CIRRELT

Centre interuniversitaire de recherche
sur les réseaux d'entreprise, la logistique et le transport

Interuniversity Research Centre
on Enterprise Networks, Logistics and Transportation

Synchronized Multi-Trip Multi-Traffic Pickup & Delivery in City Logistics

Teodor Gabriel Crainic
Phuong Khanh Nguyen
Michel Toulouse

February 2015

CIRRELT-2015-05

Bureaux de Montréal :
Université de Montréal
Pavillon André-Aisenstadt
C.P. 6128, succursale Centre-ville
Montréal (Québec)
Canada H3C 3J7
Téléphone : 514 343-7575
Télécopie : 514 343-7121

Bureaux de Québec :
Université Laval
Pavillon Palasis-Prince
2325, de la Terrasse, bureau 2642
Québec (Québec)
Canada G1V 0A6
Téléphone : 418 656-2073
Télécopie : 418 656-2624

www.cirrelt.ca

Synchronized Multi-Trip Multi-Traffic Pickup & Delivery in City Logistics

Phuong Khanh Nguyen[1,2], Teodor Gabriel Crainic[1,2,*], Michel Toulouse[1,3]

[1] Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation (CIRRELT)

[2] Department of Management and Technology, Université du Québec à Montréal, P.O. Box 8888, Station Centre-Ville, Montréal, Canada H3C 3P8

[3] Vietnamese-German University, Le Lai Street, Hoa Phu Ward, Binh Duong New City, Binh Duong Province, Vietnam

Abstract. The paper introduces the first methodology addressing with a single fleet of vehicles the routing of the three different types of transportation demands encountered in City Logistics, inbound, outbound and intra-city traffic. We propose a tabu search meta-heuristic calling on various neighbourhoods, dynamically selected, to provide an efficient search combining exploration and exploitation capabilities. The result analysis of extensive computational experiments qualify the impact of a number of major problem characteristics and search strategies on the quality of the meta-heuristic, the behaviour of the solutions, and the management of the City Logistics system.

Keywords: Vehicle routing, pickup and delivery, synchronization, inbound, outbound and intra-city transport demand, tabu search.

Acknowledgements. The authors thank the Discovery Grant and the Discovery Accelerator Supplements Program of the Natural Sciences and Engineering Research Council of Canada (NSERC), and the Strategic Clusters program of the Fonds de recherche du Québec – Nature et technologie (FRQNT) for supporting this research.

Results and views expressed in this publication are the sole responsibility of the authors and do not necessarily reflect those of CIRRELT.

Les résultats et opinions contenus dans cette publication ne reflètent pas nécessairement la position du CIRRELT et n'engagent pas sa responsabilité.

---

* Corresponding author: TeodorGabriel.Crainic@cirrelt.ca

Dépôt légal – Bibliothèque et Archives nationales du Québec
Bibliothèque et Archives Canada, 2015

© Nguyen Khanh, Crainic, Toulouse and CIRRELT, 2015

INTRODUCTION

Most City Logistics (CL) literature and projects address inbound movements only, reflecting the dominant position the traffic proceeding from the exterior of the city towards its centre occupies within the travel patterns observed in most cities. Yet, the volumes of freight produced within the city and shipped to locations within or outside it may be significant. Moreover, addressing the needs of these different types of transportation demands in a comprehensive manner and with a unique fleet of vehicles, may greatly contribute to achieve the CL mobility, environmental, and quality-of-life objectives. It is therefore relevant to investigate the possible integration of these traffic types into "normal" CL operations. The goal of this paper is to contribute to this investigation.

To our best knowledge, Crainic *et al.* (2012) were the first to investigate this issue within the context of two-tiered City Logistics systems (Crainic *et al.,* 2009). They examined, at a conceptual level, several integration strategies of three types of traffic, the *customer-to-customer* (*c2c* for traffic with origin and destination at customers located within the CL-controlled part of the city, i.e., the city centre), the *customer-to-external zone* (*c2e* from the city centre to destinations outside the city limits), and the "classical" *external zone-to-customer* (*e2c*). They discussed methodological and managerial challenges associated to the integration, but no problem definition was provided, nor any modelling or algorithmic contribution. We formally introduce and define the *Multi-trip Multi-traffic Pickup and Delivery Problem with Time Windows and Synchronization* (*MTT-PDTWS*) addressing the three traffic types.

The first algorithm for the MTT-PDTWS is our second contribution. We extend the work of Nguyen *et al.* (2013, 2015) to address the challenges of integrating c2c operations into route planning. Computational results are discussed to qualify the impact of a number of major problem characteristics, parameters and search strategies on the quality of the meta-heuristic, the behaviour of the solutions, and the management of the City Logistics system.

The paper is structured as follows. We define the problem and set it within the literature in next section. The tabu search meta-heuristic and its components are presented next, followed by the section dedicated to the experimental results, and the conclusion.

PROBLEM DESCRIPTION

In the MTT-PDTWS, a homogeneous fleet of vehicles of capacity Q operates out of a single garage g to perform multiple-tour delivery and pickup routes servicing three types of customer requests: e2c, c2e and c2c. The MTT-PDTWS is a new variant in the vehicle routing problem class, the original characteristics setting it apart from and generalizing most *Pickup and Delivery with Backhauls* (*P&DB*) problems (e.g., Cherkesly *et al.,* 2014; Vidal *et al.,* 2014) being 1) multicommodity demand defined as time-dependent origin-to-destination customer requests; 2) synchronization of activities at facilities; and 3) multi-tour routes.

We model the time-dependency characterizing demand and operations in the MTT-PDTWS through time windows. We first model facilities, which become available to receive vehicles for loading and unloading operations at particular time periods only. A particular set of loads destined to specific customers may be available at each such time period, and must be taken away and distributed. Then, as a given facility may be open at several moments during the schedule length considered, with a different set of loads at each occurrence, we define supply points as particular combinations of facilities and availability time periods. Each supply point $s \in S$ has a no-wait, hard opening time window $[t(s) - \eta, t(s)]$ specifying the earliest and latest times a vehicle may arrive at s, respectively. The vehicle may stop at a waiting station (e.g., a parking lot) $w \in W$ before moving to s.

The second time-dependency phenomenon concerns customers requesting different services: 1) receive e2c loads from different supply points, possibly within different time windows; 2) ask for c2e loads to be picked up and transported to one of a given subset of supply points; 3) specific pairs of customers may ship c2c loads between them. Each customer may require more than one of these three service types. We model this time dependency by identifying each particular load as a customer demand and defining 1) The set of *delivery-customer demands*, $d \in C^D$, each being characterized by the supply point where it is available, the customer it must be delivered to, and the time window when the delivery must be performed; 2) The set of *pickup-customer demands*, $p \in C^P$, each characterized by the customer shipping it and the time window within which the pickup must be performed, as well as by the set of admissible supply points $S_p \in S$ to which the load may be delivered, the choice of a particular one being part of the decisions characterizing the MTT-PDTWS; and 3) The set $(\overline{p}, \overline{d}) \in \mathcal{R}$ of *c2c-customer demands*, each request requiring a load to be transported from a *c2c-pickup-customer* demand $\bar{p} \in C^P_{c2c}$ to a *c2c-delivery-customer* demand $\bar{d} \in C^D_{c2c}$. Let $(i, q_i, \delta(i), [e_i, l_i])$ stand for the quantity $q_i > 0$ ($q_{\overline{p}} = -q_{\overline{d}}$ for $(\overline{p}, \overline{d}) \in \mathcal{R}$) to be delivered or picked up at the customer demand $i \in C^P \cup C^D \cup C^P_{c2c} \cup C^D_{c2c}$ within its hard time window $[e_i, l_i]$, with a service time $\delta(i)$.

Each supply point s may service a group of either pickup-customer demands $C^P_s \subseteq C^P$, or delivery-customer demands $C^D_s \subseteq C^D$, or both. The loads collected from pickup-customer demands in C^P_s are brought to s during one of its time windows. Similarly, the freight to be delivered to delivery-customer demands in C^D_s has to be loaded at s during the time window. Let $\varphi(s)$ and $\varphi'(s)$ be the times required, to load and unload a vehicle at s, respectively.

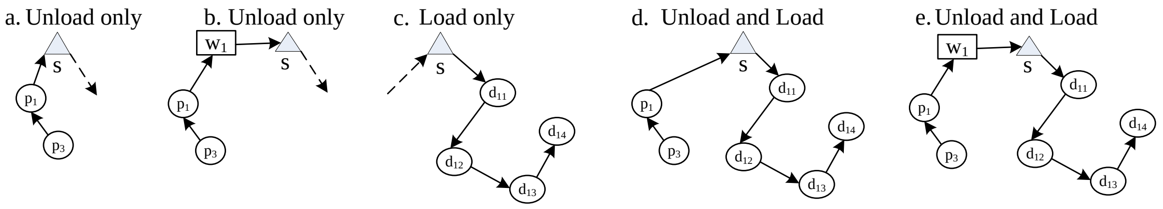

Figure 1. Activities at supply points

Let a *c2c leg* be a route servicing one or a sequence of requests following the last-in-first-out (LIFO) policy to ensure that no handling is required while unloading freight from the vehicle. Let a *pickup* or *delivery leg* be a route that links one or a sequence of pickup- or delivery-customer demands, respectively, and a supply point. Figure 1 represents the possible vehicle loading and unloading activities at supply point s, the dashed lines standing for empty moves. Figures 1a and 1b depict the case of pickup legs with "unload only" operations when the vehicle arrives with the collected freight from pickup-customer demands, unloads it and leaves empty for its next tour or the garage to end its activity. The two instances differ in the level of synchronization only. The vehicle goes directly to the supply point from the last serviced customer demand if it can arrive within the time window $[t(s) - \eta, t(s)]$ (Figure 1a), or waits at waiting station w_1 for the facility to open (Figure 1b), otherwise. Figure 1c represents a delivery leg with the "load only" case when the vehicle arrives empty and loads. Figures 1d and 1e depict instances of "unload & load" operations, when the vehicle performs a pickup leg (unloading all the freight collected from pickup-customer demands), then a delivery leg (loads freight and delivers it to the designated delivery-customer demands) at *s*.

A sequence of legs, starting and ending at the garage and performed by a single vehicle, is called a *work assignment*. Vehicles operate according to the *Pseudo-Backhaul* strategy (Crainic *et al.*, 2012), in which a leg must be completed before another one may start.

Figure 2 illustrates a six-leg work assignment, where s_1, s_2, s_3 are supply points, g and w_1 are the garage and waiting station, respectively, $C_{s_1}^D = \{d_1, d_2, .., d_5\}$, $C_{s_2}^P = \{p_1, p_2, .., p_5\}$, $C_{s_2}^D = \{d_6, d_7, .., d_{11}\}$, $C_{s_3}^P = \{p_6, p_7, .., p_{10}\}$ and $C_{s_3}^D = \{d_{12}, .., d_{15}\}$ are pickup- and delivery-customer demand sets, and $(\overline{p_1}, \overline{d_1}), (\overline{p_2}, \overline{d_2}), (\overline{p_3}, \overline{d_3}), (\overline{p_4}, \overline{d_4}), (\overline{p_5}, \overline{d_5}) \in \mathcal{R}$ with c2c-pickup-customer demands $\overline{p_1}, \dots, \overline{p_5} \in C_{c2c}^P$ and c2c-delivery-customer demands $\overline{d_1}, \dots, \overline{d_5} \in C_{c2c}^D$. Dashed lines stand for empty travel. The vehicle operating this work assignment performs a sequence of six legs $\{r_1, r_2, \dots, r_6\}$ where $r_1 = \{s_1, d_1, d_3, d_4\}$ and $r_2 = \{s_2, d_6, d_7, d_9, d_8\}$ are delivery legs, $r_3 = \{p_1, p_3, p_4, w_1, s_2\}$ and $r_4 = \{p_6, p_8, p_7, s_3\}$ are pickup legs, $r_5 = \{\overline{p_4}, \overline{p_3}, \overline{d_3}, \overline{p_5}, \overline{d_5}, \overline{d_4}\}$ and $r_6 = \{\overline{p_1}, \overline{p_2}, \overline{d_2}, \overline{d_1}\}$ are c2c legs. The vehicle first moves empty out of the garage g to supply point s_1 and starts loading delivery demands. After loading for a time $\varphi(s_1)$, it leaves s_1 to service delivery-customer demands (d_1, d_3, d_4) in $C_{s_1}^D$, then moves empty to pickup customer zone $C_{s_2}^P$ for collecting freight at pickup-customer demands (p_1, p_3, p_4). For synchronization reasons, the vehicle goes from customer demand p_4 to the waiting station w_1 and waits there in order to arrive at s_2 within its opening time window. Once at s_2 (at some arrival time t), it unloads for a duration of $\varphi'(s_2)$, loads from time $t + \varphi'(s_2)$ for a duration $\varphi(s_2)$, and leaves s_2 to service delivery-customer demands (d_6, d_7, d_9, d_8) in $C_{s_2}^D$. After servicing customer demand d_8, it services three requests $(\overline{p_3}, \overline{d_3}), (\overline{p_4}, \overline{d_4}), (\overline{p_5}, \overline{d_5})$, in the LIFO order $(\overline{p_4}, \overline{p_3}, \overline{d_3}, \overline{p_5}, \overline{d_5}, \overline{d_4})$. The vehicle then moves empty to the next pickup customer zone $C_{s_3}^P$. There, after loading freight at pickup-customer demands (p_6, p_8, p_7), the vehicle moves to supply point s_3 within its opening time window. Once at s_3, the vehicle starts unloading freight for a duration of $\varphi'(s_3)$. The vehicle finally executes its last c2c leg, collecting freight at $(\overline{p_1}, \overline{p_2})$ and delivering it to their c2c-delivery-customer demands $(\overline{d_2}, \overline{d_1})$. At the end, the vehicle moves empty back to g to complete its work assignment.

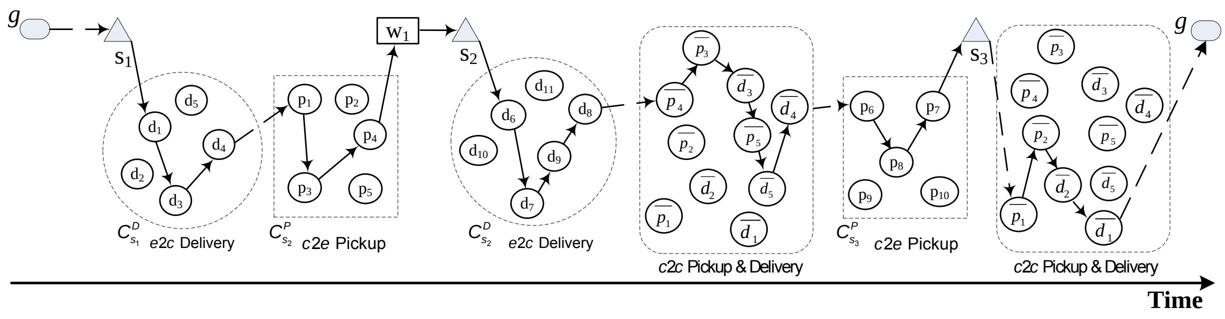

Figure 2. An integrated route illustration

Let F stand for the fixed cost for operating a vehicle, c_{ij} the cost associated with each pair of sites (supply points, waiting stations, and customer demands), and $i, j \in \{g \cup C^D \cup C^P \cup C_{c2c}^P \cup C_{c2c}^D \cup S \cup W\}$ making up the set of nodes of the complete space-time network describing the problem. The MTT-PDTWS can then be seen as the problem of (1) assigning pickup-customer demands to supply points, and (2) selecting a set of work assignments (pickup, delivery and c2c legs), each to be performed by one vehicle. The objective is to minimize the total cost, made up of the routing cost of operating the work assignments and the fixed cost of using the vehicles, while the following conditions are satisfied:

- Every vehicle starts and ends its work assignment (leg sequence) at the garage g;
- Each pickup-customer demand p is assigned to exactly one supply point $s \in S_p$;
- Every vehicle required to service (1) customer demands in $C_s^P \cup C_s^D$ must reach the supply point $s \in S$ within its hard time window (it may wait at a waiting station, eventually); (2) customer demands in $C_{c2c}^P \cup C_{c2c}^D$ must satisfy the LIFO policy;
- Every customer demand $i \in C^P \cup C^D \cup C_P^{c2c} \cup C_D^{c2c}$ is visited by exactly one vehicle (it belongs to exactly one leg) with a total load not exceeding Q, and is serviced within its hard time window.

  The Annex contains the complete mathematical model.

LITERATURE REVIEW

The MTT-PDTWS we introduce in this paper is a new vehicle routing problem class generalizing both a number of pickup and delivery problem settings and the routing problems typically studied in the City Logistics literature.

The MTT-PDTWS extends the Time-dependent Multi-zone Multi-trip Vehicle Routing problem with Time Windows (TMZT-VRPTW; Crainic *et al.* 2009; Nguyen *et al.*, 2013) and the Multi-trip Pickup and Delivery Problem with Time Windows and Synchronization (MT-PDTWS; Nguyen *et al.*, 2015) by integrating the three main types of customer demands. The TMZT-VRPTW addresses the demands for inbound transport and delivery, which corresponds to the delivery-customer demands in our setting. Crainic *et al.* (2009) introduced the TMZT-VRPTW and proposed a decomposition-based heuristic. Nguyen *et al.* (2013) proposed a tabu search framework yielding higher quality solutions when compared to the previous approach (up to 4.42% reduction in total cost with less vehicles and less usage of waiting stations). Nguyen *et al.* (2015) extended this tabu search to address the MT-PDTWS, which considers only delivery- and pickup-customer demands, identifying solutions with an average optimality gap of 1.62% for the instances with known optimal solution values (Bettinelli *et al.*, 2015). The MTT-PDTWS thus brings the literature closer to the actual requirements of a City Logistics system.

The Vehicle Routing Problem with Cross-Docking (VRPCD) partially shares the requirement of synchronizing vehicle operations with the MTT-PDTWS. In the VRPCD, products are picked up from suppliers by a fleet of vehicles, consolidated at a cross-dock facility (i.e., sorted into groups according to their destinations), and immediately delivered to customers by the same set of vehicles, without delay or storage. A supplier and its customers are not necessarily serviced by the same vehicle. At the cross-dock facility, the unloading of a vehicle must be completed before reloading starts. Constraints might be imposed on the simultaneous vehicle arrivals at the facility (Liao *et al.*, 2010), or the arrival dependency among vehicles might be determined by the consolidation decisions (Wen *et al.*, 2008). Similarly to our problem, each vehicle thus operates pickup and delivery phases separately. There are also significant differences between the VRPCD and the MTT-PDTWS, however, and one might see the former as a very particular special case of the later. Thus, in the VRPCD, each vehicle performs a single-tour route composed of a sequence of two trips, first pickup and then delivery, using the cross-dock facility as intermediate storage. There are no such limitations in the MTT-PDTWS, neither on the number of legs (trips), nor on their sequencing (note than the Pseudo-Backhaul rule permits sequencing several legs of the same type). This results in multiple synchronization requirements for each MTT-PDTWS work assignment (route).

The MTT-PDTWS generalizes one-to-many-to-one (1-M-1) and one-to-one (1-1) pickup and delivery problems, the former standing for settings where goods are delivered from a depot to delivery customers and from pickup customers to the depot, while in the latter, goods are transported between specific pickup and delivery locations. It actually combines the two settings into a general framework that also includes multiple tours and synchronization characteristics. With respect to the e2c and c2e transport demands, the MTT-PDTWS includes the 1-M-1 setting, generalizing the P&DB (e.g., Gélinas *et al.*, 1995; Thangiah *et al.*, 1996; Brandão, 2006; Vidal *et al.*, 2014), which builds single-tour routes of, first, delivery-customer demands out of the supply point s and, second, pickup-customer demands assigned to supply point s' where $t(s) < t(s')$. Time synchronization restrictions at supply points and waiting stations are not considered. Two variants with and without time windows at customers are considered in the P&DB literature. On the other hand, the handling of the c2c transport demand of the MTT-PDTWS belongs to the 1-1 scheme. The first studies of 1-1 pickup and delivery with LIFO policy focused mainly on single-tour traveling salesman problems with pickup and delivery (e.g., Carrabs et al., 2007; Li *et al.*, 2011). Later, Cheang *et al.* (2012) studied an extended variant by considering the use of multiple vehicles and setting a limitation on the total distance that a vehicle can travel. More recently, Cherkesly *et al.* (2014) introduced a variant of 1-1 pickup and delivery with LIFO policy considering time windows, which may be seen as the c2c component of the MTT-PDTWS.

SOLUTION METHOD

We propose a tabu search meta-heuristic for the MTT-PDTWS. It extends the solution method of Nguyen *et al.* (2015) for the MT-PDTWS to address the challenges of treating c2c operations. New neighbourhoods are thus proposed tackling the movements within c2c legs, and between these and the other two types of legs.

General structure

The overall structure of the MTT-PDTWS tabu search algorithm is given in Algorithm 1. An initial feasible solution z is generated first using a greedy method seeking to fully utilize vehicles and minimize the total cost. At each iteration of the TS method, one neighbourhood is selected probabilistically based on the current value of the neighbourhood-selection parameter $\overline{r}$, then the selected neighborhood is explored, and the best move is chosen (lines 5-6; an aspiration criterion is used). The algorithm adds the new solution to an elite set $\mathcal{E}$ when it improves z_{best}, records the corresponding value of the parameter $\overline{r}$, and updates the elite set $\mathcal{E}$ by removing a solution based on its value and the difference between solutions (lines 9-10).

Initially, the search freely explores the solution space by assigning the same selection probability ($\overline{r} = 1$) to each neighbourhood. Whenever the best TS solution z_{best} is not improved for IT_{cNS}TS iterations (line 13), the *Control* procedure reduces the probability $\overline{r}$ of selecting leg neighbourhood (line 22). Routing neighbourhoods are then proportionally selected more often, giving customer moves more opportunity to fully optimize routes. The search is re-initialized from the current best TS solution z_{best} after the execution of the *Control* procedure (line 23). Moreover, after C_{cNS}consecutive executions of this procedure without improving the current best solution, a solution z is selected randomly (and removed) from the elite set, the value of $\overline{r}$ is reset to the value it had when the corresponding elite solution was found, the tabu lists are set to the empty state, and a Diversification mechanism is applied to perturb z (lines 17-19). The search then proceeds from the perturbed solution. The search is stopped when the elite set $\mathcal{E}$ is empty, and a post-optimization procedure is performed to potentially improve the current best solution z_{best}.

Algorithm 1. The Tabu Search meta-heuristic for the MTT-PDTWS

```
 1: Generate an initial feasible solution $z$
 2: $z_{best} \leftarrow z$; Elite set $\mathcal{E} \leftarrow \emptyset$; $STOP \leftarrow False$
 3: Probability of selecting routing neighbourhoods relative to leg neighbourhoods $\overline{r} \leftarrow 1$
 4: **repeat**
 5: Select a neighbourhood based on the value of $\overline{r}$;
 6: Find the best solution $z'$ in the selected neighbourhood of $z$;
 7: **if** $z'$ is better than $z_{best}$ **then**
 8:     $z_{best} \leftarrow z'$;
 9:     $\overline{r}_{best} \leftarrow \overline{r}$;
10:     Add $(z_{best}, \overline{r}_{best})$ to the elite set $\mathcal{E}$; update $\mathcal{E}$;
11:   **end if**
12:     $z \leftarrow z'$;
13:   **if** $z_{best}$ not improved for $IT_{cNS}$ iterations **then**
14:     **if** $z_{best}$ not improved for $C_{cNS}$ consecutive executions of *Control* procedure **then**
15:         **if** $\mathcal{E} = \emptyset$ , $STOP \leftarrow True$;
16:         **else**
17:             Select randomly $(z, \overline{r_z})$ (and remove it) from the elite set $\mathcal{E}$;
18:             Diversify the current solution $z$;
19:             Set $\overline{r} \leftarrow \overline{r_z}$ and reset tabu list;
20:         **end if**
21:     **else**
22:         Apply *Control* procedure to update the value of $\overline{r}$;
23:         $z \leftarrow z_{best}$;
24:     **end if**
25:   **end if**
26: **until** STOP;
27: $z_{best} \leftarrow Post-optimization(z_{best})$;
29: Return $z_{best}$.
```

Search space

We allow the search to explore infeasible solutions with respect to the vehicle capacity and the time windows of customer demands and supply points, infeasible solutions being penalized proportionally to the violations of these restrictions.

For a solution z, $c(z)$ stands for the total traveling cost, $q(z)$ for the vehicle-load violation, $w_c(z) = \sum_{i \in z} \max\{(a_i - l_i), 0\}$ and $w_s(z) = \sum_{s \in z} \max\{t(s) - \eta - a_s, a_s - t(s), 0\}$ for the time-window violations at customer demands and supply points, respectively (with a_i and a_s as the arrival times at customer demand i and supply point s, respectively). Let L_{io} be the set of pickup and delivery legs, L_{c2c} the set of c2c legs, and $L = (i_0, i_1, ..., i_k)$ a leg in solution z. The vehicle load violation for a leg is $q(L) = \max\{\sum_{i \in L'} q_i - Q, 0\}$, when $L \in L_{io}$, and $q(L) = \sum_{j=0}^{k} \max\{load(i_j) - Q, 0\}$, when leg $L \in L_{c2c}$, the vehicle load must be computed after node visit ($load(i_j) = load(i_{j-1}) + q_{i_j}, 1 \leq j \leq k$, with $load(i_0) = q_{i_0}$). The violation for solution z then is $q(z) = \sum_{L \in \{L_{c2c} \cup L_{io}\}} q(L)$.

The solution z is evaluated according to the weighted fitness function $f(z) = c(z) + \alpha^Q q(z) + \alpha^C w_c(z) + \alpha^S w_s(z) + F * m$, where m is the current number of vehicles used and $\alpha^Q, \alpha^C, \alpha^S$ are penalty parameters adjusted dynamically during the search (Cordeau *et al.*, 2001). The values of $\alpha^Q, \alpha^C, \alpha^S$ are modified at each iteration. Each is either multiplied by $1 + \beta > 1$, when the current solution is feasible with respect to the corresponding restriction, or it is divided by $1 + \beta$; otherwise

Initial solution

To obtain an initial solution, the supply points are sorted and indexed in increasing order of their opening times ($t(s_1) \leq t(s_2) \leftrightarrow s_1 < s_2$). We first assign each pickup-customer demand to a supply point by using the strategy in Nguyen *et al.* (2015), aiming to balance the pickup and delivery demands at supply points, thus increasing the possibility of "unload & load" activities, which then helps to reduce empty movements. Each work assignment of the initial solution is then built sequentially. First, the supply point s with earliest opening time and unserviced customer demands is assigned as the initial supply point of the first leg of the current work assignment. Next, one or a sequence of legs between s and either another supply point s' or the garage g is created using the greedy algorithm described below. If these leg(s) end at a supply point s', the greedy algorithm is applied again to build the next leg(s) with s' as the initial supply point. Otherwise (the first created leg(s) end at the garage), the current work assignment is completed, and a new one is initiated. If all pickup- and delivery-customer demands are serviced but unserviced c2c-customer demands still exist, the greedy algorithm is used to create new work assignment(s) servicing them. The process is repeated until all customer demands are serviced (assigned to a vehicle route).

The greedy algorithm builds legs for a given supply point s by first identifying the set of supply points $S' = \{s' \in S | s'$ with unserviced customer demands and $t(s') > t(s)\}$, and then proceeds for each pair (s, s') (when $S' \neq \emptyset$). Currently unassigned customer demands are candidates for insertion into the leg, which is attempted in the order 1) c2c-customer demands, 2) pickup-customer demands of s, 3) delivery-customer demands of s, 4) c2c-customer demands and 5) pickup-customer demands of s' (Figure 3). Time windows at supply points and customer demands, distances between them, and vehicle capacity restrict the generation.

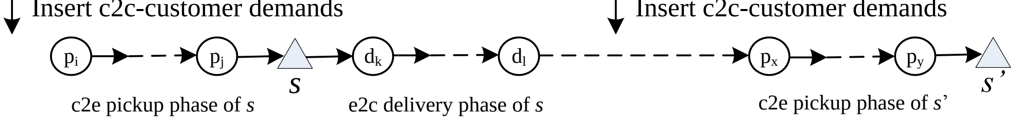

Figure 3. A generation of a sequence of legs between two supply points

Customer demands are assigned by applying the I1 heuristic of Solomon (1987) until the vehicle is full, the treatment of c2c requests ($\overline{p}, \overline{d}$) being more complicated as two customer demands $\overline{p}$ and $\overline{d}$, have to be inserted under the LIFO rule. More precisely, the feasible positions for the c2c-pickup-customer demand $\overline{p}$ are first computed, $\overline{p}$ is inserted at the first position found, and the best feasible insertion for $\overline{d}$ is determined using the heuristic

I1 of Solomon (1987). The process is reiterated with the second position found for $\overline{p}$, and so on until all feasible positions for $\overline{p}$ are tried out and the best insertion is identified.

The leg with the smallest average unit cost (ratio total traveling time over total demand carried by the vehicle between s and s; demand = 1 for empty legs) among the feasible legs generated between all pairs of (s, s') pairs is assigned to the current work assignment. When there are no feasible legs or $S' = \emptyset$, the greedy algorithm builds the last leg (s, g) by applying the heuristic I1 of Solomon (1987).

Neighbourhoods

A solution to the MTT-PDTWS is a set of work assignments, which consist of sequences of legs, which are sequences of customer demands. We thus define two types of neighbourhoods, *routing* to change the sequence of customer demands in at least one leg, and *leg* to change the sequence of legs in at least one work assignment. The (re)assignments of pickup-customer demands to supply points are performed within these neighbourhoods. Due to the limited space of the paper, we focus on the new developments addressing c2c demands.

Routing neighbourhoods execute different intra- and inter-route moves on customer demands. The *Relocation*, *Exchange* and *2-opt* neighbourhoods for pickup- and delivery-customer demands were defined in Nguyen *et al.* (2015). Two new ones are introduced for c2c-customer demands:

- *Relocation move* considered for each request $(\overline{p}, \overline{d})$ and c2c-customer demand $x \neq \overline{p}, \overline{d}$ (x is either a c2c-pickup- or a c2c-delivery-customer demand, belonging to the same or a different leg with respect to $(\overline{p}, \overline{d})$). We shift $\overline{p}$ from its current position to the position after x, and $\overline{d}$ is then relocated to a position in the same leg such that the LIFO constraint is satisfied and the fitness of the solution after the move is minimized.
- *Exchange move* concerns two requests $(\overline{p_1}, \overline{d_1})$ and $(\overline{p_2}, \overline{d_2})$. Let *pos*($i$) give the position of i in the leg. Three alternatives are considered: 1) swap the positions of $\overline{p_1}$ and $\overline{p_2}$, and of $\overline{d_1}$ and $\overline{d_2}$; 2) place $\overline{p_1}$ at pos($\overline{p_2}$) and $\overline{d_1}$ at pos($\overline{d_2}$), then place $(\overline{p_2}, \overline{d_2})$ at either pos($\overline{p_1}$) or pos($\overline{d_1}$); 3) place $\overline{p_2}$ at pos($\overline{p_1}$) and $\overline{d_2}$ at pos($\overline{d_1}$), then place $(\overline{p_1}, \overline{d_1})$ at either pos($\overline{p_2}$) or pos($\overline{d_2}$).

Two *leg neighbourhoods* focus on repositioning supply points, and the legs assigned to them, between work assignments: *Relocate supply point* to move a supply point, and the legs and customer demands assigned to it, from their current work assignment to another work assignment, and *Exchange supply point* to swap two supply points, and the customer demands assigned to them between two work assignments. Nguyen *et al.* (2015) details the procedures for pickup and delivery legs, which are assigned to the supply points where the vehicle returns the collected freight and loads new freight, respectively.

Two possibilities are evaluated in the MTT-PDTWS for each c2c leg, enriching the neighbourhoods: consider it for repositioning together with the supply points or not. In the former case, one must determine to which supply point to assign the c2c leg. Figures 4 and 5 depict all possible positions of a c2c leg within a work assignment. When in first (Figure 4(a)) or last position (Figure 4(b)) of a work assignment, it is assigned to the first (last) supply point s of the work assignment. Otherwise, it is in a middle position between two supply points s and s′ (Figure 5) and assignments to both are evaluated.

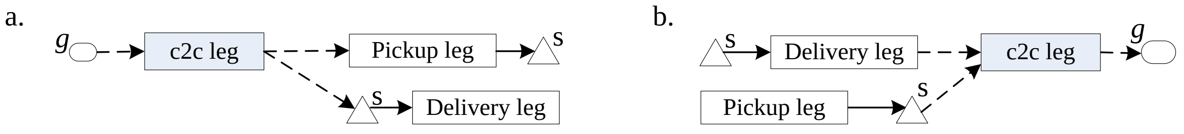

Figure 4 (a) First position c2c leg; (b) Last position c2c leg

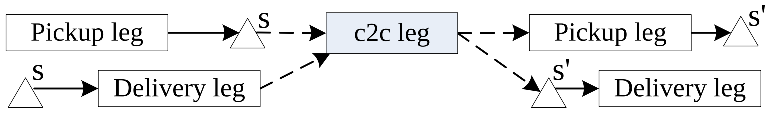

Figure 5. Middle position c2c leg

The tabu search meta-heuristic explores the solution space of MTT-PDTWS selecting probabilistically, at each iteration, one of the ten neighbourhoods described previously. Routing and leg neighbourhoods receive the selection probabilities $\overline{r}/(2+8\overline{r})$ and $1/(2+8\overline{r})$, respectively. At the beginning of the search, leg and routing neighbourhoods receive the same selection probability (setting $\overline{r}=1$), to allow the algorithm to freely explore the solution space. Given that the number of supply points is much smaller than the number of customer demands in most MTT-PDTWS instances, the algorithm should perform more customer than leg moves to ensure adequate optimization of routes. Consequently, after the initial phase, the probability of selecting leg neighbourhoods becomes gradually lower than the probability of selecting routing moves. The neighbourhood selection probability is controlled by the parameter $\overline{r}$. The *Control* procedure varies the value of $\overline{r}$ during execution to monotonically reduce (increase) the probability of selecting leg (routing) neighbourhoods after each IT_{cNS} iterations without improvement in the best solution. A linear scheme $\overline{r}_{new} = \overline{r}_{last} + \Delta\overline{r}$ is used, where $\Delta\overline{r}$ is user defined.

We use five tabu lists, one for each type of routing and leg move. A tabu status is assigned to each tabu list element forbidding for θ iterations to change its new position, unless it would improve the current best solution (aspiration criterion). The tabu duration θ is randomly selected (uniform distribution) from a particular interval.

There are $O(m'|S|)$ possible leg moves. The interval for leg moves is then set to $[m'|S|/a_1,\ m'|S|/a_2]$, where m' is the number of vehicles used in the initial solution, and $a_1 > a_2$ are user-defined parameters. The intervals for routing moves are specific for delivery-customer demands, $[a_3\log_{10}|C_s^D|,\ a_4\log_{10}|C_s^D|]$, pickup-customer demands, $[a_5\log_{10}(|C^P|),\ a_6\log_{10}(|C^P|)]$, and c2c-customer demands, $[a_7\log_{10}(|C^{c2c}|),\ a_8\log_{10}(|C^{c2c}|)]$, where $a_3<a_4$, $a_5<a_6$, $a_7<a_8$ are user defined parameters. The number of iterations during which a delivery-customer demand routing move within the delivery zone of a supply point s remains tabu is only counted each time the algorithm deals with delivery-customer demands in that zone.

Diversification strategy

Diversification is triggered after a number of iterations without improvement in the current best solution, to direct the search to potentially unexplored promising regions. The procedure uses an elite set and a frequency-based memory.

The elite set is 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 z_{best} produced by the search and the elimination of solutions in the elite set. The elimination is based on the Hamming distance computed according to Equation (1), where $T(cond)$ is a valuation function that returns 1 if the condition *cond* is true, 0, otherwise; $N_z[i]$ is the next place (i.e., a customer demand, the garage, or a supply point) visited by the vehicle after servicing customer demand i in solution z; and $S_z[i]$ is the supply point assigned to pickup-customer demand i in solution z. The solution z most similar to the newly inserted z_{best}, i.e., the one with the smallest $\Delta(z, z_{best})$, is eliminated (when latter is not full, elimination occurs only when the similarity is very strong, $\Delta(z, z_{best}) \leq 0.05(|C^D| + 2|C^P| + |C^P_{c2c}| + |C^D_{c2c}| + |S|)$.

$$\Delta(z_1, z_2) = \sum_{i \in C^P \cup C^D \cup C^P_{c2c} \cup C^D_{c2c}} T\big(N_{z_1}[i] \neq N_{z_2}[i]\big) + \sum_{i \in C^P} T(S_{z_1}[i] \neq S_{z_2}[i]) \qquad (1)$$

The long-term frequency memory keeps a history of the arcs most frequently added to the current solution as well as the supply-point assignments of pickup-customer demands most frequently used. 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 and promoting never-seen supply-point assignments. The corresponding penalties $g_1(\overline{z})$ and $g_2(\overline{z})$ that are added to the evaluation of the fitness $f(\overline{z})$ of a neighbor $\overline{z}$ of the current solution z are:

$$g_1(\overline{z}) = \overline{C}\Big(\sum_{(i,j) \in A_a} \rho_{ij} + \sum_{(i',j') \in A_r} (1 - \rho_{i'j'})\Big) \qquad (2)$$

$$g_2(\overline{z}) = \overline{C} \sum_{p \in C^P} \left(\sum_{\substack{s \in S_p \\ S_z(p) = S_{\overline{z}}(p) = s}} \chi_{ps} + \sum_{\substack{s \in S_p \\ S_z(p) \neq s \\ S_{\overline{z}}(p) = s}} \chi_{ps} + \sum_{\substack{s \in S_p \\ S_z(p) = s \\ S_{\overline{z}}(p) \neq s}} (1 - \chi_{ps}) \right) \qquad (3)$$

where $\overline{C}$ is the average cost of all arcs in the problem, ρ_{ij} is the frequency arc (i, j) has been added to the solution, χ_{ps} is the frequency pickup-customer demand p has been assigned to supply point s during the search,, and A_a and A_r are the sets of arcs that are added to and removed from the solution z in the move to $\overline{z}$, respectively.The diversification mechanism is executed IT_{div} iterations.

Post optimization

The best solution obtained through the tabu search is enhanced by applying a local search *Supply-point-improvement* procedure followed by a *Leg-improvement procedure*. The former assigns a new supply point to each pickup-customer demand, keeping those that actually improve the solution. The latter aims to improve routing through inter- and intra-route moves applied to legs of the solution. For pickup and delivery legs, two intra-route operators, the 2-opt of Lin (1965) and the Or-opt of Or (1976), and two inter-route operators, the λ-interchange of Osman (1993) where $\lambda = \{1, 2\}$, and the CROSS-exchange of Taillard *et al.* (1997), are used. More details could be found in Nguyen *et al.* (2015). For c2c legs, four operators are used either within one leg or between two legs:

- *Request relocate* moves a request $(\overline{p}, \overline{d})$. First, $\overline{p}$ and $\overline{d}$ are removed from their current positions. Next, feasible positions for $\overline{p}$ are computed. $\overline{p}$ is then inserted at the first position found, and the best feasible insertion for $\overline{d}$ is determined. The process is repeated with the second position found for $\overline{p}$, and so on until all feasible positions for $\overline{p}$ have been tried out. The best relocation is executed if it improves the solution.

- *Request exchange* for two requests $(\overline{p_1}, \overline{d_1})$ and $(\overline{p_2}, \overline{d_2})$. It works as described in the *Exchange move* in Section Neighbourhoods. All relocations of the two requests are considered and the best one is implemented if it improves the solution.
- *Multiple request relocate*. For each request $(\overline{p}, \overline{d})$, it moves the sequence of requests starting at $\overline{p}$ and ending at $\overline{d}$ (Figure 6), and executes the best relocation if it improves the solution.
- *Multiple request exchange*. Exchanges, for each pair $(\overline{p_1}, \overline{d_1})$ and $(\overline{p_2}, \overline{d_2})$, two sequences of requests, one starting at $\overline{p_1}$ and ending at $\overline{d_1}$, the other starting at $\overline{p_2}$ and ending at $\overline{d_2}$. When the two requests belong to the same c2c leg, they do not overlap (Figure 7).

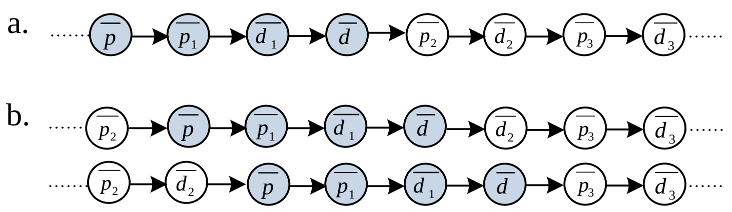

Figure 6. (a) Initial leg; (b) New legs created by *multiple request relocate* on $(\overline{p},\overline{d})$

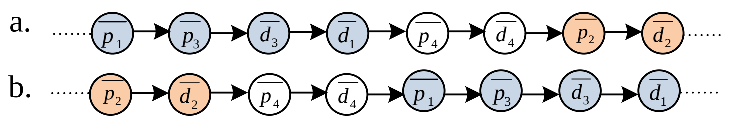

Figure 7. (a) Initial leg; (b) New leg from *multiple request exchange* on $(\overline{p_1}, \overline{d_1})$ and $(\overline{p_2}, \overline{d_2})$

The Leg improvement procedure working on c2c legs starts by applying in random order the four above operators. Each neighbourhood is searched on all possible pairs of legs (in random order) and stopped on the first feasible 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 each work assignment in turn. The procedure repeats until no more improvement is found.

EXPERIMENTS

We study the impact of a number of major problem characteristics and search strategies on the quality of the meta-heuristic, the behaviour of the solutions, and the management of the CL system.

The tabu search meta-heuristic is implemented in C++. Experiments were run on a 2.8GHz Intel Xeon 4-core processor with 16GB of RAM. The user-defined parameters $(\alpha^Q, \alpha^C, \alpha^S, \beta, a_1, a_2, a_3, a_4, a_5, a_6, a_7, a_8, \Delta\overline{r}, IT_{div}, IT_{CNS})$ were set to (1, 1, 1, 0.3, 7, 5, 6, 8, 7, 10, 5, 7, $5\log_{10}(n/|S|$, $m'|S| + n$, $3m'|S| + n$), where m' and n are the numbers of vehicles used in the initial solution and of customer demands, respectively.

Instances

The instance set consists of the two sets A1 and A2, with 15 instances each, proposed by Nguyen *et al.* (2015) for the MT-PDTWS, to which we add $(|C^P| + |C^D|)/3$ c2c requests. Table 1 summarizes the characteristics of these instances.

To generate c2c requests, we consider the supply points s_1, s_2, ..., $s|s|$ in increasing order of opening times. Then, for each request $(\overline{p}, \overline{d})$, we select randomly two nodes x_1, x_2 in set $X = \{g, s \in S\}$, and define the period $[E_{\overline{d}}, L_{\overline{d}}]$ during which the request must be serviced:

- If $x_1 = g$, the request is serviced before the opening time of the first supply point s_1; we thus set $E_{\overline{d}} = t(s_1) - \Delta$ and $L_{\overline{d}} = t(s_1)$;
- If two supply points s_a and s_b are selected and $t(s_a) < t(s_b)$, the request is to be serviced in between the opening times of these supply points; thus $E_{\overline{d}} = t(s_a) - \delta(\overline{d}) - \lceil c_{\overline{d}s_a} \rceil$ and $L_{\overline{d}} = t(s_b) - \delta(\overline{d}) - \lceil c_{\overline{d}s_b} \rceil$; if $t(s_a) = t(s_b)$, service will occur before the opening time of s_a, thus $E_{\overline{d}} = t(s_a) - \Delta$ and $L_{\overline{d}} = t(s_a)$;
- If $x_2 = g$, vehicles service the request after the opening time of the last supply point $s_{|S|}$, thus $E_{\overline{p}} = t(s_{|S|})$ and $L_{\overline{p}} = E_{\overline{p}} + \Delta$.

Table 1. Characteristics of the MTT-PDTWS instances

Test set	$\lvert S \rvert$	$\lvert W \rvert$	Customer demand $i \in C^D \cup C^P \cup C^P_{c2c} \cup C^D_{c2c}$					
			$\lvert C^D \rvert$	$\lvert C^P \rvert$	$\lvert \mathcal{R} \rvert$	[X,Y] coordinates	q_i	$\delta(i)$
A1	4	4	400	44,171,400	150,190,270	[0,100]	[5,25]	20
A2	8	4						

The time windows for the request are then created within $[E_{\overline{d}}, L_{\overline{d}}]$ to ensure feasibility. In the first two cases, the values of $e_{\overline{d}}$ and $l_{\overline{d}}$ are chosen randomly in the intervals $[E_{\overline{d}} - \Delta, E_{\overline{d}}]$ and $[L_{\overline{d}} - \Delta, L_{\overline{d}}]$, respectively. The values of $e_{\overline{p}}$ and $l_{\overline{p}}$ are then randomly selected in the intervals $[E_{\overline{p}} - \Delta, E_{\overline{p}}]$ and $[L_{\overline{p}} - \Delta, L_{\overline{p}}]$, respectively, where $E_{\overline{p}} = e_{\overline{d}} - \delta(\overline{p}) - \lceil c_{\overline{p}\overline{d}} \rceil$ and $L_{\overline{p}} = l_{\overline{d}} - \delta(\overline{p}) - \lceil c_{\overline{p}\overline{d}} \rceil$. The procedure is reversed for the last case. The values of $e_{\overline{p}}$ and $l_{\overline{p}}$ are first randomly selected in $[E_{\overline{p}} - \Delta, E_{\overline{p}}]$ and $[L_{\overline{p}} - \Delta, L_{\overline{p}}]$, respectively. The value of $l_{\overline{d}}$ is then chosen randomly in $[L_{\overline{d}}, L_{\overline{d}} + \Delta]$, where $L_{\overline{d}} = l_{\overline{p}} + \delta(\overline{p}) + \lceil c_{\overline{p}\overline{d}} \rceil$, and $e_{\overline{d}}$ is selected in the interval $[L_{\overline{d}} - \Delta, L_{\overline{d}}]$. The value of Δ is set to 300.

Elite set calibration and diversification

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 z at random and re-start the algorithm from it. The diversification mechanism is applied in the last two variants, the alternatives being defined by the initialization of the $\overline{r}$ parameter, which was set to either the full or half the value at which z was found, respectively (i.e., $\overline{r} = \overline{r}_z$ or $\overline{r} = \overline{r}_z/2$). Table 2 displays the performance comparison between the four variants with the three different values for the elite set size (1, 5, and 10). For each variant and elite-set size, the table displays the average gap of the cost of its best solution relative to the best solution of the case without the elite set and diversification, together with the corresponding average computation time in minutes over 10 runs.

As expected, results indicate that guidance using elite solutions and diversification using long-term memory contribute significantly to improve the performance of the algorithm. Without them, the algorithm requires the lowest computation effort but produces the worst solutions compared to other variants. Comparing the two variants corresponding to the two values at which $\overline{r}$ is reset, one observes that the solution quality is not very sensitive to this value, but computing effort is increasing when the value of $\overline{r}$ is lower ($\overline{r} = \overline{r}_z/2$). 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 61% more time. We therefore set the size of the elite set to 5 and reset $\overline{r} = \overline{r}_z$.

Table 2. Performance comparison between diversification settings

Elite set size	Without diversification				With diversification			
	1st variant		2nd variant		3rd variant		4th variant	
	$\overline{r} = \overline{r}_z$		$\overline{r} = \overline{r}_z/2$		$\overline{r} = \overline{r}_z$		$\overline{r} = \overline{r}_z/2$	
	GAP (%)	Time	GAP (%)	Time	GAP (%)	Time	GAP (%)	Time
0	0	38						
1	-0.47	46	-0.49	67	-1.16	62	-1.18	87
5	-0.68	71	-0.64	86	-1.77	94	-1.71	126
10	-0.82	106	-0.78	134	-1.79	152	-1.74	197

Numerical results and analyses

Table 3 displays the results obtained by the proposed tabu search meta-heuristic over 10 runs for each group of instances: average cost (*Avg10* column), best solution cost (*Best10*), number of vehicles (*#Veh*), % of times vehicles move directly to supply points without using waiting stations (*DM(%)*), % of times vehicles perform both unload and load once they arrive at supply points (*PD(%)*), % of c2c-customer demands serviced by work assignments that also containing pickup- or delivery-customer demands (*c2cInter(%)*), and average CPU times in minutes (*Time*).

Table 3. Performance of TS on all instances

Test set	Avg10	Best10	#Veh	DM(%)	PD(%)	c2cInter(%)	Time
A1	31774.3	31524.03	26	45.65	38.12	92.75	115
A2	29073.1	28895.85	21	48.20	42.03	96.09	72
Average	30423.7	30209.94	24	46.95	40.08	94.42	94

The experimental results in Table 3 show that the proposed meta-heuristic performs well and achieves the objective of integrating the multiple types of demands. The small differences in the values of the average and best solutions indicate the method is stable even though it includes a number of random choices. They also show that 94.42% of c2c-customer demands can be served together with pickup- and delivery-customer demands by the same vehicles. Overall, 705 vehicles are used in the 30 problem instances, operating a total of 3,881 legs, each vehicle servicing 6 legs on average, each leg consisting of 7 customer demands. Moreover, operations at supply points are balanced, vehicles performing both unload and load operations 40.08% of the times on average. This contributes to reduce not only the number of empty moves but the travel cost as well.

The benefits of combining three types of transport demand

To evaluate the value of integration, of servicing several types of travel demands with the same vehicles, we evaluated three policies: 1) e2c&c2e&c2c, all three types of customer demands serviced by the same vehicles; 2) (e2c&c2e) + c2c, pickup and delivery customer demands serviced by the same vehicles and c2c-customer demands by different vehicles; 3) e2c+c2e+c2c, disjoint service, each type of customer demand being serviced by its own set of vehicles. Table 4 compares the best solutions of these policies, displaying average results for all instances over 10 runs. The average number of vehicles, travelling cost, and total cost of the integrated policy are given in columns *#Veh*, *Travel cost*, and *Total cost*, respectively. Three values are displayed for each of the two other policies, giving the gaps for the same three measures relative to those of e2c&c2e&c2c.

Table 4. Comparison of disjoint and combined traffic types

Test	e2c&c2e&c2c			(e2c&c2e) + c2c			e2c+c2e+c2c		
set	#Veh	Travel cost	Total cost	GAP (%)			GAP (%)		
A1	26	18557.36	31524.03	30.00	0.63	12.85	73.07	13.28	37.21
A2	21	18362.51	28895.85	30.50	7.15	15.61	61.90	20.92	35.68
Average	24	18459.94	30209.94	30.25	3.89	14.23	67.49	17.10	36.45

Not surprisingly, the figures in Table 4 indicate that the more one integrates the service of different demand types, the larger the gain in performance, and the gains are impressive. Integration reduces empty movements and waiting times, and allows vehicles to perform more “unload & load” operations at supply points (i.e., increases the synchronization at supply points). As a result, the e2c&c2e&c2c strategy provides the best solutions in terms of both the number of vehicles and the travel cost. It is noticeable that the average work assignment composition for this strategy is 6 legs, on average, as compared to 5 for the e2c+c2e case and only 4 when e2c and c2e customers are serviced separately. The intermediary strategy, where only c2c customers are handled disjointly, already shows significant deterioration in performance, particularly relative to the waiting time and costs. Without integration, the e2c+c2e+c2c strategy gives the worst solutions, with an average increase in the total cost of 36.45%.

Impact of time windows at c2c-customer demands

Each c2c-customer demand i in the MTT-PDTWS has a hard time window $[e_i, l_i]$, i.e., the vehicle may arrive before e_i and wait to begin service, but must not arrive later than l_i. We analyse the impact on solution quality of the time windows for c2c- customer demands, $(\overline{p}, \overline{d})$, by comparing cases with time windows and settings with due-times only.

Four variants of the problem were thus studied corresponding to the existence or not of the earliest start-of-service times at $\overline{p}$ and $\overline{d}$. Table 5 sums up the solution-quality variations for the base case with time windows for both $\overline{p}$ and $\overline{d}$ (the double [sTW, eTW]), to three cases where the earliest start-of-service time does not exist for $\overline{p}$, $\overline{d}$ or both (single or double eTW column heading). The table displays the solution-quality variations in terms of the number of vehicles, travel cost, and total cost (average % over all instances) with respect to the base case. The percentage number of c2c-customer demands being serviced by work assignments including also pickup- or delivery-customer demands (*c2cInter (%)* column) is also given.

Results indicate that solutions with flexible (longer) service-period requirements at both customers making up the request are better than the cases when only one customer is flexible, the latter being better than the case when neither is flexible. For the most flexible case, all c2c-customer demands could be integrated with pickup- and delivery-customer demands in the same work assignments, eliminating the need for additional vehicles for their service. As even in the most constrained case the c2cInter percentage is very high, the gain in efficiency would have to be measured against the cost of customer flexibility.

Table 5. Impact of time windows at c2c-customer demands on solution quality

	c2c-pickup customer demands			
	[sTW, eTW]		eTW	
c2c-delivery customer demands	[sTW, eTW]	eTW	[sTW, eTW]	eTW
#Vehicles (%)	0	-1.23	-1.91	-7.49
Travel cost (%)	0	-0.13	-1.38	-19.1
Total cost (%)	0	-0.51	-1.56	-15.19
c2cInter (%)	94.42	94.62	96.22	100

Synchronization at supply points

In all previous experiments, the time period availability at each supply point s was characterized by one time window only $[e_s, l_s]$, used for both unloading and loading operations. In order to analyse the impact of the availability requirements without modifying the time windows at customer demands, we introduce into the model two time windows at each supply point, one for unloading and one for loading, but keep the total availability time unchanged. More precisely, we use $[e_s^u, l_s^u]$ and $[e_s^l, l_s^l]$, specifying the earliest and latest times at which a vehicle has to be available at s for unloading collected demands and loading delivery demands, respectively, where $l_s^u + \varphi'(s) \leq l_s^l$, $e_s^u = e_s$ and $l_s^l = l_s$. These time windows are defined by two parameters: the *length* of each unload and load time window (noted len_u and len_l, respectively; $len_u = len_l = L$ in our experiment), and the *distance D* between e_s^l and l_s^u. The total availability time at supply points was kept to the length of the time window (100 in all instances) in the base case (single time window), and we performed three runs with (L, D) equal to (20, 60), (30, 40) and (40,20), respectively.

Table 6 sums up the solution-quality variations (averages for all instances) for the three cases of two time windows compared to the base case. The table displays the increase (in %) in the values of the number of vehicles, travel cost, and total cost of each of the two-window variants relative to the single-window one. The % of times the vehicles both unload and load at supply points (*PD(%)* row) and the % of times vehicles move directly to a supply point without using waiting stations (*DM(%)*) are given for all variants.

Table 6. Impact of synchronization at supply points on solution quality

	One time window	Two time windows		
	L = 100, D = 0	L = 20, D = 60	L = 30, D = 40	L = 40, D = 20
#Vehicles (%)	0	1.24	0.87	0.63
Travel cost (%)	0	2.62	1.04	0.87
Total cost (%)	0	2.97	1.83	1.66
PD (%)	40.08	33.64	35.27	36.13
DM (%)	46.95	42.51	42.96	44.02

Results indicate that solutions with two time windows are worse than those with one time window with respect to all performance measures. Thus, separating in time the unloading and loading operations at satellite facilities is detrimental to system performance, even when the distance is small (variant (40,20)), the performance deterioration increasing with the increase in distance and the decrease in time window length. Moreover, longer activity and waiting-time capabilities at supply points result in increased system performances, vehicles moving directly to supply points more frequently and performing more "unload & load" operations (maximum increase of 46.95% and 40.08% respectively, relative to the base case).

CONCLUSION

We have addressed the issue of considering simultaneously, with a single fleet, the routing requirements of the three main types of transport demand, customer-to-customer with origin and destination at customers located within the City Logistics-controlled part of the city, the customer-to-external zone from the city centre to destinations outside the city limits, and the "classical" external zone-to-customer inbound demand.

We proposed the Multi-trip Multi-traffic Pickup and Delivery Problem with Time Windows and Synchronization to address the issue. The MTT-PDTWS is a new pickup and delivery vehicle routing variant where vehicles perform routes made up of multiple sequences of visits at facilities and customers at given time moments, modelled through supply points and customer demands, respectively, under strict time synchronization restrictions. We also proposed a tabu search meta-heuristic for this problem, integrating multiple neighbourhoods targeted to the decision sets of the problem and the different pickup and delivery sequences involved.

The computational study performed on a set of benchmark instances with up to 8 supply points and 1340 customer demands, indicates that the proposed methodology achieves the goals of efficient integration of the three types of customer service. They also numerically qualify the interest of integration and of carefully designing and negotiating the time restrictions at facilities and customers.

ACKNOWLEDGEMENTS

The authors thank the Discovery Grant and the Discovery Accelerator Supplements Program of the Natural Science and Engineering Research Council of Canada, and the Strategic Clusters program of the Fonds québécois de la recherche sur la nature et les technologies for supporting this research.

REFERENCES

Bettinelli, A., Crainic, T.G., and Vigo, D. (2015), *An Exact Approach for a Multi-Zone Multi-Trip Vehicle Routing Problem with Separate Delivery and Collection*. Publication CIRRELT, Centre interuniversitaire de recherche sur les réseaux d'entreprise, la logistique et le transport, Université de Montréal, Montréal, QC, Canada. forthcoming.

Brandão, J. (2006), A new tabu search algorithm for the Vehicle Routing Problem with backhauls. *European Journal of Operational Research*, 173(2), 540-555.

Carrabs, F., Cordeau, J.-F., Laporte, G. (2007), Variable neighborhood search for the pickup and delivery traveling salesman problem with LIFO loading. *INFORMS Journal on Computing* 19(4), 618–632.

Cheang, B., Gao, X., Lim, A., Qin, H., Zhu, W. (2012), Multiple pickup and delivery traveling salesman problem with last-in-first-out loading and distance constraints. *European Journal of Operational Research* 223(1), 60–75.

Cherkesly, M., Desaulniers, G., Laporte, G. (2014). Branch-price-and-cut algorithms for the pickup and delivery problem with time windows and last-in-first-out loading. *Transportation Science*. forthcoming.

Cordeau, J.-F., Laporte, G., and Mercier, A. (2001), 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., and Storchi, G. (2009), Models for Evaluating and Planning City Logistics Systems. *Transportation Science* 43(4), 432-454.

Crainic, T.G., Errico, F., Rei, W., and Ricciardi, N. (2012), Integrating c2e and c2c Traffic into City Logistics Planning, *Procedia - Social and Behavioral Sciences* 39: 47-60.

Gélinas, S., Desrochers, M., Desrosiers, J., and Solomon, M. (1995), A new branching strategy for time constrained routing problems with application to backhauling. *Annals of Operations Research*, 61(1), 91-109.

Li, Y., Lim, A., Oon, W.-C., Qin, H., Tu, D., (2011), The tree representation for the pickupand delivery traveling salesman problem with LIFO loading. *European Journal of Operational Research* 212(3), 482–496.

Liao, C.-J, Lin, Y. and Shih, S.C. (2010), Vehicle routing with cross-docking in the supply chain. *Expert Systems with Applications*, 37(10), 6868-6873.

Lin, S. (1965), Computer solutions of the traveling salesman problem. *Bell System Technical Journal*, 44:2245-2269.

Nguyen, P.K., Crainic, T.G. and Toulouse, M. (2013), A Tabu Search for Time-dependent Multi-zone Multi-trip Vehicle Routing Problem with Time Windows, *European Journal of Operational Research* 231(1):43-56.

Nguyen, P.K., Crainic, T.G. and Toulouse, M. (2015), *Multi-trip Pickup and Delivery Problem with Time Windows and Synchronization*, Publication CIRRELT, Centre interuniversitaire de recherche sur les réseaux d'entreprise, la logistique et le transport, Université de Montréal, Montréal, Canada. forthcoming.

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. (2993), Metastrategy simulated annealing and tabu search for the vehicle routing problem, *Annals of Operations Research*, 41, 421-451.

Solomon M.M (1987), Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations Research*, 35:254–265.

Taillard, E.D., Badeau, P., Gendreau, M., Guertin, F., and Potvin, J.-Y. (1997), A tabu search heuristic for the Vehicle Routing Problem with soft time windows. *Transportation Science*, 31:170-186.

Thangiah, S.R., Potvin, J.-Y., and Sun, T. (1996), Heuristic approaches to vehicle routing with backhauls and time windows. *Computers & Operations Research*, 23(11):1043-1057.

Vidal, T., Crainic, T.G., Gendreau, M., and Prins, C. (2014), A unified solution framework for multi-attribute vehicle routing problems, *European Journal of Operational Research*, 234(3), 658-673.

Wen, M., Larsen, J., Clausen, J., Cordeau, J.-F., and Laporte, G. (2008) Vehicle Routing with Cross-Docking. *Journal of the Operational Research Society*, 60:1708-1718.

Annex - Model Formulation

The MTT-PDTWS is defined on a space-time network $\mathcal{G} = (\mathcal{V}, \mathcal{A})$, where $\mathcal{V}$ is the set of nodes, and the arcs in $\mathcal{A}$ stand for the possible movements between these nodes. Set $\mathcal{V}$ is made up of the garage (main depot) g and the sets of customer demands, supply points and waiting stations, i.e., $\mathcal{V} = \{g \cup \mathcal{C}^P \cup \mathcal{C}^D \cup \mathcal{C}^P_{c2c} \cup \mathcal{C}^D_{c2c} \cup \mathcal{S} \cup \mathcal{W}\}$. The set of arcs $\mathcal{A}$ can be described by the following types of arcs (i, j):

- From the garage g to
  - supply point $s \in \mathcal{S}$;
  - pickup-customer demand $p \in \mathcal{C}^P$;
  - internal-pickup-customer demand $\bar{p} \in \mathcal{C}^P_{c2c}$;
- From each supply point $s \in \mathcal{S}$ to
  - its delivery-customer demand $d \in \mathcal{C}^D_s$;
  - waiting station $w \in \mathcal{W}$;
  - another supply point $s' \in \mathcal{S}$ such that $t(s') - \eta \leq t(s) - \eta + \varphi'(s) + c_{ss'} \leq t(s)$ (i.e., the vehicle could travel directly from s to s' when it does "unload only" at s)
  - (c2c-)-pickup-customer demand $p \in \{\mathcal{C}^P \cup \mathcal{C}^P_{c2c}\}$ such that $t(s) - \eta + \varphi'(s) + c_{sp} \leq l_p$ (i.e., the vehicle could arrive at p from s before the due time l_p when it does "unload only" at s) ;
- From each waiting station $w \in \mathcal{W}$ to supply point $s \in \mathcal{S}$;
- From each pickup-customer demand $p \in \mathcal{C}^P$ to
  - one of its available supply points $s \in \mathcal{S}_p$;
  - another pickup-customer-demand $p' \in \mathcal{C}^P$ such that (1) $S_p \cap S_{p'} \neq \emptyset$ (i.e., p' has at least one available supply point in common with p), and (2) $e_p + \delta(p) + c_{pp'} \leq l_{p'}$;
  - waiting station $w \in \mathcal{W}$;
- From each delivery-customer demand $d \in \mathcal{C}^D_s$, $s \in \mathcal{S}$ to
  - another delivery-customer-demand $d' \in \mathcal{C}^D_s$ such that $e_d + \delta(d) + c_{dd'} \leq l_{d'}$;
  - waiting station $w \in \mathcal{W}$;
  - another supply point $s' \in \mathcal{S}$ such that $e_d + \delta(d) + c_{ds'} \leq t(s')$;
  - (c2c-)pickup-customer demand $p \in \{\mathcal{C}^P \cup \mathcal{C}^P_{c2c}\}$ such that $e_d + \delta(d) + c_{dp} \leq l_p$;

- From each c2c-pickup-customer demand $\bar{p} \in \mathcal{C}^P_{c2c}$ to
  - its corresponding c2c-delivery-customer demand $\bar{d}$;
  - another c2c-pickup-customer demand $\bar{p}^{'} \in \mathcal{C}^P_{c2c}$ such that $e_{\bar{p}} + \delta(\bar{p}) + c_{\bar{p}\bar{p}'} \leq l_{\bar{p}'}$;
- From each c2c-delivery-customer demand $\bar{d} \in \mathcal{C}^D_{c2c}$ to another node except its corresponding c2c-pickup-customer demand $\bar{p}$ and delivery customer demands $d \in \mathcal{C}^D$, such that $e_{\bar{d}} + \delta(\bar{d}) + c_{\bar{d}i} \leq l_i$ if $i \in \{\mathcal{C}^P \cup \mathcal{C}^P_{c2c} \cup \mathcal{C}^D_{c2c}\}$, and $e_{\bar{d}} + \delta(\bar{d}) + c_{\bar{d}s} \leq t(s)$ for $s \in \mathcal{S}$.

Let F stand for the fixed cost for operating a vehicle. The set of available vehicles is denoted by $\mathcal{K}$. The maximal number of arcs included in any work assignment is given by e and we define $\mathcal{U}$ as $\{1,...,e\}$. Let Q^{min}_d and Q^{min}_p be the minimal demand of delivery-customer demands and of pickup-customer demands, respectively. M is a large positive constant.

We define the following decision variables:

- x^u_{ijk}, a binary variable that takes value 1 if arc $(i,j) \in \mathcal{A}$ is traversed by vehicle k and appears in the u^{th} position of the work assignment of vehicle k, and 0 otherwise;
- y_{ps}, a binary variable that takes value 1 if pickup-customer demand $p \in \mathcal{C}^P$ is assigned to supply point $s \in \mathcal{S}$, and 0 otherwise;
- z_{sk}, a binary variable that takes value 1 if vehicle k unloads at supply point s, and 0 otherwise.

Note that we preliminary set $y_{ps} = 0$, $\forall p \in \mathcal{C}^P, s \notin S_p$, given that such s does not service p. Demands at each supply point $s \in \mathcal{S}$, waiting station $w \in \mathcal{W}$, and the garage g are equal to zero, i.e, $q_s = q_w = q_g = 0$. For convenience, we set the demand at each delivery node $d \in \mathcal{C}^D$: $q_d = -q_d < 0$. In addition,

B_{ik}	Starting time of service at customer demand $i \in \{\mathcal{C}^P \cup \mathcal{C}^D \cup \mathcal{C}^P_{c2c} \cup \mathcal{C}^D_{c2c}\}$ by vehicle k;
B_{sk}	Arrival time of vehicle k at supply point $s \in \mathcal{S}$;
B_{iwk}	Arrival time of vehicle k at waiting station $w \in \mathcal{W}$ from a supply point or a customer demand $i \in \{\mathcal{S} \cup \mathcal{C}^P \cup \mathcal{C}^D \cup \mathcal{C}^P_{c2c} \cup \mathcal{C}^D_{c2c}\}$;
Q_{ik}	Load of vehicle k when leaving $i \in \mathcal{V}$;
L_{sk}	Load of vehicle k when arriving at supply point $s \in \mathcal{S}$.

We set $Q_{gk} = 0 \; \forall k \in \mathcal{K}$ as the vehicle leaves the depot empty. The MTT-PDTWS can then be formulated as the following:

$$\text{Minimize} \sum_{k \in \mathcal{K}} \sum_{(i,j) \in \mathcal{A}} c_{ij} \sum_{u \in \mathcal{U}} x^u_{ijk} + F \sum_{k \in \mathcal{K}} \left(\sum_{s \in \mathcal{S}} x^1_{gsk} + \sum_{p \in \{\mathcal{C}^P \cup \mathcal{C}^P_{c2c}\}} x^1_{gpk} \right) \tag{1}$$

Subject to

$$\sum_{k \in \mathcal{K}} \sum_{u \in \mathcal{U}} \sum_{j \in \mathcal{V}} x^u_{ijk} = 1 \quad \forall i \in \{\mathcal{C}^P \cup \mathcal{C}^D \cup \mathcal{C}^P_{c2c} \cup \mathcal{C}^D_{c2c}\} \tag{2}$$

$$\sum_{u \in \mathcal{U}} \sum_{j \in \mathcal{V}} x^u_{sjk} \leq 1 \quad \forall s \in \mathcal{S}, k \in \mathcal{K} \tag{3}$$

$$\sum_{u \in \mathcal{U}} \sum_{j \in \mathcal{V}} x^u_{jik} = \sum_{u \in \mathcal{U}} \sum_{j \in \mathcal{V}} x^u_{ijk} \quad \forall i \in \{\mathcal{V} \setminus g\}, k \in \mathcal{K} \tag{4}$$

$$\sum_{i \in \{\mathcal{V} \setminus g\}} x^1_{gik} = \sum_{u \in \mathcal{U}} \sum_{\{i \in \mathcal{V} \setminus g\}} x^u_{igk} \quad \forall k \in \mathcal{K} \tag{5}$$

$$\sum_{s \in \mathcal{S}_p} y_{ps} = 1 \quad \forall p \in \mathcal{C}^P \tag{6}$$

$$\sum_{k \in \mathcal{K}} \sum_{u \in \mathcal{U}} x^u_{psk} \leq y_{ps} \quad \forall p \in \mathcal{C}^P, s \in \mathcal{S} \tag{7}$$

$$x^u_{pwk} + y_{ps} \leq x^{u+1}_{wsk} + 1 \quad \forall p \in \mathcal{C}^P, s \in \mathcal{S}_p, w \in \mathcal{W}, u \in \mathcal{U}, k \in \mathcal{K} \tag{8}$$

$$\sum_{k \in \mathcal{K}} \sum_{u \in \mathcal{U}} x^u_{pp'k} + y_{ps} \leq y_{p's} + 1 \quad \forall p, p' \in \mathcal{C}^P, p \neq p', s \in \mathcal{S}_p \tag{9}$$

$$\sum_{u \in \mathcal{U}} \sum_{i \in \mathcal{V}} x^u_{\bar{p}ik} = \sum_{u \in \mathcal{U}} \sum_{i \in \mathcal{V}} x^u_{\bar{d}ik} \quad \forall (\bar{p}, \bar{d}) \in \mathcal{R}, k \in \mathcal{K} \tag{10}$$

$$Q_{\bar{d}k} = Q_{\bar{p}k} - q_{\bar{p}} \quad \forall (\bar{p}, \bar{d}) \in \mathcal{R}, k \in \mathcal{K} \tag{11}$$

$$Q_{jk} \geq (Q_{ik} + q_j) - Q(1 - \sum_{u \in \mathcal{U}} x^u_{ijk}) \quad \forall (i,j) \in \mathcal{A}, j \notin \mathcal{S}, k \in \mathcal{K} \tag{12}$$

$$L_{sk} \geq Q_{ik} - Q(1 - \sum_{u \in \mathcal{U}} x^u_{isk}) \; \forall (i,s) \in \mathcal{A}, s \in \mathcal{S}, k \in \mathcal{K} \tag{13}$$

$$max\{0, q_i\} \leq Q_{ik} \leq min\{Q, Q + q_i\} \quad \forall i \in \mathcal{V}, k \in \mathcal{K} \tag{14}$$

$$Q_{sk} \leq Q \sum_{d \in \mathcal{C}^D_s} \sum_{u \in \mathcal{U}} x^u_{sdk} \quad \forall s \in \mathcal{S}, k \in \mathcal{K} \tag{15}$$

$$Q_{sk} \geq Q^{min}_d \sum_{d \in \mathcal{C}^D_s} \sum_{u \in \mathcal{U}} x^u_{sdk} \quad \forall s \in \mathcal{S}, k \in \mathcal{K} \tag{16}$$

$$L_{sk} \geq Q^{min}_p \sum_{i \in \mathcal{V} \setminus \{\mathcal{C}^D \cup \mathcal{C}^D_{c2c}\}} \sum_{u \in \mathcal{U}} x^u_{sik} \quad \forall s \in \mathcal{S}, k \in \mathcal{K} \tag{17}$$

$$Q_{sk} \leq Q(1 - \sum_{i \in \mathcal{V} \setminus \{\mathcal{C}^D \cup \mathcal{C}^D_{c2c}\}} \sum_{u \in \mathcal{U}} x^u_{sik}) \quad \forall s \in \mathcal{S}, k \in \mathcal{K} \tag{18}$$

$$\sum_{k \in \mathcal{K}} L_{sk} = \sum_{p \in \mathcal{C}^P} q_p y_{ps} \quad \forall s \in \mathcal{S} \tag{19}$$

$$\begin{gathered} B_{jk} \geq B_{ik} + \delta(i) + c_{ij} - M(1 - \sum_{u \in \mathcal{U}} x^u_{ijk}) \\ \forall (i,j) \in \mathcal{A}, i \in \{\mathcal{C}^P \cup \mathcal{C}^D \ \cup \mathcal{C}^P_{c2c} \cup \mathcal{C}^D_{c2c}\}, j \in \mathcal{V} \setminus \{g \cup \mathcal{W}\}, i \neq j, k \in \mathcal{K} \end{gathered} \tag{20}$$

$$B_{ik} \geq B_{sk} + \varphi'(s) + c_{si} - M(1 - \sum_{u \in \mathcal{U}} x^u_{sik}) \quad \forall s \in \mathcal{S}, i \in \{\mathcal{C}^P \cup \mathcal{C}^P_{c2c} \cup \mathcal{S} \setminus s\}, k \in \mathcal{K} \tag{21}$$

$$B_{dk} \geq B_{sk} + \varphi(s) + z_{sk}\varphi'(s) + c_{sd} - M(1 - \sum_{u \in \mathcal{U}} x^u_{sdk}) \quad \forall s \in \mathcal{S}, d \in \mathcal{C}^D_s, k \in \mathcal{K} \tag{22}$$

$$B_{iwk} \geq B_{ik} + \delta(i) + c_{iw} - M(1 - \sum_{u \in \mathcal{U}} x^u_{iwk}) \quad \forall w \in \mathcal{W}, i \in \{\mathcal{C}^P \cup \mathcal{C}^D \cup \mathcal{C}^D_{c2c}\}, k \in \mathcal{K} \tag{23}$$

$$B_{swk} \geq B_{sk} + \varphi'(s) + c_{sw} - M(1 - \sum_{u \in \mathcal{U}} x^u_{swk}) \quad \forall w \in \mathcal{W}, s \in \mathcal{S}, k \in \mathcal{K} \tag{24}$$

$$\begin{gathered} \text{If } \sum_{u \in \mathcal{U} \setminus e} \left(x^u_{iwk} \ x^{u+1}_{wsk} \right) = 1 \text{ then } B_{sk} \geq B_{iwk} + c_{ws} \\ \forall w \in \mathcal{W}, s \in \mathcal{S}, i \in \{\mathcal{C}^P \cup \mathcal{C}^D \cup \mathcal{C}^D_{c2c} \cup \mathcal{S} \setminus s\}, k \in \mathcal{K} \end{gathered} \tag{25}$$

$$z_{sk} = 1 \text{ if and only if } \sum_{p \in \mathcal{C}^P} \left(\sum_{u \in \mathcal{U}} x^u_{psk} + \sum_{u \in \mathcal{U} \setminus e} \sum_{w \in \mathcal{W}} x^u_{pwk} \ x^{u+1}_{wsk} \right) = 1 \quad \forall s \in \mathcal{S}, k \in \mathcal{K} \tag{26}$$

$$(t(s) - \eta) \sum_{u \in \mathcal{U}} \sum_{i \in \mathcal{V}} x^u_{isk} \leq B_{sk} \leq t(s) \sum_{u \in \mathcal{U}} \sum_{i \in \mathcal{V}} x^u_{sik} \quad \forall s \in \mathcal{S}, k \in \mathcal{K} \tag{27}$$

$$e_i \sum_{u \in \mathcal{U}} \sum_{j \in \mathcal{V}} x^u_{ijk} \leq B_{ik} \leq l_i \sum_{u \in \mathcal{U}} \sum_{j \in \mathcal{V}} x^u_{jik} \quad \forall i \in \{\mathcal{C}^P \cup \mathcal{C}^D \cup \mathcal{C}^P_{c2c} \cup \mathcal{C}^D_{c2c}\}, k \in \mathcal{K} \tag{28}$$

$$B_{\bar{p}k} + (\delta(\bar{p}) + c_{\bar{p}\bar{d}}) \sum_{u \in \mathcal{U}} \sum_{j \in \mathcal{V}} x^u_{\bar{p}jk} \leq B_{\bar{d}k} \quad \forall (\bar{p}, \bar{d}) \in \mathcal{R}, k \in \mathcal{K} \tag{29}$$

$$0 \leq L_{sk} \leq Q \quad \forall s \in \mathcal{S}, k \in \mathcal{K} \tag{30}$$

$$x^u_{ijk} \in \{0,1\} \quad \forall (i,j) \in \mathcal{A}, u \in \mathcal{U}, k \in \mathcal{K} \tag{31}$$

$$y_{ps} \in \{0,1\} \quad \forall p \in \mathcal{C}^P, s \in \mathcal{S} \tag{32}$$

$$z_{sk} \in \{0,1\} \quad \forall s \in \mathcal{S}, k \in \mathcal{K} \tag{33}$$

The objective function (1) minimizes the total transportation cost, including the fixed costs incurred for using vehicles. Constraints (2) ensure that each customer demand is visited exactly once. The conservation of flow at nodes, except the garage, is completed by constraints (3) and (4). Constraints (5) ensure that each vehicle starts and ends at the garage. Constraints (6) impose that each pickup-customer demand must be assigned to one supply point only. Constraints (7) - (9) forbid the illegal pickup legs that would bring either pickup-customer demands to a supply point to which they are not (yet) assigned. Constraints (10) ensure that for each request, the c2c-pickup-customer demand and c2c-delivery-customer demand are visited by the same vehicle. The LIFO policy is imposed through constraints (11).

Consistency of load variables is ensured through constraints (12) and (13), while constraints (14) enforce the restrictions on the vehicle capacity. Constraints (15) and (16) ensure that the vehicle k brings a load from a supply point s to a delivery-customer demand d of the customer zone $\mathcal{C}_s^D$ if and only if it loads freight at supply point s, i.e., $Q_{sk} > 0$. Constraints (17) and (18) ensure that the vehicle k goes directly from the supply point s to either a (c2c)-pickup-customer demand, another supply point, a waiting station, or the garage g if it only unloads at s and then leaves s empty. Constraints (19) guarantee that the total pickup load entering each supply point equals the total demands of pickup-customer demands that are assigned to the corresponding supply point.

Consistency of the time variables is ensured through constraints (20) - (25). Note that when a waiting station w is reached in the u^{th} position of the work assignment of vehicle k, the outgoing arc (w, s) should be in the $(u+1)^{th}$ position of the same work assignment. Constraints (25) can be linearized by introducing new variables $v^u_{iwsk} \in \{0, 1\}$ such that $v^u_{iwsk} = x^u_{iwk} x^{u+1}_{wsk}, \forall w \in \mathcal{W}, s \in \mathcal{S}, i \in \{\mathcal{C}^P \cup \mathcal{C}^D \cup \mathcal{S} \setminus s\}, k \in \mathcal{K}$. Constraints (25) can be made explicit by means of the following linear constraints:

$$x^u_{iwk} \geq v^u_{iwsk} \quad \forall u \in \mathcal{U}, w \in \mathcal{W}, s \in \mathcal{S}, i \in \{\mathcal{C}^P \cup \mathcal{C}^D \cup \mathcal{S} \setminus s\}, k \in \mathcal{K} \tag{34}$$

$$x^{u+1}_{wsk} \geq v^u_{iwsk} \quad \forall u \in \mathcal{U}, w \in \mathcal{W}, s \in \mathcal{S}, i \in \{\mathcal{C}^P \cup \mathcal{C}^D \cup \mathcal{S} \setminus s\}, k \in \mathcal{K} \tag{35}$$

$$x^u_{iwk} + x^{u+1}_{wsk} \leq 1 + v^u_{iwsk} \quad \forall u \in \mathcal{U}, w \in \mathcal{W}, s \in \mathcal{S}, i \in \{\mathcal{C}^P \cup \mathcal{C}^D \cup \mathcal{S} \setminus s\}, k \in \mathcal{K} \tag{36}$$

$$B_{sk} \geq B_{iwk} + c_{ws} - M(1 - \sum_{u \in \mathcal{U}} v^u_{iwsk}) \quad \forall w \in \mathcal{W}, s \in \mathcal{S}, i \in \{\mathcal{C}^P \cup \mathcal{C}^D \cup \mathcal{S} \setminus s\}, k \in \mathcal{K} \tag{37}$$

Constraints (26) ensure that the vehicle k unloads at a supply point s if and only if it brings demands of pickup-customer demands to s. Because each pickup-customer demand p is serviced only once, these constraints can be linearized and rewritten as follows:

$$z_{sk} = \sum_{p \in \mathcal{C}^P} \left(\sum_{u \in \mathcal{U}} x^u_{psk} + \sum_{u \in \mathcal{U}} v^u_{pwsk} \right) \quad \forall s \in \mathcal{S}, k \in \mathcal{K} \tag{38}$$

The respect of time windows at supply points and customer demands is enforced through constraints (27) and (28), respectively. Constraints (29) impose that for each request $(\bar{p}, \bar{d}) \in \mathcal{R}$, the c2c-pickup-customer demand $\bar{p}$ is visited before its corresponding c2c-delivery-customer demand $\bar{d}$. Constraints (30) are bounding constraints for variables L_{sk}. Finally, constraints (31) - (33) define the decision variables.