of its supply point;
- From a supply point to any waiting station or to the main depot g.

A routing cost (or travel time) c_{ij} is associated with each arc $(i,j) \in \mathcal{A}$. A fleet of m identical vehicles with capacity Q is based at the main depot g. Vehicles are grouped into set $\mathcal{K}$.

Let a ***route leg*** be a trip that links a pair of supply points, or starts and ends at a supply point and the main depot g, respectively. Thus, there are two types of route legs and their feasibility is defined as follows:

- A ***single-supply point route leg*** l, starting at supply point s and ending at the main depot, is feasible if it starts loading a total of goods not exceeding Q at supply point s at time $t(s)$, then leaves s at time $t(s) + \delta(s)$ to deliver to a subset of customer-demand nodes in D_s within their time windows.
- An ***inter-supply point route leg*** l that starts and ends at a pair of supply points s and s', respectively, is feasible if it starts loading a total of goods not exceeding Q at supply point s at time $t(s)$, then leaves s at time $t(s) + \delta(s)$ to deliver to a subset of customer-demand nodes in D_s within their time windows, and arrives to s' within the opening time window $[t(s') - \eta, t(s')]$ (the vehicle can wait at a waiting station $w \in \mathcal{W}$ before moving to s' in case the direct move from the last serviced customer in leg l to s' gets the vehicle to s' before $(t(s') - \eta)$).

A sequence of route legs, starting and ending at the main depot, assigned to a vehicle is called a route or ***work assignment***. For the sake of simplicity, from now on, the terms vehicle route and work assignment are used interchangeably. Fig. 1 illustrates a three-leg work assignment, where s_1, s_2, s_3 are supply points, g and w_1 are the main depot and a waiting station, respectively, $D_{s_1} = \{d_1, d_2, d_3, d_4, d_5\}, D_{s_2} = \{d_6, d_7, d_8, d_9\}$, and $D_{s_3} = \{d_{10}, d_{11}, d_{12}, d_{13}, d_{14}, d_{15}\}$. The dashed lines stand for the empty arrival from the depot g or from a waiting station, the empty movement from the last customer in the previous leg to the supply point of the next leg or to a waiting station, and the empty movement to the depot g once the work assignment is finished. This work assignment consists of a sequence of three legs $\{l_1, l_2, l_3\}$, where $l_1 = \{s_1, d_1, d_4, d_3, s_2\}$ and $l_2 = \{s_2, d_7, d_9, w_1, s_3\}$ are two inter-supply point route legs, while $l_3 = \{s_3, d_{11}, d_{14}, d_{12}, d_{15}, g\}$ is a single-supply point route leg.

Let $\mathcal{L}$ denote the set of all feasible legs satisfying the total load of vehicles, and the time windows at customers and supply points. Define the e_{dl} and f_{ls} coefficients:

$$e_{dl} = \begin{cases} 1 & \text{if customer demand } d \in D \text{ is on leg } l \in \mathcal{L}; \\ 0 & \text{otherwise;} \end{cases}$$

$$f_{ls} = \begin{cases} 1 & \text{if leg } l \text{ starts at supply point } s \in \mathcal{S}; \\ -1 & \text{if leg ends at supply point } s \in \mathcal{S}; \\ 0 & \text{otherwise;} \end{cases}$$

Binary decision variables are used in the formulation:

- $x_l^k = \begin{cases} 1 & \text{if leg } l \in \mathcal{L} \text{ is assigned to work assignment } k \in \mathcal{K}, \\ 0 & \text{otherwise} \end{cases}$
- $y_s^k = \begin{cases} 1 & \text{if work assignment } k \text{ has the first leg starting at supply point } s, \\ 0 & \text{otherwise} \end{cases}$
- $z_s^k = \begin{cases} 1 & \text{if work assignment } k \text{ has the last leg starting at supply point } s, \\ 0 & \text{otherwise} \end{cases}$

Let π_l be the total cost of route leg l, and F be the fixed cost of using a vehicle. The TMZT-VRPTW can then be formulated as

$$\text{Minimize} \sum_{k\in\mathcal{K}} \sum_{l\in\mathcal{L}} \pi_l x_l^k + \sum_{k\in\mathcal{K}} F \sum_{s\in\mathcal{S}} y_s^k \tag{1}$$

$$\text{S.t.} \sum_{k\in\mathcal{K}} \sum_{l\in\mathcal{L}} e_{dl} x_l^k = 1 \quad \forall d \in \mathcal{D}, \tag{2}$$

$$\sum_{s\in\mathcal{S}} y_s^k \leqslant 1 \quad \forall k \in \mathcal{K}, \tag{3}$$

$$\sum_{s\in\mathcal{S}} y_s^k = \sum_{s\in\mathcal{S}} z_s^k \quad \forall k \in \mathcal{K}, \tag{4}$$

$$\sum_{l\in\mathcal{L}} f_{ls}x_l^k = y_s^k \quad \forall s \in \mathcal{S},\ k \in \mathcal{K}, \tag{5}$$

$$x_l^k \in \{0,1\} \quad \forall l \in \mathcal{L},\ k \in \mathcal{K}, \tag{6}$$

$$y_s^k \in \{0,1\} \quad \forall s \in \mathcal{S},\ k \in \mathcal{K}, \tag{7}$$

$$z_s^k \in \{0,1\} \quad \forall s \in \mathcal{S},\ k \in \mathcal{K} \tag{8}$$

The objective function (1) minimizes the total cost made up of the costs of operating and using vehicles. Constraints (2) guarantee that each customer demand is visited exactly once, while Constraints (3) state that at most one work assignment is assigned to each vehicle. Constraints (4) ensure that each work assignment starts and ends at the main depot. In fact, by summing over all supply points, the left-hand side counts the number of first legs assigned to each vehicle k, while the right-hand side counts the number of last legs assigned to each vehicle k. Then, from Constraints (3) and (4), either the numbers of first and last legs assigned to the vehicle k are both equal to zero, e.g., the vehicle k is not used $(\sum_{s\in\mathcal{S}} y_s^k = \sum_{s\in\mathcal{S}} z_s^k = 0)$, or are both equal to 1, e.g., the vehicle k is used $(\sum_{s\in\mathcal{S}} y_s^k = \sum_{s\in\mathcal{S}} z_s^k = 1)$.

Constraints (5) ensure that when a vehicle goes to a supply point, it also leaves it, except for the starting supply point of the first leg. In fact, for any given vehicle k and supply point s, the left-hand side of the equality sums the value of f_{ls} on all legs starting or ending at supply point s which are assigned to vehicle k. Consequently, when $y_s^k = 1$, the equality (5) becomes $\sum_{l\in\mathcal{L}} f_{ls}x_l^k = 1$, which means that there must be one leg starting at supply point s assigned to vehicle k as the first leg. Constraints (6)–(8) define the sets of decision variables.

4. Literature review

The literature on TMZT-VRPTW is limited. In the TMZT-VRPTW setting, customer demands are divided into zones associated to supply points, that is facilities and time periods. Taking advantage of this special structure, Crainic et al. (2009) proposed a decomposition-based heuristic approach for TMZT-VRPTW, but no implementation was reported. The general idea is to decompose the problem by (facility, period) zone, solve the resulting small VRPTW at each zone, and finally determine the flow of vehicles to operate the routes associated with these zones at minimum cost by solving a minimum cost network flow problem. Crainic et al. (2012) later implemented this idea, and calculated a lower bound by relaxing vehicle capacity and time window constraints at supply points and customers.

A number of VRP variants share the multi-trip setting with the TMZT-VRPTW, e.g., the Multi-trip Vehicle Routing Problem (or the Vehicle Routing Problem with Multiple Use of Vehicles; Taillard et al., 1995; Brandão and Mercer, 1998; Petch and Salhi, 2003; Salhi and Petch, 2007), the VRP with Intermediate Facilities or with inter-depot routes (Tarantilis et al., 2008; Crevier et al., 2007), and the Waste Collection VRP (Kim et al., 2006; Ombuki-Berman et al., 2007; Benjamin and Beasley, 2010). In the first variant, only one depot is used to replenish vehicles between their trips, while in the latter variants, vehicles may be replenished at intermediate depots along their trips. In addition, unlike the first two variants, each driver is assumed to take a lunch break in a period of time in the Waste Collection VRP. The challenging setting in our problem compared to these three variants is the time synchronization restrictions at supply points and the waiting stations.

The School Bus Routing problem (SBRP) resembles our problem setting quite closely. In general, the SBRP involves transporting students from predefined locations to their schools using a fleet of buses with varying capacities, while satisfying all school timing requirements (see the surveys of Desrosiers et al., 1981; Braca et al., 1997; Park and Kim, 2010). The SBRP consists of three components: determine the bus stop locations, assign students to bus stops, route and schedule the buses. However, most of the problems described in the literature just consider some parts of the SBRP. In the multi-school setting, the SBRP shares some constraint settings with the TMZT-VRPTW, e.g., vehicle capacity, school time window, multi-trip. Yet, there are also differences in conditions and settings between the SBRP and the TMZT-VRPTW. Mixed-load settings may occur in the SBRP, where students from different schools can be put on the same bus at the same time; Maximum riding times for students on buses may also be specified. These settings are not imposed in the TMZT-VRPTW. In the SBRP, only the exact earliest pick-up time for all students is considered, while there is a time window for each customer in the TMZT-VRPTW.

5. Tabu search meta-heuristic

Among the meta-heuristics proposed for the vehicle routing problem, tabu search has been shown to be a very effective one, providing a good compromise between solution quality and computation time. Various techniques have also been proposed to further enhance the performance of tabu search for complex problems with multiple constraints and characteristics. Inspired in part by these developments, we propose a tabu search with multiple neighborhoods and appropriate memory mechanisms for the TMZT-VRPTW. This section describes our tabu search algorithm, from its general structure to detailing its main components. The search space and initial solution generator are presented in Sections 5.2 and 5.3, respectively. Sections 5.4, 5.5, 5.6 describe the neighborhood structures, our neighborhood-selection strategy, and the tabu status mechanism for each neighborhood. We detail in Section 5.7 the management of the elite set and a diversification mechanism. Finally, a post optimization procedure is described in Section 5.8.

5.1. General structure

The TMZT-VRPTW schedules vehicles from the main depot to supply points in order to load freight. The vehicles then deliver freight from supply points to customers. Solutions to this problem involve two types of decisions: the first ones assign vehicles to supply points while the second ones assign customer demands to vehicles. We define neighborhood structures for each of these two types of decisions. The *leg neighborhoods* aim to change the vehicle assignments to supply points, while *routing neighborhoods* move customer demands among vehicle routes.

Several studies of tabu search with multiple neighborhoods exist. In some studies (e.g., Dell'Amico and Trubian, 1993; Gaspero and Schaerf, 2007), all neighborhoods are evaluated simultaneously. In other ones (e.g., Xu et al., 2006; Hamiez et al., 2009), neighborhoods are explored in a serial way, one after another in either a fixed or randomized order. Our experiments show that these approaches do not work well for the TMZT-VRPTW (see details in Annex A). This is because, in our problem, each type of neighborhood is applied to a particular decision domain, whereas size is the main difference between neighborhoods in the literature on tabu search with multiple neighborhoods. Moreover, the two decision domains of our problem are related and impact each other. Therefore, we propose an approach where only one neighborhood type is used in a given iteration of the tabu search method. Our selection strategy of the neighborhood type at each iteration is probabilistic, the distribution of our probability function being biased by the structure of our problem and the state of the search in order to obtain the right balance of exploration within each neighborhood space. The bias of our function is implemented through a parameter r that specifies *the ratio of selecting routing neighborhoods to leg neighborhoods*. The value of this

parameter is adjusted during the course of the algorithm to favor the best possible exploration of the problem solution space.

The general structure of the tabu search meta-heuristic (TS) we propose is introduced in Algorithm 1. First, an initial feasible solution p is generated using a greedy method seeking to fully utilize vehicles and minimize the total cost. At each iteration of the tabu search method, one neighborhood is selected probabilistically based on the current value of r, then the selected neighborhood is explored, and the best move is chosen (lines 7–8). This move must not be tabu, unless it improves the current best solution p_{best} (aspiration criterion). The algorithm adds the new solution to an elite set $\mathcal{E}$ if it improves on p_{best}. It also remembers the value of the parameter r when the new best solution was found (lines 9–13), and finally updates the elite set $\mathcal{E}$ by removing a solution based on its value and the difference between solutions (Section 5.7).

Algorithm 1. Tabu search

```
1: Generate an initial feasible solution p
2: p_best ← p
3: Elite set E ← ø
4: Probability of selecting routing neighborhood with
   respect to leg neighborhood r ← 1
5: STOP ← 0
6: repeat
7:    A neighborhood is selected based on the value of r
8:    Find the best solution p' in the selected neighborhood
   of p
9:    if p' is better than p_best then
10:      p_best ← p'
11:      r_best ← r
12:      Add (p_best,r_best) to the elite set E; update E
13:   end if
14:   p ← p'
15:   if p_best not improved for IT_cNS iterations then
16:      if p_best not improved after C_cNS consecutive
   executions of Control procedure then
17:         if E = ø then
18:            STOP ← 1
19:         else
20:            Select randomly (p,r_p) (and remove it) from the
   elite set E
21:            Diversify the current solution p
22:            Set r ← r_p and reset tabu lists
23:         end if
24:      else
25:            Apply Control procedure to update the value of
   r
26:            p ← p_best
27:         end if
28:      end if
29: until STOP
30: p_best← Post-optimization(p_best)
31: return p_best
```

Initially, the search freely explores the solution space by selecting each neighborhood with equal probability. Whenever the best solution is not improved for IT_{cNS} TS iterations (line 25), the *Control* procedure is called to reduce the probability of selecting leg neighborhoods. Consequently, route neighborhoods are selected proportionally more often, which gives routing moves more opportunity to optimize trips. The search is re-initialized from the current best solution p_{best} after the execution of the *Control* procedure (line 26). Moreover, after C_{cNS} consecutive executions of this procedure without improvement of the current best solution p_{best}, a solution p is selected randomly and removed from the elite set $\mathcal{E}$ (line 20), and a *Diversification* mechanism is applied to perturb p (line 21). The value of r is reset to the value it had when the corresponding elite solution was found, and all tabu lists are reset to the empty state (line 22). The search then proceeds from the perturbed solution p. The search is stopped when the elite set $\mathcal{E}$ is empty. Finally, a post-optimization procedure is performed to potentially improve the current best solution p_{best} (line 30).

5.2. Search space

As described in Section 3, a solution is a set of work assignments, each work assignment consisting of a sequence of route legs linking supply points. The search space is thus made up of feasible and unfeasible work assignments.

For a given solution p, let $c(p)$ denote the total travel cost of its work assignments, and let $q(p), w_c(p)$, and $w_s(p)$ denote the total violation of vehicle load, customer time windows, and supply-point time windows, respectively. The total vehicle-load violation is computed on a route leg basis with respect to the value Q, whereas the total violation of time windows of customers is equal to $\sum_{d\in p}\max\{(a_d - l_d), 0\}$, and the total violation of time windows of supply points is equal to $\sum_{s\in p}\max\{(t(s) - \eta - a_s), (a_s - t(s)), 0\}$, where a_d and a_s are the arrival time at customer demand d and supply point s, respectively.

Due to the time synchronization restrictions at supply points, the arrival time at the first customer d of each leg l starting at supply point s is always calculated as $a_d = t(s) + \delta(s) + c_{sd}$, no matter when the vehicle arrives at supply point s. This helps to prevent the propagating time-window unfeasibility among the legs of a work assignment.

Solutions are then evaluated according to the weighted fitness function $f(p) = c(p) + \alpha_1 q(p) + \alpha_2 w_c(p) + \alpha_3 w_s(p) + F*m$, where α_1, α_2, α_3 are penalty parameters adjusted dynamically during the search. The updating scheme is based on the idea of Cordeau et al. (2011). At each iteration, the value of α_1, α_2, and α_3 are modified by a factor $1 + \beta > 1$. If the current solution is feasible with respect to load constraints, the value of α_1 is divided by $1 + \beta$; otherwise it is multiplied by $1 + \beta$. The same rule applies to α_2 and α_3 with respect to time window constraints of customers and supply points, respectively. In our algorithm, we set $\alpha_1 = \alpha_2 = \alpha_3 = 1$ and $\beta = 0.5$.

5.3. Initial solution

We sort the supply points (customer zones) and index them in increasing order of their opening time. Thus, if $t(s_1) \leqslant t(s_2)$, then $s_1 < s_2$ and vice versa. We then construct an initial solution by building each work assignment sequentially. Each work-assignment construction consists of two phases: the first phase determines the first supply point for the current work assignment; the second phase creates sequentially each leg using a greedy algorithm.

In the first phase, the supply point s with earliest opening time and unserviced customers is assigned as the initial supply point of the first leg of the current work assignment. During the second phase, the first leg l is created using a greedy algorithm. If the leg l ends at a supply point s', we continue applying the greedy algorithm to build the next leg of l in which s' is now used as the initial supply point. Otherwise, if the leg l ends at the main depot, it means the current work assignment cannot be used anymore, and we return to the first phase to build another work assignment. This process is repeated until all customers are serviced (assigned to a vehicle route).

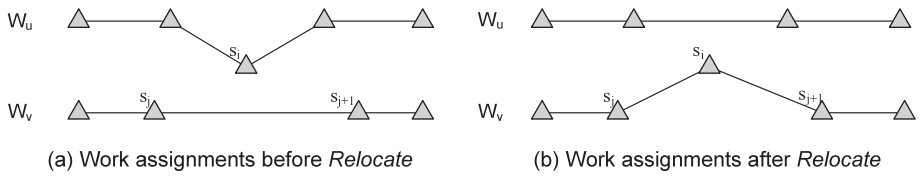

Fig. 2. Relocate supply point.

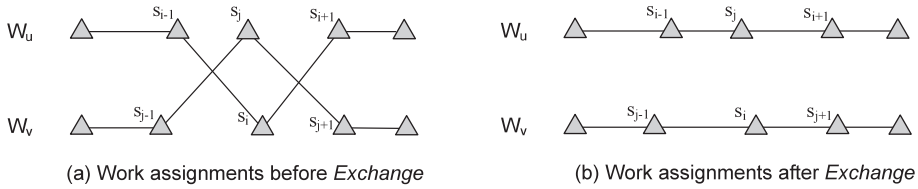

Fig. 3. Exchange supply points.

The greedy algorithm constructs each leg by attempting to minimize the cost and keep the vehicle working at full capacity as much as possible. Thus, for a given initial supply point s assigned to the leg, it finds a set of supply points $S' = \{s' \in \mathcal{S} | s'$ with unserviced customers and $t(s') > t(s)\}$. If $S' \neq \emptyset$, for each pair (s,s'), it creates an empty leg with s and s' as the initial and end supply point, respectively. It then assigns unserviced customers of customer zone s to this leg sequentially by applying the heuristic I1 of Solomon (1987) until the vehicle is full. When feasible legs exist, the one with minimum cost is selected. In the case there are no feasible legs or $S' = \emptyset$, it builds the last leg (s,g) by applying the heuristic I1 of Solomon (1987).

5.4. Neighborhoods

Leg neighborhoods focus on repositioning legs at supply points within the time restrictions. Let W_u be the work assignment assigned to vehicle u. Let s_{i-1} and s_{i+1} denote the predecessor and successor supply points, respectively, of s_i within a work assignment. The leg-move operators are:

- *Relocate supply point*. Consider two work assignments W_u and W_v as illustrated in Fig. 2. For supply point $s_i \in W_u$, such that $s_i \notin W_v$, and for each two successive supply points s_j, $s_{j+1} \in W_v$, if $s_j < s_i < s_{j+1}$ then move supply point s_i from work assignment W_u to W_v locating it between s_j and s_{j+1}. All customers serviced by s_i on W_u are also moved to W_v.
- *Exchange supply points*. Consider two work assignments W_u and W_v as illustrated in Fig. 3. For supply points $s_i \in W_u$ and $s_j \in W_v$ such that $s_{i-1} < s_j < s_{i+1}$ and $s_{j-1} < s_i < s_{j+1}$, swap s_i and s_j together with their customers.

When moving a supply point, all customers serviced by it are also moved. Therefore, the violations of load and time windows for customers are not changed by the move. The *move value* is thus defined as $\Delta f = \Delta c + F*\Delta m + \Delta w_s$. The three components of the summation are the difference in travel cost, the fixed cost of using vehicles, and the difference in violation of time windows at supply points between the value of the neighboring solution and the value of the current solution.

Routing neighborhoods try to improve trip routing by using different intra- and inter-route neighborhoods commonly used in the VRPTW literature: Relocation, Exchange and 2-opt. For each move in each neighborhood, two customers are considered.

- *Relocation move*: one of the two customers is taken from its current position and inserted after the other one.
- *Exchange move*: two customers are swapped.
- *2-opt move*: for two customers in the same leg, the edges emanating from them are removed, two edges are added, one of which connects these two customers, and the other connects their successor customers. For two customers in different legs, the work assignment segments following them are swapped preserving the order of customers succeeding them in each segment.

Moving customers could change the travel cost and the number of vehicles, as well as the level of constraint violations of load, time windows of customers, and time windows of supply points. Consequently, the value of a routing move is defined as $\Delta f = \Delta c + F* \Delta m + \Delta q + \Delta w_c + \Delta w_s$. Note that for Δc, the change in the routing cost, may involve, beside a change in the routing cost between customers, a change in the routing cost from the last customer to the supply point at the end of the modified leg(s). For example, a routing move may impact on whether a vehicle has to go to a waiting station or not, therefore impacting the traveling cost from the last customer to its supply point.

5.5. Neighborhood selection strategy

The algorithm explores one neighborhood at each iteration. The neighborhood to explore is randomly selected among the five previously defined neighborhoods. The main issue in this case is to define a probability distribution for the neighborhoods as well as whether the probability distribution should evolve over time.

Using a fixed a priori distribution has drawbacks which limit the exploration capability of the algorithm. For example, this would mean that the algorithm will display the same behavior during the entire search. Moreover, the calibration of the probabilities would be extremely challenging and instance dependent. Indeed, too low routing-neighborhood probabilities (high leg-neighborhood probabilities) would result in an insufficient number of routing moves to adequately optimize the customer routes after the leg moves. The search may easily get stuck in the opposite case, as it needs to move to less-explored regions of the search space once a succession of routing moves have "optimized" trips.

Considering the drawbacks of having a fixed probability distribution for the selection of neighborhoods, we have elected to vary these probabilities as the search progresses. At the beginning of the search, both leg and routing neighborhoods are given the same

probability of been selected, which allows the TS algorithm to freely explore the solution space. Given that the number of supply points is much smaller than the number of customers in most TMZT-VRPTW instances, the algorithm should perform more routing than leg moves to ensure adequate optimization of routes. Consequently, after the initial phase, the probability of selecting leg neighborhoods becomes lower than the probability of selecting routing neighborhoods. We control this probability distribution using the neighborhood-selection parameter r, assigning to a routing neighborhood the probability $r/(2+3r)$ of been selected, and to a leg neighborhood the probability $1/(2+3r)$ of been selected. The equal initial probabilities are then obtained by setting $r = 1$. The *Control* procedure in our algorithm varies the value of r during execution to monotonically reduce (increase) the probability of selecting leg (routing) neighborhoods after each IT_{cNS} iterations without improvement of the best solution. A linear scheme $r_{k+1} = r_k + \Delta_r$ is used, where Δ_r is a user defined parameter.

5.6. Tabu lists and tabu duration

We keep a separate tabu list for each type of move. Elements of a solution generated by a move are given a tabu status as follows:

- *Leg moves*:
  - *Relocate supply point:* the position of supply point s_i just inserted into work assignment W_v cannot be changed by another relocate supply point move while it is tabu.
  - *Exchange supply points:* supply points s_i and s_j just swapped cannot be swapped again while they are tabu.
- *Routing moves:*
  - *Relocation move:* the position of customer i just inserted after customer j, cannot be changed by the same type of move while it is tabu.
  - *Exchange move:* customers i and j just swapped cannot be swapped again while they are tabu.
  - *2-Opt move*: a 2-opt move applied to customers i and j cannot be applied again to the same customers while tabu.

A tabu status is assigned to each tabu list element for θ iterations, where θ is randomly selected from a uniform interval. Generally, the tabu status of a move stays so for a number of iterations proportional to the number of possible moves. Consequently, we use different intervals of tabu list size for leg and routing moves. Since there are $O(m'^*|\mathcal{S}|)$ possible leg moves, we set the interval of tabu list size for leg moves to $[m'^*|\mathcal{S}|/a_1, m'^*|\mathcal{S}|/a_2]$, where m' is the number of vehicles used in the initial solution, and a_1 and a_2 are user-defined parameters.

In TMZT-VRPTW, each supply point has its own customer demands. Therefore, the number of iterations during which a routing move within the zone of a supply point s remains tabu is only counted each time the algorithm deals with customer demands in that zone. The interval of tabu list size for routing moves for each supply point s with $|D_s|$ associated customer demands is therefore calculated as $[a_3\log_{10}(|D_s|), a_4\log_{10}(|D_s|)]$, where a_3 and a_4 are user defined parameters.

In our tabu search, a move declared tabu is accepted if it improves the current best solution.

5.7. Diversification strategy

A diversification strategy, based on an elite set and a frequency-based memory, moves the search to potentially unexplored promising regions when it begins to stagnate. In a nutshell, diversification aims to capitalize on the best attributes obtained so far by selecting a new working solution from the elite set and perturbing it based on long-term trends.

In more details, we use the elite set as a diversified pool of high-quality solutions found during the tabu search. The elite set starts empty and is limited in size. The quality and diversity of the elite set is controlled by the insertion of new best solutions produced by the tabu search and the elimination of the existing solutions in the elite set. The elimination is based on the Hamming distance $\Delta(p_1, p_2)$ measuring the number of customer positions that differ between solutions p_1 and p_2. (see, e.g., Ehmke et al., 2012; Vidal et al., 2012, for the utilization of the Hamming distance in other VRP settings).

The elimination of a solution from the elite set is considered each time a new best solution p_{best} is inserted. There are two cases. While the elite set is not yet full, we delete only when there exists a solution very similar to the new p_{best}, i.e., we delete the solution p with the smallest $\Delta(p, p_{best}) \leqslant 0.05(n + |\mathcal{S}|)$. This aims to balance the impact on pool quality and diversity. When the elite set is full, p_{best} replaces the solution p that is the most similar to it, i.e., the one with the smallest $\Delta(p, p_{best})$.

The long-term frequency memory keeps a history of the arcs most frequently added to the current solution. Let t_{ij} be the number of times arc (i,j) has been added to the solution during the search process. The frequency of arc (i,j) is then defined as $\rho_{ij} = t_{ij}/T$, where T is the total number of iterations executed so far.

Diversification then proceeds to perturb the search that starts from the solution taken from the elite set by removing arcs with high frequency and inserting arcs with low frequency. Thus, the evaluation of neighbor solutions is biased so as to penalize the arcs most frequently added to the current solution. More precisely, a penalty $g(\bar{p}) = \overline{C}(\sum_{(i,j)\in A_a}\rho_{ij} + \sum_{(i',j')\in A_r}(1-\rho_{i'j'}))$ is added to the evaluation of the fitness $f(\bar{p})$ (Section 5.2) of a neighbor $\bar{p}$ of the current solution p, where $\overline{C}$ is the average cost of all arcs in the problem, and A_a and A_r are the sets of arcs that are added to and removed from the solution p in the move to $\bar{p}$, respectively. The diversification mechanism is executed IT_{div} iterations.

5.8. Post-optimization

The best solution obtained through the tabu search is enhanced by applying a number of well-known local search route improvement techniques. Two are intra-route operators, the 2-opt of Lin (1965) and the Or-opt of Or (1976). The others are inter-route operators, the λ-interchange of Osman (1993), and the CROSS-exchange of Taillard et al. (1997). For the λ-interchange, we only consider the cases where $\lambda = 1$ and $\lambda = 2$ corresponding to the (1,0), (1,1), (2,0), (2,1), and (2,2)-interchange operators. A customer is re-allocated only to legs with the same initial supply point. The post-optimization procedure is executed for each customer zone separately.

The post-optimization procedure starts by applying in random order the five λ-interchange and CROSS-exchange inter-route operators. Each neighborhood is searched on all possible pairs of legs (in random order) of the same starting supply point and stopped on the first improvement. The solution is then modified and the process is repeated until no further improvement can be found. The search is then continued by locally improving each leg of the current starting supply point in turn. The intra-route 2-opt and Or-opt neighborhoods are sequentially and repeatedly applied until no more improvement is found.

6. Computational results

The objective of the numerical experimentation is threefold. First, to study the impact of a number of major parameters and search strategies on the performance of the proposed algorithm in order to identify the most efficient ones. The second objective con-

sists in evaluating the performance of the method through comparisons with currently published results. We finally analyze the impact of synchronization and vehicle fixed cost on solution quality.

Our tabu search algorithm is implemented in C++. Experiments were run on a 2.8 GHz Intel Xeon 4-core processor with 16 GB of RAM. Six sets of ten instances each, generated by Crainic et al. (2012), were used throughout the experiments. These Euclidean instances are identified as A1, A2, B1, B2, C1, and C2. The numbers of customer zones for these sets are 4, 8, 16, 32, 36, and 72, respectively. The numbers of customer demands are 400, 1600, and 3600 for set of type A, B, and C, respectively. Supply points, waiting stations, and customers are uniformly distributed in a square, with the X and Y coordinates in the interval [0,100], [0,200], and [0,300] for set of type A, B, and C, respectively. The opening times of supply points are generated randomly in the [0,14,400] range, while the limited allowable waiting time at supply points $\eta = 100$. The vehicle-loading times at supply points are set to 30, for all supply points. The time window of each customer demand is then generated based on the opening time of the supply point to which it is assigned. The ready time and time window duration of each customer demand are generated randomly in the interval [0,300] and [150,450], respectively. The due time for delivery is thus set by adding the time window duration to the ready time. To ensure feasibility of movements from a supply point to its customer demands, a number of values are added to the ready times and due times: the opening time of the supply point to which the customer demand is assigned, the loading time at the supply point, and the smallest integer higher than the value of the distance between the customer demand and the supply point. One waiting station is set for every 100 customers in all instances. The fixed cost and the capacity of each vehicle are set to 500 and 100, respectively, for all instance sets.

6.1. Algorithm design and calibration

We aim for a general algorithmic structure avoiding instance-related parameter settings. We therefore defined settings as functions of problem size for the main parameters of the proposed algorithm, tabu tenure, neighborhood selection-control, diversification triggering, and size of the elite set.

6.1.1. Tabu tenure calibration

The intervals for the tabu list tenures for leg and routing moves were defined in Section 5.6 as $[m'^*|\mathcal{S}|/sa_1, m'^*|\mathcal{S}|/a_2]$ and $[a_3\log_{10}(|D_s|), a_4\log_{10}(|D_s|)]$, respectively. Using a large interval for routing moves,[10,20], we tested different values for a_1 in the integer interval [7,9] and for a_2 in the integer interval [4,6]. We observed that too large an interval is not productive as low values cannot prevent cycling, while high ones overly restrict the search path. We therefore set a_1 and a_2 to 7 and 5, respectively.

A similar process has been used to explore different values of a_3 in the integer interval [6,8] and a_4 in the integer interval [10,12] using the leg-move tabu tag interval as defined above. We found that the most appropriate values for a_3 and a_4 are 7 and 10, respectively.

6.1.2. Calibration of the neighborhood selection probabilities

Adjustments to the neighborhood selection probabilities depend on two parameters: IT_{cNS}, the number of consecutive iterations without improvement of the best solution (this number triggers the execution of the *Control* procedure that modifies probabilities), and Δ_r, the adjustment factor of the neighborhood-selection parameter r.

The value of IT_{cNS} is defined as a function of the problem size. This value should be large enough to give each customer demand and supply point in each leg the possibility to be moved. Thus, $IT_{cNS} = e_1 * (m'^*|\mathcal{S}| + n)$, where m' is the number of vehicles used in the initial solution, $|\mathcal{S}|$ and n are the numbers of supply points and customer demands, respectively, and e_1 is a user defined parameter. Similarly, Δ_r, the amplitude of the modifications in the probabilities, is set to be proportional to the ratio of the number of customers with the number of supply points. Thus, $\Delta_r = e_2 \cdot \log_{10}(n/|S|)$, where e_2 is a user defined parameter.

Searching for a good combination of values for e_1 and e_2 concerns balancing between exploration and exploitation. On one hand, the higher the value of IT_{cNS}, the more chances customer demands and supply points are to be moved between routes, thus favoring exploration. On the other hand, a too high IT_{cNS} value may waste time in useless moves. We experimented with different values of e_1 in the integer interval [1,5] and e_2 in the integer interval [1,7]. Three runs were performed for each instance for 1 million iterations. Computational results for each combination of values (e_1, e_2) over all 60 instances are summed up in Table 1, which displays the average gaps to the previous best known solutions (BKS) of the solutions obtained by each combination.

Table 1 indicates that (3,5) is the most appropriate combination for (e_1,e_2), improving the solution quality by 3.44% on average. We also observed that executing the algorithm with r greater than $50\log_{10}(n/|S|)$ yields an average improvement of the best solution of less than 0.1%, while requiring about 37.3% more time. Based on these results, we used $(e_1,e_2) = (3,5)$ and $r_{\max} = 50\log_{10}(n/|S|)$, the maximum value of r, in the remaining experiments.

6.1.3. Neighborhood search strategy

The neighborhood exploration strategy specifies how the different neighborhoods can be used to explore efficiently the solution space of the problem. Several such strategies can be envisioned and we actually experimented with quite a number of them before selecting the one introduced in Section 5.5. For concision's sake, we placed in Annex A the description of the alternate strategies we explored and the numerical results supporting our selection.

The neighborhood-search strategy also specifies which move in the neighborhood is to be chosen at each iteration. We studied two strategies, first and best improvement, respectively. The former chooses the first neighbor solution that improves the objective function as the next starting solution, while the latter chooses the best neighbor thus requiring to evaluate all neighbors. The customers in each route are searched sequentially.

Table 2 reports comparison results between these two strategies. Corresponding average gaps to the previous BKS and average computation times are displayed in Columns *GAP to BKS* and *Time (min)*, respectively.

The computational results in Table 2 show that using the same elite set size (=5), best improvement gives better solutions for all sets, while first improvement has a lower computation time. One observes, however, that the difference in computation time is smaller than the difference in solution quality, indicating that the best improvement strategy yields a better algorithm. More importantly, even doubling size of the elite set (=10), which results in

Table 1
Performance comparison between (e_1,e_2) combinations.

e_1	e_2						
	1 (%)	2 (%)	3 (%)	4 (%)	5 (%)	6 (%)	7 (%)
1	−2.21	−2.42	−2.98	−3.09	−3.11	−3.18	−3.15
2	−2.27	−2.45	−3.24	−3.21	−3.17	−3.14	−3.12
3	−2.34	−2.73	−3.37	−3.39	−3.44	−3.36	−3.28
4	−2.46	−2.74	−3.31	−3.36	−3.41	−3.28	−3.24
5	−2.41	−2.78	−3.32	−3.37	−3.39	−3.29	−3.19

Table 2
Comparative performances between neighborhood search strategies.

Problem set	Best improvement		First improvement			
	Elite set size = 5		Elite set size = 5		Elite set size = 10	
	GAP to BKS (%)	Time (min)	GAP to BKS (%)	Time (min)	GAP to BKS (%)	Time (min)
A1	−3.01	18	−1.95	16	−2.06	24
A2	−6.22	10	−4.42	8	−4.57	14
B1	−4.47	60	−2.82	45	−2.98	69
B2	−4.71	39	−3.23	27	−3.35	46
C1	−3.78	165	−1.14	115	−1.24	176
C2	−3.82	104	−1.89	80	−1.91	108
Average	−4.33	66	−2.57	49	−2.68	73

Table 3
Performance comparison between diversification settings.

Elite set size	Without diversification				With diversification			
	1st variant		2nd variant		3rd variant		4th variant	
	$r = r_p$		$r = r_p/2$		$r = r_p$		$r = r_p/2$	
	GAP to BKS (%)	Time	GAP to BKS (%)	Time	GAP to BKS (%)	Time	GAP to BKS (%)	Time
0	−2.96	24	–	–	–	–	–	–
1	−3.18	29	−3.03	38	−3.93	35	−3.97	47
5	−3.39	39	−3.34	47	−4.33	66	−4.24	71
10	−3.53	50	−3.61	59	−4.35	92	−4.25	98

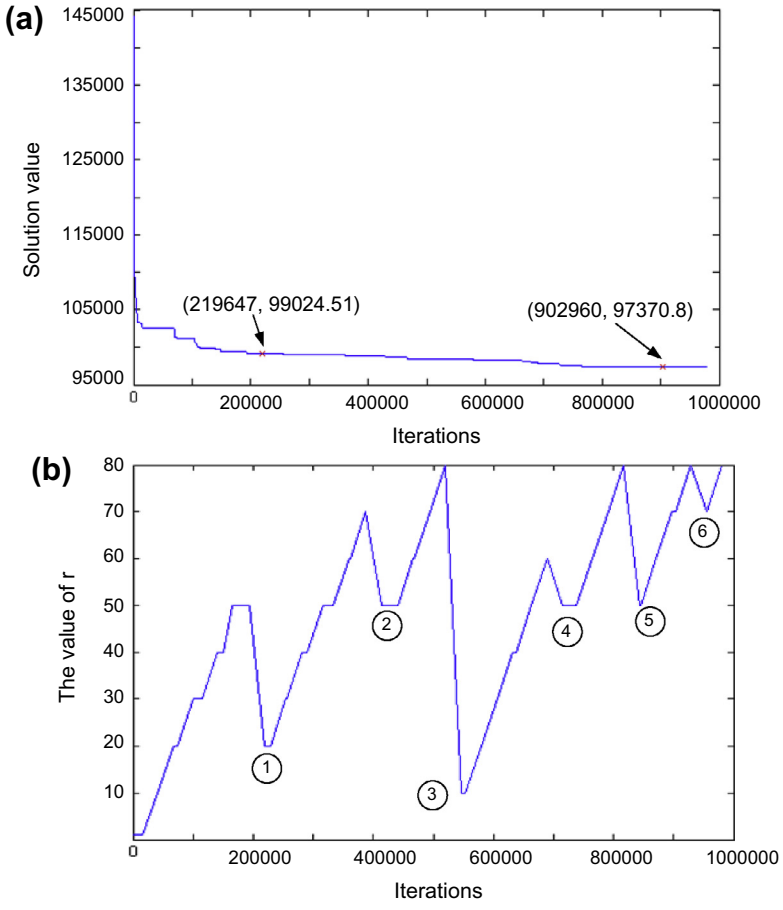

Fig. 4. Impact of diversification and elite-set utilization on algorithm performance.

Table 4
Performance comparison with meta-heuristics for the VRPTW.

Authors	R1	R2	C1	C2	RC1	RC2	CNV/CTD
Garcia et al. (1994)	12.92	3.09	10.00	3.00	12.88	3.75	436
	1317.70	1222.60	877.10	602.30	1473.50	1527.00	65977.00
Rochat and Taillard (1995)	12.25	2.91	10.00	3.00	11.88	3.38	415
	1208.50	961.72	828.38	589.86	1377.39	1119.59	57231.00
Potvin et al. (1996)	12.50	3.09	10.00	3.00	12.63	3.38	426
	1294.50	1154.40	850.20	594.60	1456.30	1404.80	63530.00
Taillard et al. (1997)	12.17	2.82	10.00	3.00	11.50	3.38	410
	1209.35	980.27	828.38	589.86	1389.22	1117.44	57523.00
Chiang and Russell (1997)	12.17	2.73	10.00	3.00	11.88	3.25	411
	1204.19	986.32	828.38	591.42	1397.44	1229.54	58502.00
Backer and Furnon (1999)	14.17	5.27	10.00	3.00	14.25	6.25	508
	1214.86	930.18	829.77	604.84	1385.12	1099.96	56998.00
Schulze and Fahle (1999)	12.25	2.82	10.00	3.00	11.75	3.38	414
	1239.15	1066.68	828.94	589.93	1409.26	1286.05	60346.00
Tan et al. (2001)	13.83	3.82	10.00	3.25	13.63	4.25	467
	1266.37	1080.24	870.87	634.85	1458.16	1293.38	62008.00
Cordeau et al. (2011)	12.08	2.73	10.00	3.00	11.50	3.25	407.00
	1210.14	969.57	828.38	589.86	1389.78	1134.52	57556.00
Lau et al. (2003)	12.17	3.00	10.00	3.00	12.25	3.38	418
	1211.55	1001.12	832.13	589.86	1418.77	1170.93	58477.00
Vidal et al. (2013)	11.92	2.73	10.00	3.00	11.50	3.25	405
	1210.69	951.51	828.38	589.86	1384.17	1119.24	57,196
TS	12.66	3.72	10.00	3.00	12.50	4.25	442
	1231.40	986.70	850.34	609.11	1396.84	1136.72	58425.00

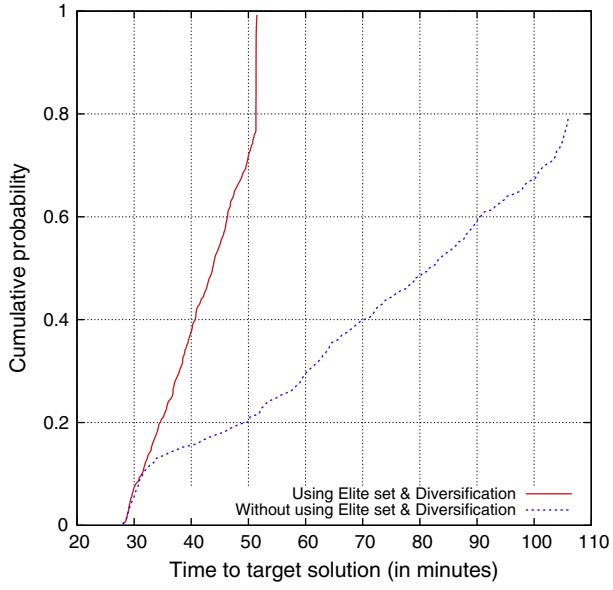

Fig. 5. Time-to-target plot for the instance C2-1.

longer computation times, the solutions of the first improvement strategy are still significantly worse than the solutions of the best improvement strategy.

6.1.4. Elite set calibration, diversification, and run-time behavior

We now turn to the parameters characterizing the diversification procedure and the elite set utilization, and examine their impact on the performance of the algorithm. We complete this part of reporting on the computational experiments, by examining the run-time behavior of the proposed tabu search meta-heuristic.

Four variants of the algorithm were studied corresponding to the different ways to set an elite solution as the new working solution and the inclusion, or not, of the diversification phase. The first two variants simply select an elite solution p at random and re-start the algorithm from it. The *Diversification* mechanism described in Section 5.7 is applied in the last two variants to diversify from the elite solution p.

The initialization of the r parameter following the selection of p is a component common to the four variants. We studied two alternatives where r was set to either the full or half the value at which p was found, respectively (i.e., $r = r_p$ or $r = r_p/2$). The size of the elite set is relevant for the *Diversification* mechanism only. Three values were tested, 1, 5, and 10.

Similar to previous experiments, we used formulas dependent on the problem dimensions for IT_{div} and C_{cNS}, which determine for how long exploration can proceed. Thus, the number of diversification phases is set to $IT_{div} = m'^{*}|\mathcal{S}| + n$, where m' is the number of vehicles used in the initial solution, and $|\mathcal{S}|$ and n are the numbers of supply points and customers, respectively. We also set the number of consecutive executions of the *Control* procedure without improvement of the best solution to $C_{cNS} = \min(3\log_{10}(n/|S|), (r_{\max} - r)/\Delta_r)$, which keeps the value of C_{cNS} sufficiently high during the course of the algorithm, even though *Control* procedure is started with different values of r (remember that $r_{\max} = 50\log_{10}(n/|S|)$). Intuitively, in the beginning, r is small and C_{cNS} takes the value $3\log_{10}(n/|S|)$, while when r becomes large enough, C_{cNS} takes the value $(r_{\max} - r)/\Delta_r$.

Table 3 displays the performance comparison between the four variants with the three different values for the elite set size. For each variant and size of the elite set, the table shows the average gaps to the previous BKS of the average cost of the best solutions of all instances, together with the corresponding average computation time in minutes over 10 runs.

As expected, results indicate that guidance using elite solutions contributes significantly to improve the performance of the algorithm. Without using the elite set, the algorithm requires the lowest computation effort but produces solutions with the lowest average (improvement) gap to the previous BKS of −2.96%, compared to all the variants using the elite set. Comparing the two variants corresponding to the two values at which r is reset, one observes that the solution quality is not very sensitive to this value, but computing effort is increasing when the value of r is lower ($r = r_p/2$).

Table 5
Comparative performances on Crainic et al. (2012) instances.

Problem set	Crainic et al. (2012)					TS						GAP to BKS (%)
	Best	#Vehicles	DM	MWS	Time	Avg 10	Best 10	#Vehicles	DM	MWS	Time	
A1	18,575	24	1	36	5	18043.84	18010.52	23	2	36	18	−3.04
A2	15,411	19	2	42	3	14495.23	14440.05	17	5	42	10	−6.29
B1	55,653	51	14	180	22	53124.00	53036.13	45	31	168	60	−4.62
B2	47,396	39	20	193	10	45272.96	45074.16	34	40	177	39	−4.79
C1	117,426	87	39	423	50	112975.99	112810.90	79	73	394	165	−3.92
C2	101,570	64	68	434	23	97747.18	97585.83	60	108	391	104	−3.91
Average	59338.50	47.33	24	218	18.83	56943.20	56826.27	43	43.17	201.50	66	−4.42

One observes that the third and fourth variants are significantly better in terms of finding high quality solutions. This indicates that the long-term memory and the diversification mechanism added to the algorithm are important features for high performance. Moreover, setting the size of the elite set to 5 achieves a better balance between solution quality and computation time, compared to a larger size of 10. Indeed, doubling the size of the elite set improves only slightly the solution quality, 0.02%, but requires 39% more time. We therefore set the size of the elite set to 5 and reset $r = r_p$.

Fig. 4a and b illustrate the impact of the diversification procedure and the elite set on search performance. These figures report the solution value and the value of r, respectively, along the iterations of the tabu search algorithm on instance C2-1. The number of customers n and supply points $|\mathcal{S}|$ for the instance C2-1 are 3600 and 72, respectively, therefore, the maximum value of the neighborhood-selection parameter $r_{\max} = 80$. The value of r was reset six times (Fig. 4b) corresponding to the re-initialization of the search using six different solutions extracted from the elite set. Prior to the first reset of r, the algorithm proceeded from the solution generated by the initialization procedure, the value of r increasing from 1 to 50 the during that search sequence. Aligning vertically the iteration axis of Fig. 4b and a, one obtains the representation of how the solution quality varies with each search sequence between two consecutive resets. We can see that without using the elite set, the search stopped quite early, namely at iteration 219,647, with a best solution value of 99024.51. On the other hand, using the elite set and the diversification procedure, new improved solutions with substantially better costs were found through the next five search sequences when a solution from the elite set was used to diversify the search. The best solution was found at iteration 902960 (with $r = 70$) from the fifth solution extracted from the elite set. Its value is 97370.8, improving the solution quality by 1.67% compared to the case without using the elite set. This improvement indicates the efficiency of the diversification procedure and the elite set applied in the algorithm. The search stopped at iteration 980723.

To further emphasize the importance of the diversification feature of the proposed meta-heuristic on performance, we focused on the run-time behavior of the proposed algorithm and constructed a time-to-target (TTT) plot (Aiex et al., 2007). A TTT plot is generated by executing an algorithm κ times and measuring the time required to reach a solution at least as good as a target solution. The running times are sorted in increasing order. The i-sorted running time t_i is associated with a probability $\rho_i = (i - 1/2)/\kappa$ and the points $z_i = [t_i, \rho_i]$, $i = 1, \ldots, \kappa$, are plotted. Each plotted point indicates the probability (vertical axis) for the algorithm to achieve the target solution in the indicated time (horizontal axis). Once again, we used the C2-1 instance, with a best known solution value of 97350.2, for this experiment. The plot in Fig. 5 compares two algorithms: the first does not use the elite set and diversification mechanisms while the second applies them. This plot is produced by the execution of both algorithms 200 times, using a target value of 1.5% above the best known solution, i.e., value of 98810.45, together with a maximum computational time of 120 min. We observe in Fig. 5 that both algorithms have similar behaviors until about 31 min. During that period, the second algorithm has not yet applied the elite set and diversification mechanisms. Starting from that point, the probability for the second algorithm to find the target value starts to be greater than for the first algorithm. Moreover, within 120 min, the first algorithm reaches the target with a maximum probability of 0.8. While the second algorithm always reaches the target and requires less than 55 min. This analysis indeed indicates that better performances are obtained when the elite set and diversification procedure are integrated to the search algorithm.

6.2. Comparing with results in the literature

This section compares the performance of the proposed tabu search algorithm with results available in the literature. We initiate this analysis examining the effectiveness of the routing component of the search, as well as the contribution of each set of decisions to the solution quality.

6.2.1. Effectiveness of routing neighborhoods

We have run the proposed tabu search algorithm using only our routing neighborhoods on the Solomon and Desrosiers (1988) 56 VRPTW 100-customer instances. All parameters related to the supply points were discarded. Therefore, $IT_{cNS} = 3{*}n$, and $IT_{div} = n$. Moreover, when the elite set is not yet full, the solution which is most similar to p_{best} is deleted, i.e., the solution p with the smallest $\Delta(p, p_{best}) \leqslant 0.05{*}n$. We executed three runs for each instance, and report the best one.

Table 4 compares the performances obtained by this striped-down version of the tabu search algorithm with the results of other tabu search algorithms reported in the survey on the VRPTW of Bräysy and Gendreau (2005). For completion sake, we also included the results of the currently best-metaheuristic, the hybrid genetic algorithm with adaptive diversity management of Vidal et al. (2013) (best in five repetitions). The first column gives the name of the authors of the study. Columns R1, R2, C1, C2, RC1, and RC2 present the average number of vehicles and average total distance with respect to the six groups of problem instances, respectively. Finally, the rightmost column indicates the cumulative number of vehicles (CNV) and cumulative total distance (CTD) over all 56 instances. The performance of the routing neighborhood search of our algorithm appears satisfactory, considering that all algorithms in the literature were tailored for the VRPTW, most of them (except Backer and Furnon, 1999; Tan et al., 2001) actually aiming first to reduce the number of vehicles. We do not, as our algorithm treats vehicles through supply-point neighborhoods not considered in this experiments, and we do not compete with the other meta-heuristics on this count. We do compete with respect to the total distance, though, outperforming six out of the eleven meta-heuristics.

Table 6
Impact of waiting cost on solution quality.

Impact on solution quality	Increase the cost of waiting by					
	10%	20%	30%	40%	50%	100%
Total cost (%)	2.87	5.15	7.46	9.75	11.77	20.23
Routing cost (%)	3.78	6.78	10.04	13.13	15.39	26.59
#Vehicles (%)	0.82	1.47	1.64	2.13	3.61	5.91
DM (%)	3.91	15.14	23.24	31.30	34.19	58.41
MWS (%)	−5.45	−8.68	−10.90	−12.97	−14.14	−21.38

Table 7
Impact of vehicle fixed cost on solution quality.

Fixed cost *F*	Total cost	Traveling cost	#Vehicles	DM	MWS	#Legs
0	17584.74	17584.74	290	0	0	290
250	46105.71	35105.71	44	41	202	287
500	56943.06	35443.06	43	42	202	287
1000	77740.31	35740.31	42	44	201	287

Using routing neighborhoods only, is not sufficient for the problem setting at hand, however. We have actually also run our algorithm by fixing the supply point-to-vehicle assignment of the initial solution and applying only routing neighborhoods. We obtained worse solutions than the proposed tabu search handling both neighborhood types, with an average error gap of 5.91% (see details in Annex A). Therefore, although we have created efficient routing neighborhoods, we cannot get good solutions by considering routing only. Both sets of decisions are important. Our algorithm handles both neighborhood types efficiently, and is able to enhance the solution quality significantly by using long-term memory and diversification mechanisms. Performance comparisons of the proposed algorithm with the literature are presented in the next subsection.

6.2.2. The performance of the proposed algorithm

The performance of the proposed tabu search meta-heuristic is evaluated by comparing its output with published results on the instances provided by Crainic et al. (2012). For concision sake, only aggregated results are provided in this section. Details can be found in Annex B.

Table 5 displays the comparison between the (best) results reported by Crainic et al. (2012) and those obtained by the proposed tabu search meta-heuristic run 10 times for each instance. For comparison sake, we report the best results we obtained for the 10 runs, but provide both best and average results in the detailed tables of Annex B. Table 5 gives the best results (*Best* column), the number of vehicles (*#Vehicles* column), the number of times vehicles move directly from one customer zone to another customer zone without using waiting stations (*DM* column), and the number of times waiting stations are used for moving between customer zones (*MWS* column). Average computation times in minutes are displayed in the *Time* column, while the corresponding gaps to the previous BKS are given in the last column.

TS produces high-quality solutions, with an average improvement gap of −4.42% compared to the previous BKS, yielding better solutions than Crainic et al. (2012) on all instances. Moreover, the proposed tabu search meta-heuristic produces consistently good solutions, the average solution quality being very close to that of the best one. Noteworthy, as shown in Table 3, without post-optimization and using the same size of the elite set (=5), TS obtains an average gap to the previous BKS of −4.33%. Thus, the post-optimization process helped to improve solution quality by 0.09% on average, requiring only a few extra seconds.

TS produces solutions that not only require less vehicles (8.87% on average), but also require less usage of waiting stations. More precisely, the TS we propose used waiting stations 12,050 times compared to 13,071 times in Crainic et al. (2012), vehicles move directly from one customer zone to another customer zone on 2590 occasions compared to 1449 occasions in Crainic et al. (2012). Thus, in our TS, 17.69% of the times vehicles move directly to another customer zone without using waiting stations, compared to 9.98% in Crainic et al. (2012). Moreover, the proposed TS provides better customer routing (i.e., traveling cost), with an average gap of −1.30% to the previous BKS. This advantage goes beyond the simple numerical performance in terms of cost to propose a distribution system that is structurally better with less vehicles traveling for idling purposes.

6.3. Synchronization at supply points

Waiting stations are introduced as locations where vehicles can wait when the direct move would get them at supply points sooner than the opening times. In this section, we analyze the impact of waiting stations on solution quality.

In all previous experiments, traveling cost and time were equivalent. In order to analyze the impact of waiting without modifying the travel costs, we explicitly introduced into the model a waiting cost measure related to the customer-to-waiting station movement generating the need to wait. Thus, the waiting cost for a (customer demand, supply point) pair is computed as a percentage of the total cost from the customer to the waiting station and from the latter to the supply point. We performed 6 runs corresponding to a percentage equal to 10%, 20%, 30%, 40%, 50%, and 100%.

The experiment was run on the C2 set, which includes the largest instances in terms of the numbers of customers, supply points, and waiting stations. Table 6 sums up the solution-quality variations for the six cases compared to the case without waiting costs. The table displays the solution-quality variations in terms of the total cost, traveling cost, number of vehicles, and synchronization requirements at supply points. One observes that higher waiting cost result in vehicles performing longer routes (the routing cost increases) to avoid going to waiting stations. Consequently, the number of direct moves increases, and accordingly, the number of moves using waiting stations is reduced. Moreover, it also requires more vehicles, resulting in a higher total cost.

6.4. Impact of vehicle fixed cost

The objective of the TMZT-VRPTW is to minimize the total cost of the system, comprised on the total "fixed" cost of using vehicles and the "variable" cost of delivering the loads to customers by the vehicles. In this section, we analyze the impact of vehicle fixed cost on solution quality.

In all previous experiments, a value of 500 was set to the fixed cost *F* of using a vehicle. We performed three additional runs setting the fixed cost *F* equal to 0, 250, and 1000. Table 7 sums up the solution-quality variations in terms of total cost, traveling cost, number of vehicles, synchronization requirements at supply points, and number of legs. One observes that vehicle fixed cost

plays an important role in the problem solution. Setting the fixed cost to zero represents the case where the objective of the problem aims only to minimize the traveling cost. The experimental results in Table 7 illustrate clearly that the traveling cost is lower for this case compared to the other cases where vehicle fixed costs are considered. On the other hand, this case puts on the road the largest number of vehicles, each vehicle performing exactly one leg. While this case eliminates the need for waiting stations, it would correspond to the highest cost in terms of manpower, as well as impact on congestion and emissions. Introducing a vehicle fixed cost dramatically reduces the number of vehicles and generates solutions where each vehicle performs a sequence of several of legs and, sometimes stops at waiting stations. Increasing the vehicle fixed cost continues to decrease the number of vehicles, as well as the utilization of the waiting stations, while increasing the traveling cost. These modifications are not dramatic, however. Thus, for example, doubling the value of F (i.e., from 250 to 500, or from 500 to 1000 in this experiment) increases the traveling cost by less than 1% and reduces the number of vehicles by less than 2.5%. Generally speaking, once vehicle fixed costs are introduced, the pattern of solutions is not very sensitive to its value.

7. Conclusions

We proposed a tabu search meta-heuristic for the Time-dependent Multi-zone Multi-trip Vehicle Routing Problem with Time Windows. The proposed model formulation provided the means to identify clearly the main components of the decision set of the problem. We could thus propose a tabu search method that works on multiple neighborhoods, which are used to improve both the routing and the assignment of routes to vehicles. The selection of neighborhoods is dynamically adjusted along the search to keep the balance between exploration and exploitation. Moreover, a diversification strategy guided by an elite set and a frequency-based memory is introduced to not only provide a certain level of diversity to the search, but also help incorporate good attributes into newly created solutions.

Experimental results illustrated clearly the superior performance of the proposed methodology compared to the literature. It yields higher quality solutions in terms of both required number of vehicles and traveling cost. In addition, the utilization of waiting stations, resulting from the synchronization restriction at supply points, was significantly reduced. The quality of these results in terms of number of required vehicles, costs, times, and frequency of direct movements are not only extremely satisfying when evaluating the proposed meta-heuristic, but are also interesting from a managerial point of view. Indeed, they indicate that our tabu search will prove magisterially efficient in actual applications, contributing to provide system managers with solution requiring less vehicles to perform efficiently the same amount of work. This is particularly interesting when City Logistics systems are contemplated as our results indicate a reduction in the presence of vehicles on the streets of the city and, thus, in their negative impact on congestion and environment.

Acknowledgments

Partial funding for this project has been provided by the Natural Sciences and Engineering Council of Canada (NSERC), through its Industrial Research Chair and Discovery Grant programs, by the Université de Montréal, and by the EMME/2-STAN Royalty Research Funds (Dr. Heinz Spiess). We also gratefully acknowledge the support of Fonds de recherche du Québec through their infrastructure grants and of Calcul Québec and Compute Canada through access to their high-performance computing infrastructure. The authors want also to thank three anonymous referees whose comments and inquiries helped write a better paper.

Appendix A. Supplementary material

Supplementary data associated with this article can be found, in the online version, at http://dx.doi.org/10.1016/j.ejor.2013.05.026.

References

Aiex, R.M., Resende, M.G.C., Ribeiro, C.C., 2007. TTTPLOTS: a perl program to create time-to-target plots. Optimization Letters 1, 355–366.
Backer, B.D., Furnon, V., 1999. Local search in constraint programming: experiments with tabu search on the vehicle routing problem. In: Voss, S., Martello, S., Osman, I., Roucairol, C. (Eds.), Meta-Heuristics. Springer, US, pp. 63–76.
Benjamin, A., Beasley, J., 2010. Metaheuristics for the waste collection vehicle routing problem with time windows, driver rest period and multiple disposal facilities. Computers & Operations Research 37 (12), 2270–2280.
Braca, J., Bramel, J., Simchi-levi, D., 1997. A computerized approach to the New York city school bus routing problem. IIE Transactions 29, 693–702.
Brandão, J., Mercer, A., 1998. The multi-trip vehicle routing problem. The Journal of the Operational Research Society 49 (8), 799–805.
Bräysy, O., Gendreau, M., 2005. Vehicle Routing Problem with time windows, Part II: Metaheuristics. Transportation Science 39 (1), 119–139.
Chiang, W.-C., Russell, R.A., 1997. A reactive tabu search metaheuristic for the vehicle routing problem with time windows. INFORMS Journal on Computing 9 (4), 417–443.
Cordeau, J.F., Laporte, G., Mercier, A., 2011. A unified tabu search heuristic for vehicle routing problems with time windows. Journal of the Operational Research Society 52, 928–936.
Crainic, T.G., Ricciardi, N., Storchi, G., 2009. Models for evaluating and planning city logistics systems. Transportation Science 43 (4), 432–454.
Crainic, T.G., Gendreau, M., Gajpal, Y., 2012. Multi-zone Multi-trip Vehicle Routing Problem with Time Windows. Technical Report CIRRELT-2012-36, Centre interuniversitaire de recherche sur les réseaux d'entreprise, la logistique et le transport, Université de Montréal, Montréal, QC, Canada.
Crevier, B., Cordeau, J.-F., Laporte, G., 2007. The multi-depot vehicle routing problem with inter-depot routes. European Journal of Operational Research 176 (2), 756–773.
Dell'Amico, M., Trubian, M., 1993. Applying tabu search to the job-shop scheduling problem. Annals of Operations Research 41, 231–252.
Desrosiers, J., Ferland, J., Rousseau, J.-M., Lapalme, G., Chapleau, L., 1981. An overview of a school busing system. In: Jaiswal, N. (Ed.), Scientific Management of Transport Systems. North-Holland, pp. 235–243.
Ehmke, J.F., Steinert, A., Mattfeld, D.C., 2012. Advanced routing for city logistics service providers based on time-dependent travel times. Journal of Computational Science 3 (4), 193–205.
Garcia, B.-L., Potvin, J.-Y., Rousseau, J.-M., 1994. A parallel implementation of the tabu search heuristic for vehicle routing problems with time window constraints. Computers & Operations Research 21 (9), 1025–1033.
Gaspero, L.D., Schaerf, A., 2007. A composite-neighborhood tabu search approach to the traveling tournament problem. Journal of Heuristics 13 (2), 189–207.
Hamiez, J.-P., Robet, J., Hao, J.-K., 2009. A tabu search algorithm with direct representation for strip packing. In: Cotta, C., Cowling, P. (Eds.), Evolutionary Computation in Combinatorial Optimization, Lecture Notes in Computer Science, vol. 5482. Springer, Berlin Heidelberg, pp. 61–72.
Kim, B.-I., Kim, S., Sahoo, S., 2006. Waste collection vehicle routing problem with time windows. Computers & Operations Research 33 (12), 3624–3642.
Lau, H.C., Sim, M., Teo, K.M., 2003. Vehicle routing problem with time windows and a limited number of vehicles. European Journal of Operational Research 148 (3), 559–569.
Lin, S., 1965. Computer solutions of the traveling salesman problem. Bell System Technical Journal 44, 2245–2269.
Ombuki-Berman, B.M., Runka, A., Hanshar, F.T., 2007. Waste collection vehicle routing problem with time windows using multi-objective genetic algorithms. In: Andonie, R. (Ed.), Proceedings of the Third IASTED International Conference on Computational Intelligence. ACTA Press, pp. 91–97.
Or, I., 1976. Traveling Salesman-type Combinatorial Problems and their relation to the Logistics of Blood Banking. PhD thesis, Department of Industrial Engineering and Management Science, Northwestern University, Evanston, IL.
Osman, I.H., 1993. Metastrategy simulated annealing and tabu search algorithms for the vehicle routing problems. Annals of Operations Research 41, 421–452.
Park, J., Kim, B.-I., 2010. The school bus routing problem: a review. European Journal of Operational Research 202 (2), 311–319.
Petch, R.J., Salhi, S., 2003. A multi-phase constructive heuristic for the vehicle routing problem with multiple trips. Discrete Applied Mathematics 133, 69–92.
Potvin, J.-Y., Kervahut, T., Garcia, B.-L., Rousseau, J.-M., 1996. The vehicle routing problem with time windows, Part I: Tabu search. INFORMS Journal on Computing 8, 157–164.
Rochat, Y., Taillard, R.D., 1995. Probabilistic diversification and intensification in local search for vehicle routing. Journal of Heuristics 1, 147–167.
Salhi, S., Petch, R., 2007. A GA based heuristic for the vehicle routing problem with multiple trips. Journal of Mathematical Modelling and Algorithms 6, 591–613.

Schulze, J., Fahle, T., 1999. A parallel algorithm for the vehicle routing problem with time window constraints. Annals of Operations Research 86, 585–607.
Solomon, M.M., 1987. Algorithms for the vehicle routing and scheduling problems with time window constraints. Operations Research 35, 254–265.
Solomon, M.M., Desrosiers, J., 1988. Time window constrained routing and scheduling problems. Transportation Science 2 (1), 1–13.
Taillard, E.D., Laporte, G., Gendreau, M., 1995. Vehicle routing with multiple use of vehicles. The Journal of the Operational Research Society 47 (8), 1065–1070.
Taillard, E.D., Badeau, P., Gendreau, M., Guertin, F., Potvin, J.-Y., 1997. A tabu search heuristic for the vehicle routing problem with soft windows. Transportation Science 31, 170–186.
Tan, K.C., Lee, L.H., Zhu, Q.L., Ou, K., 2001. Heuristic methods for vehicle routing problem with time windows. Artificial Intelligence in Engineering 15 (3), 281–295.
Tarantilis, C.D., Zachariadis, E.E., Kiranoudis, C.T., 2008. A hybrid guided local search for the vehicle-routing problem with intermediate replenishment facilities. INFORMS Journal on Computing 20 (1), 154–168.
Vidal, T., Crainic, T.G., Gendreau, M., Lahrichi, N., Rei, W., 2012. A hybrid genetic algorithm for multi-depot and periodic vehicle routing problems. Operations Research 60 (3), 611–624.
Vidal, T., Crainic, T.G., Gendreau, M., Prins, C., 2013. A hybrid genetic algorithm with adaptive diversity management for a large class of vehicle routing problems with time windows. Computers & Operations Research 40 (1), 475–489.
Xu, J., Sohoni, M., McCleery, M., Bailey, T.G., 2006. A dynamic neighborhood based tabu search algorithm for real-world flight instructor scheduling problems. European Journal of Operational Research 169 (3), 978–993.