CIRRELT

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

Interuniversity Research Centre
on Enterprise Networks, Logistics and Transportation

A Branch-and-Cut-and-Price for the Multi-Trip Pickup and Delivery Problem with Time Windows and Synchronization

Andrea Bettinelli
Valentina Cacchiani
Teodor Gabriel Crainic
Daniele Vigo

January 2018

CIRRELT-2018-05

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

A Branch-and-Cut-and-Price for the Multi-Trip Pickup and Delivery Problem with Time Windows and Synchronization

Andrea Bettinelli[1], Valentina Chacchiani[2], Teodor Gabriel Crainic[3,*], Daniele Vigo[2]

[1] OPTIT Srl, Viale Amendola 56/D, 40026 Imola (BO), Italy

[2] DEI, University of Bologna, Viale del Risorgimento 2, 40136 Bologna, Italy

[3] Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation (CIRRELT) and Department of Management and Technology, Université du Québec à Montréal, P.O. Box 8888, Station Centre-Ville, Montréal, Canada H3C 3P8

Abstract. In this paper, we study the Multi-trip Pickup and Delivery Problem with Time Windows and Synchronization (MT-PDTWS). It arises in the context of two-tiered city logistics systems, considering both inbound and outbound traffic. The first tier refers to the transportation between the city distribution centers, in the out-skirts of the city, and intermediate facilities, while the second tier refers to the transportation of goods between the intermediate facilities and the (pickup and delivery) customers. Customers and facilities have time windows in which they can be visited, and waiting is possible at waiting stations for free or at customers and facilities at a given cost. Therefore, synchronization of vehicle arrivals and operating time windows of facilities and customers is an important element of the problem setting. The MT-PDTWS calls for determining minimum (fixed, routing and waiting) cost multi-trip routes, for a given fleet of vehicles, to service pickup and delivery customers, while taking into account vehicle capacity, time windows, and synchronization. We propose the first exact algorithm for MT-PDTWS: a Branch-and-Cut-and-Price algorithm. It is based on column generation, where the pricing problem is solved by a bi-directional dynamic programming algorithm designed to cope with the features of the problem. Subset-row and rounded capacity inequalities are adapted to deal with MT-PDTWS and inserted in the Branch-and-Cut-and-Price algorithm. The performance of the proposed algorithm is tested on benchmark instances with up to 200 customers, showing its effectiveness.

Keywords: Multi-trip pickup and delivery problem with time windows and synchronization, synchronization, branch-and-cut-and-price, bi-directional dynamic programming.

Acknowledgements. While working on this project, the third author was Adjunct Professor with the Computer Science and Operations Research Department, Université de Montréal. The authors thank Dr. Phuong Khanh Nguyen for fruitful discussion and cooperation on the topic. Partial funding for this project comes from the Discovery Grant and the Discovery Accelerator Supplements programs of the Natural Science and Engineering Research Council of Canada (NSERC), and the Strategic Clusters program of the Fonds québécois de la recherche - Nature et Technologies (FRQNT). The authors thank the institutions for supporting this research. This research is also based upon work supported by the Air Force Office of Scientific Research under award numbers FA9550-17-1-0025 and FA9550-17-1-0234.

---

* Corresponding author: TeodorGabriel.Crainic@cirrelt.ca

Dépôt légal – Bibliothèque et Archives nationales du Québec
Bibliothèque et Archives Canada, 2018

1 Introduction

We address the Multi-trip Pickup and Delivery Problem with Time Windows and Synchronization (MT-PDTWS), which arises in the context of two-tiered City Logistics systems (Crainic et al., 2009, 2012; Cattaruzza et al., 2017). In these systems, the first tier refers to the consolidation and transportation of loads between the city distribution centers, in the outskirts of the city, and intermediate facilities, called *satellites*, located inside the city (or close to it), by using large-capacity vehicles. The second tier considers the delivery/pickup of goods between the satellites and the customers, by using smaller-capacity vehicles that can travel inside the city. The integration of inbound and outbound operations is aimed at reducing the number of empty vehicle movements of all vehicle fleets and the freight traffic in the city. A key element of the problem is the synchronization at the satellites, since the latter are used as cross-dock facilities without intermediate storage. This class of problems has recently received significant attention in the literature (see, e.g., Nguyen et al., 2013; Crainic et al., 2015b; Nguyen et al., 2015; Crainic et al., 2016; Grangier et al., 2016; Guastaroba et al., 2016; Nguyen et al., 2017), as environmental issues and traffic congestion have become more critical in every day life (Guastaroba et al., 2016).

In MT-PDTWS, a homogeneous fleet of vehicles operates *multi-trip* routes out of a single garage to visit customers where loads are delivered and picked up. The loads, that are customer-specific, are available at the satellites within specified hard time windows, and must be delivered within the hard *time window* of the respective customer. The same or different customers have loads that must be picked up, within customer-specific hard time windows, and be brought to the satellite associated with the customer, within the period of activity of the satellite. A vehicle must complete a satellite-to-customer delivery sequence before starting a pickup sequence or moving to a satellite for another delivery phase. Waiting at satellites and customers is undesirable and, thus, *synchronization* of vehicle arrivals and operating time windows of satellites and customers is an important characteristic of the problem setting. *Waiting stations* (e.g., parking lots) are available where the vehicles can wait before moving to a satellite or to a customer. The goal is to find minimum (fixed, routing and waiting) cost multi-trip routes for a given fleet of vehicles that service all pickup and delivery customers, while respecting capacity, time windows and synchronization constraints.

The problem under study is similar to the one introduced in Nguyen et al. (2017). Two are the main differences between the two problems: the first one is that, in Nguyen et al. (2017), only delivery customers are pre-assigned to a specific satellite, while the choice on the specific satellites serving pickup customers has to be determined; the second difference is that MT-PDTWS introduces the flexibility of waiting at customers and/or satellites at a given cost, and the possibility of going to a waiting station at any time, including between customer visits. Indeed, in Nguyen et al. (2017), waiting is only possible at waiting stations before moving to a satellite. The additional flexibility of

MT-PDTWS allows considering alternative options of waiting at customers, satellites and waiting stations, that can further reduce traffic congestion, while taking into account the trade-off between waiting and routing costs. In other words, it is possible to let a vehicle wait at a customer or at a satellite, if, for example, there are parking facilities near the customer, or the waiting time is short. Moreover, it is possible to have the vehicle waiting at a waiting station between two customer visits, if there is no convenient waiting place at the customers or the waiting time is rather long.

To the best knowledge of the authors, Nguyen et al. (2017) proposed the only solution method directly aimed at this class of problems, which is a tabu search meta-heuristic. No exact method has been proposed for this class of problems yet. Our objective is to contribute to filling this gap. We propose an Integer Linear Programming (ILP) formulation and a Branch-and-Cut-and-Price algorithm for the MT-PDTWS. The ILP model contains exponentially many variables, associated with trips, that are combined to form complete routes. To compute a lower bound, we developed a bi-directional dynamic programming algorithm (Righini and Salani, 2006, 2008), employed in a column generation procedure. It is designed to cope with the specific features of MT-PDTWS, namely multi-trip, pickup and delivery customers associated with intermediate facilities, and synchronization. Especially the latter needs to be carefully addressed, by considering the possibility of stopping at waiting stations or at customers and facilities at a given cost, and influences the structure of the algorithm. Effective dominance rules and label filtering procedures are proposed to reduce the number of labels. Starting from the well-known subset-row and rounded capacity inequalities, we defined valid inequalities that are embedded in the Branch-and-Cut-and-Price algorithm and allow deriving significantly stronger lower bounds. Different branching rules are applied to obtain integer optimal solutions. We analyze the behavior and performance of the proposed Branch-and-Cut-and-Price algorithm through a comprehensive set of experimentations.

The paper starts with a brief overview of related literature, in Section 2, followed by the formal description of the problem and the model we introduce in Section 3. Section 4 describes the solution method we propose, which is evaluated by computational experiments reported in Section 5. Finally, we draw our conclusions and present future research directions in Section 6. In an Appendix we report the used notation.

2 Literature review

MT-PDTWS generalizes several Vehicle Routing Problems (VRP), as it includes pickup and delivery, multi-trips and time windows (Toth and Vigo, 2002, 2014; Vidal et al., 2013; Lahyani et al., 2015). The most challenging aspect of the MT-PDTWS, often neglected in previous VRP variants, is the synchronization at the intermediate facilities.

Pickup and Delivery Problems (P&DP) have been extensively studied in the literature (Savelsbergh and Solomon, 1995; Parragh et al., 2008a,b; Berbeglia et al., 2007, 2010). There are two main types of P&DP: goods can be transported from the depot to the delivery customers and from pickup customers to the depot, or between pickup and delivery customers. MT-PDTWS falls in the first category.

A related problem is the *Vehicle Routing Problem with Backhauls* (VRPB). In VRPB, the set of customers is partitioned into two subsets: *linehaul* and *backhaul* customers. Each linehaul customer requires the delivery of a given quantity of product from the depot, whereas a given quantity of product must be picked up from each backhaul customer and transported to the depot. The VRPB has been studied since the 1980s by Golden et al. (1985). Toth and Vigo (1997) and Mingozzi et al. (1999) propose exact methods based on ILP formulations. Many classical heuristics and meta-heuristics have also been proposed for the VRPB. Some of the recent references include Brandao (2006), Wassan (2007), Gajpal and Abad (2009b), Cuervo et al. (2014), Lai et al. (2015), Brandão (2016), Vidal et al. (2014). MT-PDTWS generalizes VRPB, as the latter determines a single-trip routing with first delivery customers and then pickup ones, and synchronization is not taken into account. Other related problems are the *Vehicle Routing Problem with Mixed linehauls and backhauls* (Ropke and Pisinger, 2006) and the *Vehicle Routing Problem with Simultaneous Delivery and Pickup* (Montané and Galvao, 2006; Chen and Wu, 2006; Bianchessi and Righini, 2007; Gajpal and Abad, 2009a; Subramanian et al., 2010), in which no synchronization appears as well.

The *Vehicle Routing Problem with Cross-Docking* (VRPCD), on the contrary, addresses synchronization of vehicle operations at a cross-dock facility. It consists of picking up goods, consolidating them at an intermediate facility without storage, and then delivering goods to the customers. Synchronization takes place at the intermediate facility. VRPCD has recently received increasing attention in the literature and mostly heuristic algorithms have been developed: see, e.g., Wen et al. (2009), Santos et al. (2011), Petersen and Ropke (2011), Tarantilis (2013), Santos et al. (2013), Maknoon and Laporte (2017), Grangier et al. (2017). MT-PDTWS generalizes VRPCD, as in the latter each vehicle performs a single tour composed of two trips with first pickup and then delivery, while in MT-PDTWS there are no restrictions on the number of trips nor on their sequencing.

The multi-trip setting appears in several VRP variants, such as the *Multi-trip Vehicle Routing Problem* (MTVRP) (Taillard et al., 1996; Brandão and Mercer, 1998; Petch and Salhi, 2003; Salhi and Petch, 2007; Olivera and Viera, 2007; Battarra et al., 2009; Azi et al., 2010; Mingozzi et al., 2013; Azi et al., 2014; Cattaruzza et al., 2014a,b; Hernandez et al., 2014; Cattaruzza et al., 2016; Hernandez et al., 2016), the *VRP with Intermediate Facilities* (VRPIF) or with inter-depot routes (Crevier et al., 2007; Tarantilis et al., 2008; Schneider et al., 2015), the *Multi-trip VRP with Backhauls* (MTVRPB) (Wassan et al., 2017), and the Waste Collection VRP (Kim et al., 2006; Ombuki-Berman et al., 2007;

Benjamin and Beasley, 2010; Markov et al., 2016). In VRPIF, the route of a vehicle can be composed of multiple stops at intermediate depots in order for the vehicle to be replenished. In MTVRPB, as in VRPB, all delivery customers are served before pickup ones, however each vehicle may perform more than one trip in a single planning period. MT-PDTWS generalizes both problems, as each route can visit several satellites (multi-trip feature) and synchronization constraints and costs are also taken into account.

In the recent literature, we can find several works that are more closely related to MT-PDTWS, in the context of two-tiered City Logistics systems. In particular, MT-PDTWS extends the *Time-dependent Multi-zone Multi-trip Vehicle Routing problem with Time Windows* (TMZT-VRPTW), which considers synchronization, but only delivery customers. TMZT-VRPTW problems have been introduced by Crainic et al. (2009), who presented several variants of the problem and mathematical formulations, and proposed a heuristic approach based on decomposition, but did not report an implementation. The idea is to first solve the Vehicle Routing with Time Windows subproblems representing the delivery to the customers of each satellite; then a minimum cost network flow problem is solved to put together the vehicle trips resulting from the solution of the subproblems. An implementation of this method is presented by Crainic et al. (2015a,b). To assess the quality of the solutions, they also compute a lower bound by relaxing vehicle capacity and time windows constraints at satellites and customers. Nguyen et al. (2013) propose a tabu search meta-heuristic for TMZT-VRPTW, featuring two types of neighborhoods, corresponding to the two sets of decisions of the problem, namely, the assignment of vehicles to satellites and the assignment of customers to vehicles. The selection of the neighborhoods is dynamically adjusted along the search. A diversification strategy, guided by an elite set, and a frequency-based memory, are proposed. The reported computational experiments show that the method produces good quality results on instances with up to 3600 customers.

MT-PDTWS belongs to the problem class studied in Nguyen et al. (2017), where a tabu search algorithm is proposed. It extends the method by Nguyen et al. (2013) by new neighborhoods to deal with both pickup and delivery customers. Instances with up to 72 facilities and 7200 customers are tested. This algorithm was the first method developed for MT-PDTWS: to evaluate its performance, the authors compared their results with those obtained by a preliminary version of the Branch-and-Cut-and-Price algorithm we propose (after adapting the corresponding methods due to the differences explained in Section 1), on instances with up to 180 customers. The comparison shows that an average optimality gap of 1.71% is obtained by the tabu search, but larger optimality gaps occur for instances with more customers and satellites. We refer the interested reader to Nguyen et al. (2017) for this comparison. We also recall that MT-PDTWS introduces the flexibility of waiting at customers and/or satellites at a given cost, and the possibility of going to a waiting station at any time, including between customer visits, which are realistic extensions that can be useful in practice.

Recently, an extension of the problem, called *Multi-trip Multi-traffic Pickup and Delivery Problem with Time Windows and Synchronization* (MTT-PDTWS) has been studied in Nguyen et al. (2015) and Crainic et al. (2016). In MTT-PDTWS, multi-trip delivery and pickup routes are executed to serve three types of customer requests: customer-to-external zone (i.e. pickup customers), external zone-to-customer (i.e. delivery customers) and customer-to-customer (i.e. load to be transported from a pickup customer to a delivery customer). Therefore, MTT-PDTWS extends MT-PDTWS by considering customer-to-customer requests. In Nguyen et al. (2015) and Crainic et al. (2016), the tabu search proposed in Nguyen et al. (2013) is extended to deal with this additional type of service.

From the literature analysis above, it turns out that our problem generalizes P&DP, VRPB, VRPCD, and MTVRP. To the best of our knowledge, no exact method has been proposed so far that directly applies to such extension, which has only been faced through heuristic and meta-heuristic algorithms.

3 Problem description and ILP formulation

The Multi-trip Pickup and Delivery Problem with Time Windows and Synchronization (MT-PDTWS) can be described as follows. We consider a single garage (or depot) g and a fleet of m identical vehicles of capacity Q and fixed cost F based at g. A set of intermediate facilities, called *satellites*, is given, where customer loads are available and where loads picked-up at customers have to be brought, during specific time windows. A set of customers is also given, who can require delivery or pickup of loads, or both, during specific time windows, in the considered planning horizon T. All time windows are considered hard.

We underline that loads can be available at or be brought to the facilities during different time windows of the planning horizon. Similarly, a customer can be a delivery customer or a pickup customer or both during different time windows of the planning horizon. We model this combination of resources and time periods as in Nguyen et al. (2013, 2017): we define *supply point* a combination of a facility and a time window when it is available for delivery or pickup. Each supply point $s \in \mathcal{S}$ has an associated time window $[t(s) - \eta, t(s)]$, specifying the earliest and latest times at which a vehicle can be at s: η is a small value, thus allowing only a short time window for visiting s. Each supply point $s \in \mathcal{S}$ also has an associated time $\varphi(s)$ representing the time required for all loading and unloading operations of a vehicle at s.

In addition, we define each load as a customer demand, not to be split into multiple visits, and distinguish between *delivery-customer demands* and *pickup-customer demands*, characterized by the customer, the supply point where the load is available

or has to be brought, and the customer time window. More precisely, we define the set of delivery-customer demands $\mathcal{C}^D$ and the set of pickup-customer demands $\mathcal{C}^P$. In addition, we define the subset $\mathcal{C}_s^D \subseteq \mathcal{C}^D$ of delivery-customer demands and the subset $\mathcal{C}_s^P \subseteq \mathcal{C}^P$ of pickup-customer demands that can be serviced by supply point $s \in \mathcal{S}$ (with $\mathcal{C}^D = \cup_{s\in\mathcal{S}}\mathcal{C}_s^D$ and $\mathcal{C}^P = \cup_{s\in\mathcal{S}}\mathcal{C}_s^P$). Sets $\mathcal{C}_s^P$ and $\mathcal{C}_s^D$ constitute, respectively, the *pickup service zone* and the *delivery service zone* of supply point $s \in \mathcal{S}$. For each supply point $s \in \mathcal{S}$, we call $\mathcal{C}_s^P \cup \mathcal{C}_s^D$ the *service zone* of s. For every supply point $s \in \mathcal{S}$, each (delivery or pickup) customer demand $i \in \mathcal{C}_s^D \cup \mathcal{C}_s^P$ is characterized by $(i, q_i, \delta(i), [e_i, l_i])$, where q_i is the quantity to be delivered or picked up at i, $\delta(i)$ is the required service time, and $[e_i, l_i]$ is the customer time window.

An important feature of the considered problem is synchronization of vehicle arrivals and operating time windows of satellites and customers. A set of *waiting stations* $w \in \mathcal{W}$ is given, where the vehicle can freely wait before going to a supply point or to a customer demand. Vehicles are also allowed to wait, before the start of the time window, at supply points and/or customers at a given cost: we define σ the unit waiting-time cost.

The detailed modeling of the operations in the facility is beyond the scope of this paper. Consequently, we assume (i) there is no limit on the number of vehicles that can be simultaneously present at a supply point $s \in \mathcal{S}$; (ii) all the loading and unloading operations happen at the end of the time window for a duration of $\varphi(s)$ time units; (iii) all vehicles leave at $t(s) + \varphi(s)$ from the supply point.

We represent supply points, waiting stations and customer demands through a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{A})$, with node set $\mathcal{V} = \{g\} \cup \mathcal{S} \cup \mathcal{C}^D \cup \mathcal{C}^P \cup \mathcal{W}$ and arc set $\mathcal{A}$, representing the possible movements between the nodes (see the Appendix of Supplementary material of Nguyen et al. (2017) for more details). Each arc $(i, j) \in \mathcal{A}$ has an associated positive routing coefficient c_{ij} representing the routing cost and time associated with the movement between i and j $(i, j \in \mathcal{V})$.

The *work assignment*, also called *multi-trip route*, of each vehicle is a, possibly empty, sequence of feasible legs (trips), starting and ending at the depot, where a *leg* is a sequence of customer services between either two supply points or a supply point and the depot. Feasibility conditions for the legs are defined as follows:

- A *starting leg* l, starting at the external depot g and ending at a supply point $s \in \mathcal{S}$, is feasible if it collects a total of goods not exceeding Q from a subset of customers in $\mathcal{C}_s^P$ within their time windows and arrives at s within the opening time window $[t(s) - \eta, t(s)]$;
- An *ending leg* l, starting at supply point $s \in \mathcal{S}$ and ending at the external depot g, is feasible if it loads a total of goods not exceeding Q at s at time $t(s)$, then leaves s at time $t(s) + \varphi(s)$ to deliver to a subset of delivery-customers in $\mathcal{C}_s^D$ within their time windows;

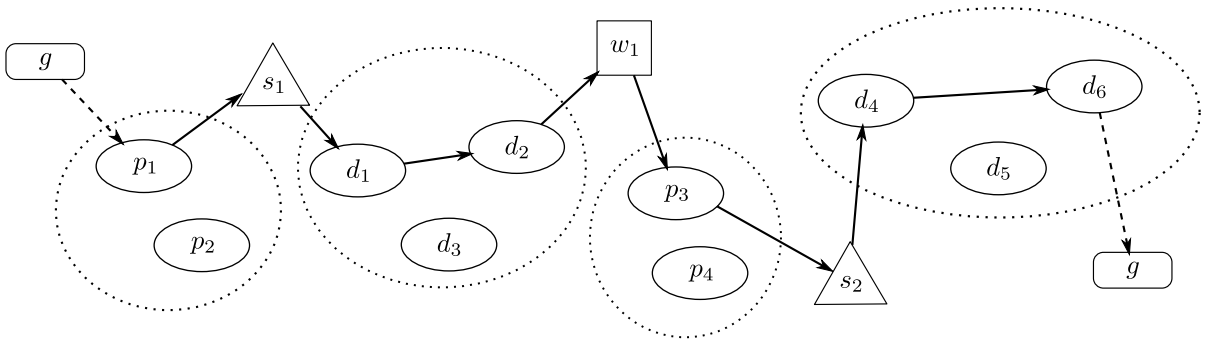

Figure 1: Example of three-leg work assignment.

- An *inter-facility leg* l that starts at supply point s and ends at supply point s', is feasible if it starts loading a total amount of goods not exceeding Q at supply point s at time $t(s)$, then leaves s at time $t(s) + \varphi(s)$ to deliver to a (possibly empty) subset of delivery-customers in $\mathcal{C}_s^D$ within their time windows; then visits a (possibly empty) subset of pickup-customers in $\mathcal{C}_{s'}^P$ for a total amount not exceeding Q, and arrives at s' within the opening time window $[t(s') - \eta, t(s')]$.

We recall that vehicles may arrive early at a customer i or at a supply point s, and wait, paying a unit waiting-time cost σ. Anywhere along the leg the vehicle can wait at a waiting station $w \in \mathcal{W}$ without paying any cost.

Figure 1 illustrates an example of a three-leg work assignment, where s_1 and s_2 are the supply points, g is the depot, w_1 is a waiting station, $P_{s_1} = \{p_1, p_2\}$ and $D_{s_1} = \{d_1, d_2, d_3\}$ are the sets of pickup and delivery customer demands, respectively, associated with supply point s_1 and $P_{s_2} = \{p_3, p_4\}$, $D_{s_2} = \{d_4, d_5, d_6\}$ are the pickup and delivery customer demands, respectively, associated with supply point s_2. The work assignment consists of a sequence of three legs $\{l_1, l_2, l_3\}$, where $l_1 = \{g, p_1, s_1\}$ is a starting leg, $l_2 = \{s_1, d_1, d_2, w_1, p_3, s_2\}$ is an inter-facility leg and $l_3 = \{s_2, d_4, d_6, g\}$ is an ending leg. As one can see, waiting station w_1 is visited between two customers (d_2 and p_3). However, it is also possible to visit the waiting station before a supply point.

The MT-PDTWS can be seen as the problem of determining a set of feasible work assignments for the vehicles to serve all customer demands. The objective is to minimize the total cost, which is given by the sum of the fixed cost of using the vehicles, the routing costs of servicing the customers and moving between supply points, and the waiting costs at customers and supply points.

We now define the proposed mathematical formulation. Let $\mathcal{L}$ be the set of all feasible (starting, inter-facility and ending) legs. We denote by $\mathcal{L}^+(s)$ the set of legs starting from $s \in \mathcal{S} \cup \{g\}$, and with $\mathcal{L}^-(s)$ the set of legs ending at $s \in \mathcal{S} \cup \{g\}$. We define a_{il} coefficients

as

$$a_{il} = \begin{cases} 1 & \text{if customer demand } i \in \mathcal{C}^P \cup \mathcal{C}^D \text{ is visited by leg } l \in \mathcal{L}; \\ 0 & \text{otherwise;} \end{cases}$$

and define the binary decision variables

$$x_l = \begin{cases} 1 & \text{if leg } l \in \mathcal{L} \text{ belongs to a work assignment (i.e., is selected in the solution),} \\ 0 & \text{otherwise.} \end{cases}$$

Let π_l be the total routing and waiting cost of leg l ($l \in \mathcal{L}$), defined as follows. Let $\mathcal{V}_l = \mathcal{V}_l^C \cup \mathcal{V}_l^S \cup \mathcal{V}_l^W$ be the set of nodes visited by l, with $\mathcal{V}_l^C$ as the set of customer demand nodes, $\mathcal{V}_l^S$ as the set of supply point nodes in l and $\mathcal{V}_l^W$ as the set of waiting stations in l. Let $\mathcal{A}_l$ be the set of arcs belonging to l. Finally, let T_l be the set of arrival time instants θ_i at each node $i \in \mathcal{V}_l$. We define the routing cost of l as $c_l := \sum_{(i,j)\in A_l} c_{ij}$ and its waiting cost as $w_l := \sum_{i\in\mathcal{V}_l^C:\theta_i<e_i} \sigma(e_i-\theta_i) + \sum_{i\in\mathcal{V}_l^S:\theta_s<t(s)-\eta} \sigma(t(s)-\eta-\theta_s)$. Then, $\pi_l := c_l + w_l$.

The MT-PDTWS can then be formulated as

$$\text{Minimize} \sum_{l\in\mathcal{L}} \pi_l x_l + \sum_{l\in\mathcal{L}^+(g)} F x_l \tag{1a}$$

$$\text{s.t.} \sum_{l\in\mathcal{L}} a_{il} x_l = 1 \qquad \forall i \in \mathcal{C}^P \cup \mathcal{C}^D \tag{1b}$$

$$\sum_{l\in\mathcal{L}^+(g)} x_l \leq m \tag{1c}$$

$$\sum_{l\in\mathcal{L}^+(s)} x_l = \sum_{l\in\mathcal{L}^-(s)} x_l \qquad \forall s \in \mathcal{S} \tag{1d}$$

$$x_l \in \{0,1\} \qquad \forall l \in \mathcal{L}. \tag{1e}$$

The objective function (1a) minimizes the total cost of operating and using the vehicles. Constraints (1b) ensure that every customer demand is visited exactly once. Constraint (1c) states that at most m work assignments can be selected. Constraints (1d) guarantee the flow conservation at each supply point, since no vehicle can stop at a supply point. These constraints allow combining legs into a (multi-trip) work assignment for a vehicle. In addition, these constraints ensure that each work assignment starts and ends at the depot (i.e., it is not possible to only select inter-facility legs). Note that, since we have time windows associated with the supply points and they are short (η is a small value), a work assignment cannot return to the same supply point, thus avoiding the need for supply point-subtour elimination constraints. Finally, let us call λ_i, μ and γ_s the dual variables associated with constraints (1b), (1c) and (1d), respectively, with λ_i free ($i \in \mathcal{C}^P \cup \mathcal{C}^D$), $\mu \leq 0$ and γ_s free ($s \in \mathcal{S}$).

4 Solution method

To solve MT-PDTWS we propose a Branch-and-Cut-and-Price algorithm, based on ILP model (1). As the model contains exponentially many variables, a lower bound on the optimal solution value is obtained by solving its Linear Programming (LP) relaxation by column generation. The pricing problem to derive negative reduced cost legs consists of a *Resource Constrained Elementary Shortest Path Problem* (RCESPP), which is NP-hard (Dror, 1994). To solve it, we propose a bi-directional dynamic programming algorithm (see, e.g., Righini and Salani, 2006, 2008), designed to cope with the features of MT-PDTWS, and combined with effective dominance rules and label filtering to reduce the number of labels. A heuristic pricing algorithm is also developed to speed-up the column generation procedure. Two types of inequalities, that extend powerful well-known inequalities from the literature, are applied to significantly improve the quality of the lower bound. Finally, three branching rules are sequentially applied on fractional solutions to determine an optimal integer solution.

The proposed Branch-and-Cut-and-Price algorithm takes into account the specific features of MT-PDTWS, namely multi-trip, (pickup and delivery) customer demands belonging to different service zones, and synchronization. Since a route is composed by several legs (trips), the dynamic programming algorithm computes single legs and then they are combined in the master problem. As we have customer demands divided into service zones, the dynamic programming algorithm considers them separately in label propagation and then combines labels of different zones. For the same reason, the proposed inequalities are defined on specific service zones: this allows dealing with a smaller number of inequalities, that can be easily considered in the dynamic programming algorithm. Finally, the key issue is synchronization and how to deal with time resource: indeed, a smaller time consumption in a leg is not necessarily an advantage because waiting implies a cost. To tackle this issue, we define a particular label structure, and specific dominance rules and label filtering that allow fathoming labels to speed-up the solution process. All these features are detailed in the next sections.

Section 4.1 describes how we initialize the solution process. The main features of the algorithm are then presented: the lower bound computation with the two types of inequalities in Section 4.2, the pricing algorithm combined with effective dominance rules and label filtering in Section 4.3, the heuristic pricing algorithm in Section 4.4, and the branching rules in Section 4.5.

4.1 Initialization

The first step of the proposed algorithm is to reduce the size of network $\mathcal{G}$ by removing arcs that cannot be used in any optimal solution. In particular, for given $i, j \in \mathcal{C}^P \cup \mathcal{C}^D$,

if

$$e_i + \delta(i) + c_{ij} > l_j$$

then arc (i, j) can be removed from $\mathcal{A}$. Then we start with a restricted master problem, which contains, for each constraint (1b), a very high cost dummy variable to guarantee feasibility.

4.2 Lower bound

To compute a lower bound, we iteratively search for promising legs, i.e., variables with negative reduced cost and add them to the restricted problem. When negative reduced cost variables no longer exist, the linear relaxation of the restricted master problem is equivalent to the LP relaxation of model (1) and gives a valid lower bound for the MT-PDTWS.

The pricing problem is modeled as a RCESPP, where all the feasibility constrains on the route legs are enforced (see Section 3). We observe that each inter-facility leg is composed of a delivery phase and a pickup phase: therefore, it is possible to compute the two *partial legs* independently and then join them to find a complete leg. Each starting leg or each ending leg can be seen as a special inter-facility leg, where one of the two phases is empty. Before describing, in Section 4.3, the algorithm used to find negative reduced cost route legs, we explain how a stronger lower bound can be obtained by adding valid inequalities to the linear relaxation of model (1). In particular, we consider subset-row inequalities and rounded capacity inequalities. Both types of inequalities are adapted to deal with MT-PDTWS by taking into account subsets of customers (in the same or different service zones), and are defined so as to effectively be used within the dynamic programming solution framework.

Zone-based subset-row inequalities (ZSR3) We consider the subset-row inequalities introduced by Jepsen et al. (2008) for the VRP with Time Windows. In particular, we consider the case in which subsets have cardinality three. Let $\mathcal{C}_3 = \{C \subseteq (\mathcal{C}^P \cup \mathcal{C}^D) : |C| = 3\}$ be the set of all subsets C of customer demands of cardinality 3, and, for $C \in \mathcal{C}_3$, let $\mathcal{L}(C) \subseteq \mathcal{L}$ be the set of legs visiting at least 2 customer demands in C. The following inequalities are valid for model (1):

$$\sum_{l \in \mathcal{L}(C)} x_l \leq 1 \quad \forall C \in \mathcal{C}_3. \tag{2}$$

Given a subset of three customer demands, these inequalities require to select at most one leg among all legs that visit at least two customer demands in the subset. Let us note ρ_C the dual variables associated with inequalities (2), with $\rho_C \leq 0$ ($C \in \mathcal{C}_3$). Since the

number of inequalities in (2) is polynomial in the number of customers, the separation can be easily performed by complete enumeration. In the Branch-and-Cut-and-Price algorithm, we perform the separation when no negative reduced cost variable can be found by the pricing algorithm.

To embed inequalities of the form (2) within the dynamic programming algorithm, instead of considering any subset of three customer demands, we divide them into two groups:

- *Intra-zone subset-row inequalities*, when the three customer demands in C belong to the same (pickup or delivery) service-zone, i.e., the three customer demands are all pickup customer demands or all delivery customer demands of a supply point;
- *Inter-zone subset-row inequalities*, when the three customer demands in C belong to different service-zones, i.e., the three customer demands belong to service zones of different supply points.

Given a supply point $s \in \mathcal{S}$, we call $\mathcal{C}_3^{intra,\mathcal{P}}(s) = \{C \subseteq \mathcal{C}_s^P : |C| = 3\}$ the set of subsets of pickup customer demands of cardinality 3 that induce *intra-zone* subset-row inequalities, and $\mathcal{C}_3^{intra,\mathcal{D}}(s) = \{C \subseteq \mathcal{C}_s^D : |C| = 3\}$ the set of subsets of delivery customer demands of cardinality 3 that induce *intra-zone* subset-row inequalities. Given a pair of supply points $s, s' \in \mathcal{S}$, we call $\mathcal{C}_3^{inter}(s, s') = \{C \subseteq (\mathcal{C}_s^D \cup \mathcal{C}_{s'}^P) : |C| = 3, |C \cap \mathcal{C}_s^D| \geq 1, |C \cap \mathcal{C}_{s'}^P| \geq 1\}$ the set of subsets of customer demands of cardinality 3, with at least one customer in the delivery zone of s and another customer in the pickup zone of s', that induce *inter-zone* subset-row inequalities. Intra-zone and inter-zone inequalities will be used separately by the dynamic programming algorithm. In particular, intra-zone inequalities will be considered when building a partial leg of a delivery or a pickup service zone, while inter-zone inequalities will be considered when joining partial legs into a complete one. We refer to Section 4.3 for further details.

Zone-based rounded-capacity inequalities (ZCAP) Rounded capacity inequalities are well known valid inequalities for routing problems (see Naddef and Rinaldi, 2002), which require all subsets of customers to be served by enough vehicles. Instead of considering any subset of customers, we consider a special case of such inequalities in which the set of customers contains either all the pickup customer demands or all the delivery customer demands of a supply point. When we consider all the pickup customer demands associated with a supply point $s \in \mathcal{S}$, the inequalities take the form:

$$\sum_{l \in \mathcal{L}^-(s)} x_l \geq \left\lceil \frac{\sum_{i \in \mathcal{C}_s^P} q_i}{Q} \right\rceil \quad \forall s \in \mathcal{S}. \tag{3}$$

They require that the number of legs visiting pickup customer demands to be served by supply point s is at least the rounded ratio of the sum of the pickup customer demands

over the capacity of the vehicle. Similar inequalities can be written for the delivery customer demands. Note that we define the rounded capacity inequalities on a delivery service zone or on a pickup service zone: this allows to easily deal with these inequalities within the dynamic programming algorithm (see Section 4.3). In addition, having defined these inequalities on the service zones allows to have a small number of inequalities: thus, they are directly added to the master problem before starting the column generation procedure. We denote $\xi_{\mathcal{P}_s}$ the dual variables associated with inequalities (3), referred to pickup customers, and $\xi_{\mathcal{D}_s}$ the dual variables associated with the same inequalities but referred to delivery customers, with $\xi_{\mathcal{P}_s} \geq 0$ and $\xi_{\mathcal{D}_s} \geq 0$ ($s \in \mathcal{S}$).

4.3 Pricing algorithm

Dynamic programming techniques are very effective in solving RCESPPs, and, in particular, we focus on bi-directional extension of node labels (Righini and Salani, 2006, 2008), which is based on forward and backward label propagation (see, e.g., Dell'Amico et al., 2006; Ceselli et al., 2009; Righini and Salani, 2009; Bettinelli et al., 2011; Salani and Vacca, 2011; Bettinelli et al., 2014; Vidal et al., 2015). Labels are associated with nodes of $\mathcal{G}$. The proposed dynamic programming algorithm applies bi-directional extension of node labels, which we designed to effectively cope with multi-trip, service zones and synchronization features that required ad hoc adaptations of the standard framework. In the following we report the pricing problem, and describe label structure, propagation, join of labels, dominance rules, and label filtering.

Pricing problem. The pricing problem calls for finding a negative reduced cost feasible leg $l \in \mathcal{L}$. Let us consider an inter-facility leg $l \in \mathcal{L}$ with supply points $s, s' \in \mathcal{S}$. As before, we let $\mathcal{V}_l = \mathcal{V}_l^C \cup \mathcal{V}_l^S \cup \mathcal{V}_l^W$ be the set of nodes visited by l, with $\mathcal{V}_l^C$ as the set of customer demand nodes, $\mathcal{V}_l^S$ as the set of supply point nodes in l and $\mathcal{V}_l^W$ as the set of waiting stations in l, and $\mathcal{A}_l$ be the set of arcs belonging to l. Finally, T_l indicates the set of arrival time instants θ_i at each node $i \in \mathcal{V}_l$. In addition, we define $\mathcal{C}^*_{intra,\mathcal{D}}(s)$, $\mathcal{C}^*_{intra,\mathcal{P}}(s)$ and $\mathcal{C}^*_{inter}(s,s')$ as the sets of all subsets of customer demands of cardinality 3 that induce, respectively, delivery intra-zone, pickup intra-zone and inter-zone subset-row inequalities, such that at least two customer demands are visited by leg l: $\mathcal{C}^*_{intra,\mathcal{D}}(s) = \{C \in \mathcal{C}_3^{intra,\mathcal{D}}(s) : |V_l \cap C| \geq 2\}$, $\mathcal{C}^*_{intra,\mathcal{P}}(s) = \{C \in \mathcal{C}_3^{intra,\mathcal{P}}(s) : |V_l \cap C| \geq 2\}$ and $\mathcal{C}^*_{inter}(s,s') = \{C \in \mathcal{C}_3^{inter}(s,s') : |V_l \cap C| \geq 2\}$. The pricing problem is to find a leg $l \in \mathcal{L}$, satisfying feasibility conditions described in Section 3, such that it has negative reduced cost:

$$\begin{aligned}
&\sum_{(i,j)\in\mathcal{A}_l} c_{ij} + \sum_{i\in\mathcal{V}_l^C:\theta_i<e_i} \sigma(e_i-\theta_i) + \sum_{i\in\mathcal{V}_l^S:\theta_s<t(s)-\eta} \sigma(t(s)-\eta-\theta_s) \\
&- \sum_{i\in\mathcal{V}_l^C} \lambda_i - \gamma_s + \gamma_{s'} \\
&- \sum_{C\in\mathcal{C}^*_{intra,\mathcal{D}}(s)} \rho_C - \sum_{C\in\mathcal{C}^*_{intra,\mathcal{P}}(s')} \rho_C - \sum_{C\in\mathcal{C}^*_{inter}(s,s')} \rho_C - \xi_{\mathcal{D}_s} - \xi_{\mathcal{P}_{s'}} < 0.
\end{aligned} \tag{4}$$

In the first line we can see the routing and waiting cost of the leg, the second line takes into account the dual variables of constraints (1b) and (1d), while the last line considers the dual variables of the intra-zone and inter-zone subset-row inequalities and of the rounded capacity inequalities. Recall that rounded capacity inequalities are inserted in the master before starting the column generation procedure, while subset-row inequalities are separated by enumeration.

If we consider a starting leg, we need to add $F-\mu$ to the reduced cost, as the fixed cost of the vehicle and the dual variable of constraint (1c) need to be taken into account. In addition, for a starting leg from g to s', we do not have the terms including dual variables referred to s, while, for an ending leg from s to g, we do not have the terms including dual variables referred to s'.

To solve this problem, we developed a dynamic programming algorithm. We consider each delivery and pickup service zone of a supply point $s \in \mathcal{S}$. Labels are propagated forward from each supply point s to its delivery customers $i \in \mathcal{C}_s^D$ (delivery service zone) and backward to its pickup customers $j \in \mathcal{C}_s^P$ (pickup service zone). In particular, forward labels represent paths from s to i and backward labels represent paths from j to s. Each label includes information on its resource consumption (e.g., time, vehicle capacity), visited customers and reduced cost. Each label is iteratively considered and the corresponding path is extended to adjacent nodes. Note that, in the extension, we need to consider both the direct arc between two nodes and the possibility to stop at a waiting station in between.

Forward and backward labels belonging to zones of different supply points are joined in pairs to form complete legs. Before giving details on the label structure, we explain the main novelties of the dynamic programming algorithm.

As mentioned above, multi-trip, service zones and synchronization features need to be carefully handled in the bi-directional extension of node labels. Since we have multi-trips and (delivery and pickup) customer demands associated with specific supply points, it is important to consider delivery and pickup service zone separately in the label extension and combine them through the joining of labels. This also allows to effectively deal with subset-row and rounded-capacity inequalities. Moreover, we need to consider the

opportunity of visiting waiting stations when extending labels of a delivery or a pickup service zone, and when joining labels of different supply points. Another issue consists of the way to cope with synchronization and time consumption. Due to the synchronization constraints and to the additional flexibility of waiting at supply points or customers that we allow, a smaller time consumption is not necessarily an advantage when evaluating states of the dynamic programming: indeed, it might imply larger waiting costs when propagating to other nodes. On the other hand, a larger time consumption is clearly a limitation, as in classical routing problems with time windows, since it might inhibit to visit some nodes. Thus, there might be an infinite number of non-dominated states associated with the same path, having the same cost, but different time consumptions, obtained by delaying the departure time from a waiting station (or from the depot). To cope with this issue, we group them into a single label by introducing an additional resource ϕ representing the *forward shift* of the label, i.e., the amount of delay that can be introduced without violating the time window constraints of the visited customers.

Label structure. A label is a tuple $L = (i, \tau, \phi, RC, \chi, \Psi)$, where $i \in \mathcal{S} \cup \mathcal{C}^P \cup \mathcal{C}^D$ is the last customer or supply point reached, τ is the time consumption (i.e., the service starting time at i), ϕ is the forward shift, RC is the reduced cost, χ is the vehicle load, and Ψ is the set of visited customers.

Supply-point labels are initialized as follows:

- Forward label of supply point $s \in \mathcal{S}$ is $(i = s, \tau = t(s), \phi = 0, RC = -\gamma_s, \chi = 0, \Psi = \emptyset)$;
- Backward label of supply point $s \in \mathcal{S}$ is $(i = s, \tau = t(s), \phi = \eta, RC = \gamma_s, \chi = 0, \Psi = \emptyset)$;
- Forward label of g is $(i = g, \tau = 0, \phi = +\infty, RC = F - \mu, \chi = 0, \Psi = \emptyset)$;
- Bbackward label of g is $(i = g, \tau = \infty, \phi = +\infty, RC = 0, \chi = 0, \Psi = \emptyset)$.

Both forward and backward labels of supply point $s \in \mathcal{S}$ have time consumption $\tau = t(s)$, since $t(s)$ is the time at which the vehicle starts the (loading/unloading) service at s. The reduced cost of the forward labels of supply point $s \in \mathcal{S}$ is initialized as $-\gamma_s$, since the leg departs from s towards the delivery customers, while one of the backward labels is initialized to γ_s, since the leg arrives at s after servicing the pickup customers. Notice that, since there are no time constraints on the departure and arrival times at the main depot, the labels associated with g have infinite forward shift. In addition, we associate with the forward label of g the fixed cost F of the vehicle minus the dual variable μ of constraint (1c).

Label propagation. We describe in detail the extension rules for forward labels, i.e., from a delivery customer label to another delivery customer label or from a supply point label to a delivery customer label. Similar rules are applied in the backward search.

When a label $L = (i, \tau, \phi, RC, \chi, \Psi)$, associated with node $i \in \mathcal{C}_s^D$ is extended to node $j \in \mathcal{C}_s^D$, the new label $L' = (j, \tau', \phi', RC', \chi', \Psi')$ is computed according to the following rules:

$$\tau' = \max\{\tau + \delta(i) + c_{ij}, e_j\} \tag{5a}$$
$$\phi' = \max\{0, \min\{\phi - \nu, l_j - \tau'\}\} \tag{5b}$$
$$RC' = RC + c_{ij} - \lambda_j + \sigma\omega \tag{5c}$$
$$\chi' = \chi + q_j \tag{5d}$$
$$\Psi' = \Psi \cup \{j\} \tag{5e}$$

where $\nu = \max\{e_j - (\tau + \delta(i) + c_{ij}), 0\}$ is the waiting time and $\omega = \max\{\nu - \phi, 0\}$ is the waiting time reduced by ϕ, since ϕ can (partially or fully) absorb it. In label L', the time consumption τ is increased by the service time duration $\delta(i)$ at node i plus the routing time c_{ij} from i to j. If this value is smaller than e_j, then the vehicle will wait at node j, and the time consumption is set equal to e_j. The forward shift ϕ' (that must be greater or equal to zero) is set as the minimum between the previous forward shift ϕ decreased by waiting time ν and $l_j - \tau'$. Indeed, ϕ' is the remaining amount of delay, after visiting customer j, that can be introduced without violation of the time window constraints of the customers in Ψ'. The reduced cost is updated by adding the routing and waiting costs, and subtracting the dual variable λ_j of the visited customer j. Finally, the vehicle load is increased by the customer demand q_j, and j is added to the visited customers.

In Figure 2, we show an example of forward label propagation: on the top, we can see the time window $[e_i, l_i]$ of label L and, on the bottom, the time window $[e_j, l_j]$ of label L'. In L, we have a time consumption τ and a forward shift ϕ. The arrow from τ indicates the arrival time at node j, i.e., $\tau + \delta(i) + c_{ij}$. As we can see, the vehicle arrives earlier than e_j and therefore must wait for time ν. However, the forward shift ϕ can be used to reduce the waiting time at j. Thus, the reduced cost RC' takes into account the waiting cost $\sigma\omega$. In this example, the new time consumption τ' is set to e_j, and the new forward shift ϕ' becomes zero.

When a label $L = (i, \tau, \phi, RC, \chi, \Psi)$, associated with node $i \in \mathcal{S}$ is extended to node $j \in \mathcal{C}_s^D$, the same computation is performed except for the time consumption that is set to $\tau' = \max\{\tau + \varphi(i) + c_{ij}, e_j\}$.

As described earlier, vehicles can stop at waiting stations instead of going directly from one customer to another. If a label of customer i is extended to customer j through

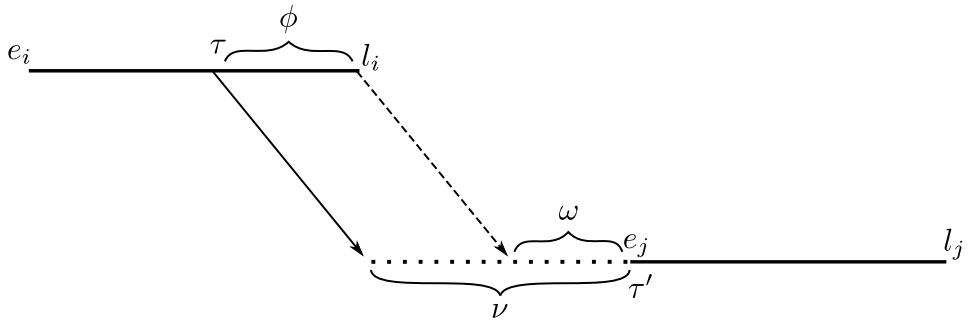

Figure 2: Example of forward label propagation.

the waiting station w, the new label is computed as:

$$\tau' = \max\{\tau + \delta(i) + c_{iw} + c_{wj}, e_j\} \tag{6a}$$
$$\phi' = l_j - \tau' \tag{6b}$$
$$RC' = RC + c_{iw} + c_{wj} - \lambda_j \tag{6c}$$
$$\chi' = \chi + q_j \tag{6d}$$
$$\Psi' = \Psi \cup \{j\} \tag{6e}$$

Note that we need routing cost $c_{iw} + c_{wj}$ for going from node i to the waiting station w and then to node j. In this case, the vehicle never waits at customer j (since it can wait at the waiting station at no cost), thus $\phi' = l_j - \tau'$. Given a pair of customers i and j, the most convenient waiting station where to stop is $\bar{w} \in \text{argmin}_{w \in \mathcal{W}}\{c_{iw} + c_{wj}\}$. This value $c_{iw} + c_{wj}$ does not depend on the dual variables. Hence, we precompute $\bar{w}$ for each pair of customers.

The label L' is feasible if

$$j \notin \Psi$$
$$\tau' \leq l_j$$
$$\chi' \leq Q$$

At the end of the search phase, *intra-zone subset-row inequalities* are evaluated and the reduced cost of the labels are updated accordingly. In particular, let $\mathcal{C}^*_{intra,\mathcal{D}}(s) = \left\{C \in \mathcal{C}_3^{intra,\mathcal{D}}(s) : |\Psi' \cap C| \geq 2\right\}$, i.e., $\mathcal{C}^*_{intra,\mathcal{D}}(s)$ is the set of all subsets of delivery customer demands of cardinality 3 that induce intra-zone subset-row inequalities, such that at least two delivery customer demands are visited by the partial leg corresponding to label L'. The reduced cost RC' is updated as $RC' - \sum_{C \in \mathcal{C}^*_{intra,\mathcal{D}}(s)} \rho_C$.

Join. In order to obtain complete (starting, ending or inter-facility) route legs, forward and backward labels of delivery and pickup service zones of different supply points are

joined in pairs. In other words, the join operation is used to join a label of a delivery service zone of a supply point s with a label of a pickup service zone of a supply point s'. Let us consider the join between a label $L_{fw} = (i, \tau_{fw}, \phi_{fw}, RC_{fw}, \chi_{fw}, \Psi_{fw})$ of delivery customer $i \in \mathcal{C}_s^D$ and label $L_{bw} = (j, \tau_{bw}, \phi_{bw}, RC_{bw}, \chi_{bw}, \Psi_{bw})$ of pickup customer $j \in \mathcal{C}_{s'}^P$. The two labels L_{fw} and L_{bw} are compatible if

$$\tau_{fw} + \delta(i) + c_{ij} \leq \tau_{bw}.$$

Indeed, it is necessary that the service starting time τ_{fw} at i plus the service time $\delta(i)$ and the routing cost c_{ij} from i to j is smaller or equal to the service starting time τ_{bw} at j. Since the two labels are linked to different service zones, the two labels have visited disjoint sets of customers. Also, the vehicle load resources are always compatible: the forward label performs delivery operations, while the backward label performs pickup operations. Thus at the join point the vehicle is empty.

Let s and s' be the starting supply points of L_{fw} and L_{bw}, respectively, and let $\xi_{\mathcal{D}_s}$ and $\xi_{\mathcal{P}_{s'}}$ be the dual variables of the rounded capacity inequalities associated with the delivery service zone of s and with the pickup service zone of s', respectively. Also, let $\mathcal{C}^*_{inter}(s, s') = \{C \in \mathcal{C}_3^{inter}(s, s') : |(\Psi_{fw} \cup \Psi_{bw}) \cap C| \geq 2\}$, i.e., $\mathcal{C}^*_{inter}(s, s')$ is the set of all subsets of customer demands of cardinality 3 that induce inter-zone subset-row inequalities, such that at least two customer demands are visited by the leg, between s and s', that will result from the join of labels L_{fw} and L_{bw}.

The reduced cost of the route leg resulting from the join of L_{fw} and L_{bw} is

$$RC_{fw} - \xi_{\mathcal{D}_s} + c_{ij} + \sigma\omega - \xi_{\mathcal{P}_{s'}} + RC_{bw} - \sum_{C \in \mathcal{C}^*_{inter}(s,s')} \rho_C,$$

where ω denotes the necessary waiting time between i and j that cannot be absorbed by the forward shift of both labels. It is defined as

$$\omega = \max\{0, \tau_{bw} - (\tau_{fw} + c_{ij} + \delta(i)) - (\phi_{bw} + \phi_{fw})\}.$$

Indeed, the waiting time at node j is given by the difference between the service starting time τ_{bw} at j, minus the sum of the ending time of service $\tau_{fw} + \delta(i)$ at i, plus the routing time c_{ij} between the two nodes, reduced by the sum of the forward shifts $\phi_{fw} + \phi_{bw}$.

As previously, it is necessary also for the join to consider the possibility to stop at a waiting station $w \in \mathcal{W}$. In this case, the feasibility condition for the join is

$$\tau_{fw} + \delta(i) + c_{iw} + c_{wj} \leq \tau_{bw}$$

and the resulting reduced cost is

$$RC_{fw} - \xi_{\mathcal{D}_s} + c_{iw} + c_{wj} - \xi_{\mathcal{P}_{s'}} + RC_{bw} - \sum_{C \in \mathcal{C}^*_{inter}(s,s')} \rho_C.$$

Notice that, to obtain a complete starting leg, the join operation is applied between a backward label of a supply point $s \in \mathcal{S}$ and the label of g, while to obtain a complete ending leg the join operation is applied between the label of g and a forward label of a supply point $s \in \mathcal{S}$. After joining, we discard all the labels with $RC \geq 0$.

Dominance rules. Effective dominance rules, capable of fathoming a large number of labels, are a fundamental ingredient in labeling algorithm for the RCESPP. As mentioned above, it is not easy to detect whether a larger time consumption is an advantage or a disadvantage when comparing labels. Hence, in principle, only labels with the same time consumption can be directly compared. This limitation significantly reduces the number of labels that can be fathomed. We use the two following dominance rules that combine the information on time consumption, cost and forward shift to compare labels with different time consumption.

Proposition 1 (Dominance Rule 1)**.** *A forward label $L' = (i, \tau', \phi', RC', \chi', \Psi')$, with $i \in \mathcal{C}_s^D$, dominates a forward label $L'' = (j, \tau'', \phi'', RC'', \chi'', \Psi'')$ if*

$$\begin{cases} i = j \\ \tau' \leq \tau'' \\ RC' + \sigma \max\{0, \tau'' + \phi'' - (\tau' + \phi')\} \leq RC'' \\ \chi' \leq \chi'' \\ \Psi' \subseteq \Psi'' \end{cases} \tag{7}$$

The term $\sigma \max\{0, \tau'' + \phi'' - (\tau' + \phi')\}$ is the cost incurred by label L' to wait until time $\tau'' + \phi''$, the latest feasible service starting time for label L''. In other words, it is not enough to have $RC' \leq RC''$, since L'' might have more possibilities to delay without violating the time window constraints of the visited customers. Therefore, we consider the time needed in L' to reach the latest feasible service starting time of L'', and evaluate the corresponding waiting cost. If RC' increased by this cost is still smaller or equal to RC'', then L'' is dominated.

Proposition 2 (Dominance Rule 2)**.** *A forward label $L' = (i, \tau', \phi', RC', \chi', \Psi')$, with $i \in \mathcal{C}_s^D$, dominates a forward label $L'' = (j, \tau'', \phi'', RC'', \chi'', \Psi'')$ if*

$$\begin{cases} i = j \\ \tau' + \max_{h \in \mathcal{C}_s^D} \{\min_{w \in \mathcal{W}} \{c_{iw} + c_{wh} - c_{ih}\}\} \leq \tau'' \\ RC' + \max_{h \in \mathcal{C}_s^D} \{\min_{w \in \mathcal{W}} \{c_{iw} + c_{wh} - c_{ih}\}\} \leq RC'' \\ \chi' \leq \chi'' \\ \Psi' \subseteq \Psi'' \end{cases} \tag{8}$$

Here, $\max_{h \in \mathcal{C}_s^D} \{\min_{w \in \mathcal{W}} \{c_{iw} + c_{wh} - c_{ih}\}\}$ is the maximum routing time (and cost) over all possible next customers $h \in \mathcal{C}_s^D$, which can be introduced by visiting a waiting

station before h. Dominance Rule 2 is used to dominate labels by considering the possibility of visiting a waiting station from the current node i before visiting the next node h. Instead of directly going from node i to node h (at cost c_{ih}), we consider the possibility of going from node i to the best waiting station for pair i, h (at cost $c_{iw} + c_{wh}$). If the time consumption τ' increased by the maximum additional routing time towards/from the waiting station is still smaller or equal to τ'' and the reduced cost RC' increased by the maximum additional routing cost is still smaller or equal to RC'', then L'' is dominated.

With Dominance Rule 1, we try to dominate a label L'' with a label L' by allowing to wait at the customer (if necessary), and thus paying the corresponding waiting cost. With Dominance Rule 2, we try to dominate a label L'' with a label L' by allowing to wait at a waiting station w: the waiting station is chosen such that it is the most convenient for customers i and h, and the additional routing time (and cost) to reach customer h through the waiting station is chosen as the maximum ones among all possible customers (since L' dominates L'' only if the *worst* situation is considered).

Similar dominance rules can be defined for backward labels. Dominance rules are applied during the label propagation, i.e., we check if a label is dominated by other labels before extending it.

Filtering final labels. Once the label propagation phase is terminated, and before applying the join operation, it is possible to filter the final labels using more effective dominance rules, which require weaker conditions to be satisfied. In particular, it is possible to drop conditions on the load of the vehicle and on the set of visited customer demands. By removing these conditions, we can be able to dominate additional labels. Recall that join is applied to combine a label of a delivery service zone and a label of a pickup service zone. Therefore, joining occurs with an empty vehicle and it is not necessary to check the vehicle load for final labels. In addition, forward and backward propagations operate on disjoint sets of customer demands and, thus, joining a forward and a backward labels never causes visiting a customer demand more than once. This allows removing the condition on the visited customer set. However, since inter-zone inequalities are used, they must be taken into account in this phase. Indeed, these inequalities contribute to increase the reduced cost. In particular, we add to the labels an additional resource α_C for each inequality $C \in \cup_{s' \in \mathcal{S}, s' \neq s} \mathcal{C}_3^{inter}(s, s')$, where $s \in \mathcal{S}$ is the supply point of the considered label. The value of α_C represents the number of customer demands of the considered label that are involved in inequality C. We apply the following rule for the filtering of final labels: A forward final label $L' = (i, \tau', \phi', RC', \chi', \Psi')$, with

$i \in \mathcal{C}_s^D$, dominates a forward final label $L'' = (j, \tau'', \phi'', RC'', \chi'', \Psi'')$ if

$$\begin{cases} i = j \\ \tau' \leq \tau'' \\ RC' + \sigma \max\{0, \tau'' + \phi'' - (\tau' + \phi')\} \leq RC'' \\ \alpha'_C \leq \alpha''_C \quad \forall C \in \cup_{s' \in \mathcal{S}, s' \neq s} \mathcal{C}_3^{inter}(s, s'). \end{cases} \tag{9}$$

In other words, it is not necessary to check that $\Psi' \subseteq \Psi''$, but only that $\alpha'_C \leq \alpha''_C$, $\forall C \in \cup_{s' \in \mathcal{S}, s' \neq s} \mathcal{C}_3^{inter}(s, s')$. Indeed, if the number α'_C of customer demands involved in an inter-zone inequality C for label L' is smaller or equal to the number α''_C of customer demands involved in the same inequality for label L'', the contribution of the dual variable ρ_C is either the same in both labels or it is larger for L''. Thus, since $\rho_C \leq 0$, this ensures dominance of L' over L'', provided that the other conditions above are satisfied.

4.4 Heuristic pricing algorithm

In order to speed-up the column generation procedure, we developed a heuristic pricing algorithm (HP). It is similar to the exact pricing algorithm described in Section 4.3, but uses relaxed dominance rules to fathom a larger number of non-promising labels. More precisely, the check on the set of visited customers $\Psi' \subseteq \Psi''$ is removed from both rules (7) and (8). HP is used to quickly find negative reduced cost legs. We noticed, during preliminary experiments, that the failure rate of HP in finding negative reduced cost columns increases when subset-row inequalities are added to the master problem. Hence, we use HP in the initial iterations of column generation and stop calling it as subset-row inequalities are generated. Clearly, the exact pricing algorithm is applied to derive a valid lower bound, when HP is not able to find a negative reduced cost column.

4.5 Branching rules

We consider three branching rules to be applied alternatively when a fractional solution is found. The first one limits the total number of vehicles, the second one imposes bounds on the number of vehicles visiting a supply point, and the third branching rule fixes or forbids an arc.

Let $v = \sum_{l \in \mathcal{L}^+(g)} x_l$ be the number of vehicles used in the optimal solution of the linear relaxation master problem. When v is fractional, we impose in one branch to use at most $\lfloor v \rfloor$ and at least $\lceil v \rceil$ vehicles in the other. This branching decision can be

enforced by introducing a constraint of the form

$$\sum_{l \in \mathcal{L}^+(g)} x_l \leq \lfloor v \rfloor$$

or

$$\sum_{l \in \mathcal{L}^+(g)} x_l \geq \lceil v \rceil$$

in the master problem and it does not affect the structure of the pricing problem. Indeed, we have an additional constraint whose dual variable can be taken into account, in a similar way as variable μ, in the initialization of the forward and backward labels of the depot g.

When an integer number of vehicles is used, we search for a supply point $\bar{s} \in \mathcal{S}$ visited by a fractional number of vehicles. Let $v_s = \sum_{l \in \mathcal{L}^+(s)} x_l$ be the number of vehicles visiting supply point $s \in \mathcal{S}$ in the optimal solution of the linear master problem; we perform a binary branching similar to the previous case, by adding to the master problem one of the following constraints in each child subproblem:

$$\sum_{l \in \mathcal{L}^+(\bar{s})} x_l \leq \lfloor v_{\bar{s}} \rfloor$$

or

$$\sum_{l \in \mathcal{L}^+(\bar{s})} x_l \leq \lceil v_{\bar{s}} \rceil.$$

When there are multiple supply points with fractional vehicle flow, the one with the fractional part closest to 0.5 is selected, as it often yields a stronger effect on the fractional solution when branching. The structure of the pricing problem is not destroyed in this case as well.

When no supply point has fractional vehicle flow, we branch on the arc selection. We compute the flow on the arcs corresponding to the optimal fractional solution of the master problem, and select the arc (i, j) whose flow is the closest to 0.75. Arcs with integer flow are not considered. Arc (i, j) is forbidden in one branch. The arc is then removed from $\mathcal{G}$ and, in the dynamic programming procedure, labels of node i will not be extended to node j. In this way, the pricing algorithm will not produce any route leg containing (i, j). All the variables already present in the master problem associated with route legs visiting i and j in sequence (even if stopping at a waiting station in between) are discarded as well. We impose to use arc (i, j) in the second branch. This is enforced by removing from $\mathcal{G}$ the arcs (i, j') for $j \neq j'$ and by removing from the master problem all the variables associated with route legs vising vertex i followed by a vertex different from j. Since this branching rule has generally a weaker impact on fractional solutions, it is only used when the two previous rules cannot be applied.

5 Experimental analysis

The goal of the experimental analysis is to evaluate the performance of the proposed Branch-and-Cut-and-Price algorithm. In particular, we first evaluate the impact of the proposed zone-based subset-row and zone-based rounded capacity inequalities on the computation of the lower bound at the root node (Section 5.2). Then, we report the results obtained by the Branch-and-Cut-and-Price algorithm (Section 5.3). Finally, in Section 5.4, we consider other problem settings of MT-PDTWS.

The algorithm described in Section 4 has been implemented in C++ using SCIP 4.0.1, linked to CPLEX 12.6.1, as a Branch-and-Cut-and-Price framework. We turned off preprocessing and automatic cut generation features, but kept all other parameters at their default values, including the decision tree search policy. The results reported in this section are obtained using a single core of an Intel i7-3820, 3.6GHz workstation.

5.1 Test instances

We generated seven sets of 10 problem instances, using the same parameters proposed in Crainic et al. (2015b) and Nguyen et al. (2017). The first three sets, called D1-D3, have the same number of supply points and different number of customers per service zone. The following three sets, called D4-D6, instead, have the same number of customers per service zone, but a different number of supply points. Finally, the last set, called E1, has 10 supply points and an average of 10 pickup and 10 delivery demands for each service zone. We decided to identify the last set with a different letter, because the larger number of customers per service zone makes them considerably more difficult than the previous six sets.

Supply points, waiting stations and customers are uniformly distributed in a square of size 200. The opening time of the supply points is randomly generated in the interval $[0, 14400]$. The parameters η for the time window and φ for the vehicle loading and unloading operations at the supply points are set to 100 and 30 respectively.

Customers are assigned to supply points based on their proximity. More precisely a customer is assigned to its first, second, third and fourth nearest supply point with probability 0.5, 0.25, 0.15, and 0.1, respectively. The ready time of a delivery customer is computed as the sum of the opening time of the corresponding supply point, the loading time at the supply point, the routing time between the supply point and the customer and a random value in the interval $[0, 300]$. The time window duration is uniformly selected in the $[150, 450]$ range. Similarly, the due date of a pickup customer is generated by subtracting the service time at the customer, the routing time from the customer to the supply point, and a random value in the interval $[0, 300]$ from the opening time of

the supply point. The service time of all customers is set to 20. The demand of each customer is randomly generated in the interval $[5, 25]$ and the vehicle capacity is set to 100. The main depot is set at the middle point of the square. A fixed cost $F = 500$ is accounted for each vehicle used. Waiting at customer and supply points is allowed with a penalty $\sigma = 0.5$.

The description of the datasets is summarized in Table 1.

dataset	$\|\mathcal{S}\|$	cust/zone	cust demands
D1	5	5	50
D2	5	7	70
D3	5	9	90
D4	5	6	60
D5	10	6	120
D6	15	6	180
E1	10	10	200

Table 1: Datasets description

The time limit was set to one hour in all tests with D1-D6 and to two hours in all tests with E1.

5.2 Lower bound

We have considered four configurations: the linear relaxation of model (1) without any additional cut (no cuts in Table 2), with zone-based subset-row inequalities (ZSR3), with zone-based rounded-capacity inequalities (ZCAP), and with both. The results are summarized in Table 2. The average percentage gap with respect to the optimal solution values and the average computing time, expressed in seconds, are reported for each configuration and each class of instances. We report in the last two rows the average values of gaps and computing times over all classes of instances and over D instances, respectively.

Without valid inequalities, the linear relaxation of model (1) gives a rather weak bound (the average gap is 8.67% on all instances). Zone-based subset-row inequalities are able to considerably improve the lower bound, reducing the average gap to 1.17%, but the computing times also increase, especially on class D3 and E1 (recall that we allow two hours of time limit for E1 instances, and one hour for D instances). The zone-based rounded-capacity inequalities are also effective in reducing the duality gap, even if not as much as ZSR3, and the computing time often decreases. We recall that such inequalities are added to the master problem since the beginning and they seem

	no cuts		ZSR3		ZCAP		ZSR3+ZCAP	
class	gap	time	gap	time	gap	time	gap	time
D1	6.83%	0.17	1.40%	0.89	1.99%	0.16	0.00%	0.60
D2	11.62%	0.91	0.68%	16.91	2.59%	0.58	0.39%	2.98
D3	8.65%	7.37	1.04%	238.82	5.24%	5.03	1.03%	150.47
D4	10.59%	0.39	1.44%	5.06	1.30%	0.32	0.52%	0.55
D5	8.28%	2.42	1.24%	12.44	2.23%	1.57	0.84%	2.70
D6	7.87%	7.96	0.86%	28.87	2.20%	5.57	0.63%	9.76
E1	6.83%	158.61	1.51%	3042.55	4.15%	122.14	0.26%	1878.38
avg. all	8.67%	25.40	1.17%	477.93	2.81%	19.34	0.52%	292.21

Table 2: Lower bound evaluation.

to significantly help a fast convergence of the column generation procedure. We achieve the strongest bound (the average gap is about 0.5%) within reasonable computing times (292 seconds on average for all instances and only 28 seconds on D instances) using both types of inequalities (2) and (3).

As frequently happens with exact algorithms, the variance of the gap values reported in Table 2 can be large. However, we observed a drastic reduction of the variance when the valid inequalities are present. Hence, the configuration with both ZSR3 and ZCAP is the one we used in the Branch-and-Cut-and-Price algorithm.

5.3 Branch-and-cut-and-price

Table 3 reports the results obtained on the seven classes of instances by the Branch-and-Cut-and-Price algorithm described in Section 4. We report for each class of instances, the average gap and computing time at the root node, the average gap when the time limit is reached, the average number of explored nodes for the solved instances, the average computing time for the solved instances, and the number of instances solved to optimality by the proposed Branch-and-Cut-and-Price algorithm (BCP).

All the instances of sets D1 to D6 could be solved to optimality within the time limit of 1 hour, exploring a small number of nodes: 44 instances out of 60 are closed at the root node, and a maximum of 9 nodes are explored.

All instances of set E1 were solved to optimality within the time limit of two hours. In particular, all instances but E1-10 were solved to optimality within one hour time limit with an average computing time of 1663 seconds, while E1-10 required 4511 seconds.

	lower bound		BCP			
class	gap	time	gap	nodes	time	solved
D1	0.00%	0.60	0.00%	1.0	0.61	10
D2	0.39%	2.98	0.00%	1.4	3.05	10
D3	1.03%	150.47	0.00%	2.4	167.61	10
D4	0.52%	0.55	0.00%	1.4	0.58	10
D5	0.84%	2.70	0.00%	2.4	2.98	10
D6	0.63%	9.76	0.00%	2.0	10.21	10
E1	0.26%	1878.38	0.00%	1.6	1947.57	10

Table 3: Results of the Branch-and-Cut-and-Price algorithm on MT-PDTWS.

The algorithm seems to be more sensitive to the size of the service zones than to the number of supply points. This is not surprising. The synchronization constraints force to use more strict dominance rules in the dynamic programming procedure. In fact, having consumed a shorter amount of time is not necessarily an advantage. Thus, less labels can be fathomed out and the dynamic programming procedure becomes more time consuming with respect to classical VRP problems.

5.4 Other problem settings

5.4.1 No-wait policy

In the MT-PDTWS a penalty σ is incurred for each time unit of waiting at customers or supply points. We investigated also the case in which waiting is completely forbidden, i.e., $\sigma = \infty$, since it is a realistic setting for congested city centers. We still allow waiting at waiting stations. In this setting, Dominance Rule 1 (7) is less effective since it always fails if $\tau' + \phi' < \tau'' + \phi''$. To improve the performance of the labeling algorithm, it is convenient to block a label extension if

$$\tau + \delta(i) + c_{ij} < e_j.$$

This means that we avoid propagating a label from node i to node j if it implies waiting at the customer j. Of course it is still possible to extend the label from i to j through a waiting station.

The results obtained on the seven classes of instances are summarized in Table 4. The table has the same structure of the previous one. The Branch-and-Cut-and-Price algorithm is again able to solve all the 60 instances in classes D1 to D6. Seven instances in class E1 could be solved within the time limit of two hours. However, for three E1

	lower bound		BCP			
class	gap	time	gap	nodes	time	solved
D1	0.00%	0.61	0.00%	1.0	0.62	10
D2	0.36%	2.67	0.00%	1.4	2.75	10
D3	0.83%	207.40	0.00%	2.2	233.88	10
D4	0.55%	0.48	0.00%	1.6	0.51	10
D5	0.92%	2.21	0.00%	3.4	2.95	10
D6	0.58%	8.90	0.00%	114.9	33.69	10
E1	19.35%	4683.41	19.08%	2.7	3865.63	7

Table 4: Results of the Branch-and-Cut-and-Price algorithm with no-wait policy.

instances the algorithm was still processing the root node after two hours of computing time: in this case, the quality of the heuristic solution obtained by SCIP was very poor. Indeed, we can observe that the number of nodes to determine the optimal solution is usually rather small, and many instances can directly be solved to optimality at the root node. When the root node computation is completed, finding a good quality heuristic solution becomes easier for the solver. However, the solution process of the root node requires long computing times for the E1 instances and this leads to larger gaps. E1 instances are more difficult mainly due to the larger number of customers in each service zone: this difficulty is more evident in this setting due to the weakness of dominance rule (7), as explained above.

5.4.2 Configuration of Nguyen et al. (2017)

We have experimented the behavior of the Branch-and-Cut-and-Price algorithm with a third problem setting, where vehicles are allowed to stop at waiting stations only immediately before going to a supply point. Waiting at customer and supply points is allowed with a penalty ($\sigma = 0.5$). This setting corresponds to the one used in Nguyen et al. (2017) on D instances for comparison with a preliminary version of the proposed Branch-and-Cut-and-Price algorithm. In this setting, we insert a cutoff given by the upper bound computed by the tabu search proposed in Nguyen et al. (2017). The obtained results are reported in Table 5. We can see that all but three instances in class D3 are solved to optimality. This class is the most difficult for the Branch-and-Cut-and-Price algorithm in this setting, since it has the largest number of customers in each service zone. Restricting the possibility of going to a waiting station only right before a supply point creates more opportunities to trade waiting costs with routing costs in the dynamic programming, thus making the dominance rule less effective.

	lower bound		BCP			
class	gap	time	gap	nodes	time	solved
D1	0.38%	0.12	0.00%	3.80	0.14	10
D2	0.86%	11.56	0.00%	393.30	76.31	10
D3	1.12%	2046.65	0.9%	2.71	1622.75	7
D4	0.38%	0.37	0.00%	2.0	0.42	10
D5	0.51%	3.43	0.00%	413.70	26.80	10
D6	0.72%	6.53	0.00%	534.30	42.46	10

Table 5: Results of the Branch-and-Cut-and-Price algorithm on the configuration in Nguyen et al. (2017).

5.5 Solution analysis

We analyze the characteristics of the optimal solutions obtained for the three problem settings. In particular, we report in Table 6, for each of the three problem settings considering only the instances solved to optimality, the average number of used vehicles, the average number of performed legs, the average number of legs per vehicle, the average fixed cost, routing cost and waiting cost.

We can see that the average numbers of used vehicles and legs are similar in the standard and in the no wait settings. However, larger routing costs clearly arise in the latter setting, since waiting at customers and supply point is forbidden, and the vehicle needs to travel to a waiting station. We also observe that, for all classes, the total routing and waiting cost in the standard setting is smaller (about 3% on average) than the routing cost in the no wait setting. Therefore, the flexibility of waiting at customers and supply points at a given cost can help reducing the total cost and the number of vehicles.

When it is allowed to go to a waiting station only right before a supply point (setting used in Nguyen et al. (2017) for comparison with the Branch-and-Cut-and-Price algorithm), the number of used vehicles and fixed costs significantly increase (about 40% on average). In addition, the number of legs also increases, and each vehicle performs a smaller number of legs. This is not desired, as it can also increase the traffic congestion. In addition, even the total routing and waiting cost is larger in this setting than in the standard setting (more than 20% on average).

standard setting						
class	vehicles	legs	legs/vehicles	fixed cost	routing cost	waiting cost
D1	2.2	7.6	3.5	1100.0	1535.9	79.6
D2	3.0	12.4	4.3	1500.0	1942.6	59.9
D3	3.1	13.2	4.5	1550.0	2205.6	81.2
D4	2.6	10.0	4.0	1300.0	1788.5	33.3
D5	3.2	17.9	6.1	1600.0	3146.2	98.8
D6	4.5	27.3	6.2	2250.0	4212.4	200.4
E1	5.3	25.6	5.1	2650.0	4011.4	219.3

no wait setting						
class	vehicles	legs	legs / vehicles	fixed cost	routing cost	waiting cost
D1	2.2	7.7	3.6	1100.0	1685.1	0.0
D2	3.0	12.4	4.3	1500.0	2069.7	0.0
D3	3.1	13.1	4.5	1550.0	2351.7	0.0
D4	2.6	10.0	4.0	1300.0	1846.7	0.0
D5	3.2	17.9	6.1	1600.0	3332.7	0.0
D6	4.5	27.4	6.2	2250.0	4576.0	0.0
E1	5.3	25.7	5.1	2642.9	4450.1	0.0

configuration of Nguyen et al. (2017)						
class	vehicles	legs	legs / vehicles	fixed cost	routing cost	waiting cost
D1	3.8	9.5	2.6	1900.0	1482.4	548.0
D2	6.4	16.2	2.6	3200.0	1842.2	536.9
D3	6.1	16.7	2.8	3071.4	2140.4	887.5
D4	4.3	11.6	2.8	2150.0	1730.0	568.0
D5	6.4	22.2	3.6	3200.0	3061.8	1475.0
D6	7.2	33.4	5.0	3600.0	4336.7	1853.3

Table 6: Solution characteristics for the three problem settings.

6 Conclusions

We studied the Multi-trip Pickup and Delivery Problem with Time Windows and Synchronization, arising in two-tiered city logistics systems, which includes several practical features such as multi-trip, pickup and delivery customers, and synchronization. The problem was introduced by Nguyen et al. (2017), and we extended it with the possibility of waiting at waiting stations at any stage, as well as waiting at customers and facilities at a given cost. We proposed an Integer Linear Programming model with exponentially many variables, and a Branch-and-Cut-and-Price algorithm, the first exact method for this class of two-tiered city logistics routing problems. In this algorithm, column generation is applied to derive a lower bound on the optimal solution value, the pricing problem, which consists of an Elementary Shortest Path Problem with Resource Constraints, being solved by a bi-directional dynamic programming algorithm, tailored for the problem at study. Valid inequalities, extended from the subset-row and rounded capacity inequalities, are embedded in the Branch-and-Cut-and-Price. To speed-up the solution process, we propose effective dominance rules, label filtering, and a heuristic dynamic programming algorithm.

The computational study was performed on a set of instances with up to 200 customers by considering three problem settings that differ for the possibility of waiting at customers and supply points. The obtained results show that, for the general setting in which waiting is allowed at customers and supply points at a given cost, the proposed Branch-and-Cut-and-Price algorithm is able to solve to optimality all instances with up to 180 customers within one hour of time limit, and all instances with up to 200 customers in two hours of time limit. As explained in the computational results, not only the total number of customers affects the performance of the proposed algorithm, but especially the number of customers per service zone, which is 10 for the instances with 200 customers. When waiting is not allowed at customers and supply points, all instances with up to 180 customers are solved to optimality within one hour of time limit, while three instances with 200 customers remain unsolved. The setting in which waiting at customers and supply points is allowed at a given cost, but it is possible to use a waiting station only right before a supply point, turns out to be the most difficult one: three instances with 90 customers and 9 customers per service zone remain unsolved, even though with a small average optimality gap.

The analysis of the characteristics of the obtained solutions suggests that waiting at customers and supply points is a real-world feature that can be easily handled by the proposed algorithm. With respect to the no wait policy, smaller routing costs arise when waiting is allowed. In addition, going to a waiting station only right before a supply point is more restrictive and can cause larger global costs. Therefore, the introduced flexibility of waiting at customers and supply points at a given cost, and going to a waiting station at any time, can help reducing costs and traffic congestion. Future research will be dedicated to study further extensions of the two-tiered city logistics systems, by including additional

real-world features and taking into account stochastic elements, such as uncertainty in the customer demand and in the routing time.

Acknowledgments

While working on this project, the third author was Adjunct Professor with the Computer Science and Operations Research Department, Université de Montréal. The authors thank Dr. Phuong Khanh Nguyen for fruitful discussion and cooperation on the topic. Partial funding for this project comes from the Discovery Grant and the Discovery Accelerator Supplements programs 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. The authors thank the institutions for supporting this research. This research is also based upon work supported by the Air Force Office of Scientific Research under award numbers FA9550-17-1-0025 and FA9550-17-1-0234.

References

N. Azi, M. Gendreau, and J.-Y. Potvin. An exact algorithm for a vehicle routing problem with time windows and multiple use of vehicles. *European Journal of Operational Research*, 202(3):756–763, 2010.

N. Azi, M. Gendreau, and J.-Y. Potvin. An adaptive large neighborhood search for a vehicle routing problem with multiple routes. *Computers & Operations Research*, 41: 167–173, 2014.

M. Battarra, M. Monaci, and D. Vigo. An adaptive guidance approach for the heuristic solution of a minimum multiple trip vehicle routing problem. *Computers & Operations Research*, 36(11):3041–3050, 2009.

A. Benjamin and J. Beasley. 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, 2010.

G. Berbeglia, J.-F. Cordeau, I. Gribkovskaia, and G. Laporte. Static pickup and delivery problems: a classification scheme and survey. *TOP*, 15:1–31, 2007.

G. Berbeglia, J.-F. Cordeau, and G. Laporte. Dynamic pickup and delivery problems. *European Journal of Operational Research*, 202:8–15, 2010.

A. Bettinelli, A. Ceselli, and G. Righini. A branch-and-cut-and-price algorithm for the multi-depot heterogeneous vehicle routing problem with time windows. *Transportation Research Part C: Emerging Technologies*, 19(5):723–740, 2011.

A. Bettinelli, A. Ceselli, and G. Righini. A branch-and-price algorithm for the multi-depot heterogeneous-fleet pickup and delivery problem with soft time windows. *Mathematical Programming Computation*, 6(2):171–197, 2014.

N. Bianchessi and G. Righini. Heuristic algorithms for the vehicle routing problem with simultaneous pick-up and delivery. *Computers & Operations Research*, 34(2):578–594, 2007.

J. Brandão and A. Mercer. The multi-trip vehicle routing problem. *Journal of the Operational research society*, pages 799–805, 1998.

J. Brandao. A new tabu search algorithm for the vehicle routing problem with backhauls. *European Journal of Operational Research*, 173(2):540–555, 2006.

J. Brandão. A deterministic iterated local search algorithm for the vehicle routing problem with backhauls. *TOP*, 24(2):445–465, 2016.

D. Cattaruzza, N. Absi, D. Feillet, and T. Vidal. A memetic algorithm for the multi trip vehicle routing problem. *European Journal of Operational Research*, 236(3):833–848, 2014a.

D. Cattaruzza, N. Absi, D. Feillet, and D. Vigo. An iterated local search for the multi-commodity multi-trip vehicle routing problem with time windows. *Computers & Operations Research*, 51:257–267, 2014b.

D. Cattaruzza, N. Absi, and D. Feillet. The multi-trip vehicle routing problem with time windows and release dates. *Transportation Science*, 50(2):676–693, 2016.

D. Cattaruzza, N. Absi, D. Feillet, and J. González-Feliu. Vehicle routing problems for city logistics. *EURO Journal on Transportation and Logistics*, 6(1):51–79, 2017.

A. Ceselli, G. Righini, and M. Salani. A column generation algorithm for a rich vehicle-routing problem. *Transportation Science*, 43(1):56–69, 2009.

J.-F. Chen and T.-H. Wu. Vehicle routing problem with simultaneous deliveries and pickups. *Journal of the Operational Research Society*, 57(5):579–587, 2006.

T. G. Crainic, N. Ricciardi, and G. Storchi. Models for evaluating and planning city logistics systems. *Transportation science*, 43(4):432–454, 2009.

T. G. Crainic, F. Errico, W. Rei, and N. Ricciardi. Integrating c2e and c2c traffic into city logistics planning. *Procedia-social and behavioral sciences*, 39:47–60, 2012.

T. G. Crainic, F. Errico, W. Rei, and N. Ricciardi. Modeling demand uncertainty in two-tier city logistics tactical planning. *Transportation Science*, 50(2):559–578, 2015a.

T. G. Crainic, Y. Gajpal, and M. Gendreau. Multi-zone multi-trip vehicle routing problem with time windows. *INFOR: Information Systems and Operational Research*, 53 (2):49–67, 2015b.

T. G. Crainic, P. K. Nguyen, and M. Toulouse. Synchronized multi-trip multi-traffic pickup & delivery in city logistics. *Transportation Research Procedia*, 12:26–39, 2016.

B. Crevier, J.-F. Cordeau, and G. Laporte. The multi-depot vehicle routing problem with inter-depot routes. *European Journal of Operational Research*, 176(2):756–773, 2007.

D. P. Cuervo, P. Goos, K. Sörensen, and E. Arráiz. An iterated local search algorithm for the vehicle routing problem with backhauls. *European Journal of Operational Research*, 237(2):454–464, 2014.

M. Dell'Amico, G. Righini, and M. Salani. A branch-and-price approach to the vehicle routing problem with simultaneous distribution and collection. *Transportation Science*, 40(2):235–247, 2006.

M. Dror. Note on the complexity of the shortest path models for column generation in vrptw. *Operations Research*, 42(5):977–978, 1994.

Y. Gajpal and P. Abad. An ant colony system (acs) for vehicle routing problem with simultaneous delivery and pickup. *Computers & Operations Research*, 36(12):3215–3223, 2009a.

Y. Gajpal and P. L. Abad. Multi-ant colony system (macs) for a vehicle routing problem with backhauls. *European Journal of Operational Research*, 196(1):102–117, 2009b.

B. Golden, E. Baker, J. Alfaro, and J. Schaffer. The vehicle routing problem with backhauling: two approaches. In *Proceedings of the twenty-first annual meeting of the SE TIMS, Myrtle Beach, SC, USA*, 1985.

P. Grangier, M. Gendreau, F. Lehuédé, and L.-M. Rousseau. An adaptive large neighborhood search for the two-echelon multiple-trip vehicle routing problem with satellite synchronization. *European Journal of Operational Research*, 254(1):80–91, 2016.

P. Grangier, M. Gendreau, F. Lehuédé, and L.-M. Rousseau. A matheuristic based on large neighborhood search for the vehicle routing problem with cross-docking. *Computers & Operations Research*, 84:116–126, 2017.

G. Guastaroba, M. G. Speranza, and D. Vigo. Intermediate facilities in freight transportation planning: a survey. *Transportation Science*, 50(3):763–789, 2016.

F. Hernandez, D. Feillet, R. Giroudeau, and O. Naud. A new exact algorithm to solve the multi-trip vehicle routing problem with time windows and limited duration. *4OR*, 12(3):235–259, 2014.

F. Hernandez, D. Feillet, R. Giroudeau, and O. Naud. Branch-and-price algorithms for the solution of the multi-trip vehicle routing problem with time windows. *European Journal of Operational Research*, 249(2):551–559, 2016.

M. Jepsen, B. Petersen, S. Spoorendonk, and D. Pisinger. Subset-row inequalities applied to the vehicle routing problem with time windows. *Operations Research*, 56:497—511, 2008.

B.-I. Kim, S. Kim, and S. Sahoo. Waste collection vehicle routing problem with time windows. *Computers & Operations Research*, 33(12):3624–3642, 2006.

R. Lahyani, M. Khemakhem, and F. Semet. Rich vehicle routing problems: From a taxonomy to a definition. *European Journal of Operational Research*, 241(1):1–14, 2015.

M. Lai, M. Battarra, M. Di Francesco, and P. Zuddas. An adaptive guidance metaheuristic for the vehicle routing problem with splits and clustered backhauls. *Journal of the Operational Research society*, 66(7):1222–1235, 2015.

Y. Maknoon and G. Laporte. Vehicle routing with cross-dock selection. *Computers & Operations Research*, 77:254–266, 2017.

I. Markov, S. Varone, and M. Bierlaire. Integrating a heterogeneous fixed fleet and a flexible assignment of destination depots in the waste collection vrp with intermediate facilities. *Transportation research part B: methodological*, 84:256–273, 2016.

A. Mingozzi, S. Giorgi, and R. Baldacci. An exact method for the vehicle routing problem with backhauls. *Transportation Science*, 33(3):315–329, 1999.

A. Mingozzi, R. Roberti, and P. Toth. An exact algorithm for the multitrip vehicle routing problem. *INFORMS Journal on Computing*, 25(2):193–207, 2013.

F. A. T. Montané and R. D. Galvao. A tabu search algorithm for the vehicle routing problem with simultaneous pick-up and delivery service. *Computers & Operations Research*, 33(3):595–619, 2006.

D. Naddef and G. Rinaldi. Branch-and-cut algorithms for the capacitated vrp. In *The vehicle routing problem*, pages 53–84. SIAM Monographs on Discrete Mathematics and Applications. SIAM, Philadelphia,, 2002.

P. K. Nguyen, T. G. Crainic, and M. Toulouse. 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, 2013.

P. K. Nguyen, T. G. Crainic, and M. Toulouse. Synchronized multi-trip multi-traffic pickup & delivery in city logistics. Technical Report CIRRELT-2015-05, Centre Interuniversitaire de Recherche sul les Réseaux d'Enterprise, la Logistique et le Transport (CIRRELT), Université de Montréal, Canada, 2015.

P. K. Nguyen, T. G. Crainic, and M. Toulouse. Multi-trip pickup and delivery problem with time windows and synchronization. *Annals of Operations Research*, 253(2):899–934, 2017.

A. Olivera and O. Viera. Adaptive memory programming for the vehicle routing problem with multiple trips. *Computers & Operations Research*, 34(1):28–47, 2007.

B. M. Ombuki-Berman, A. Runka, and F. Hanshar. Waste collection vehicle routing problem with time windows using multi-objective genetic algorithms. In *Proceedings of the Third IASTED International Conference on Computational Intelligence*, pages 91–97. ACTA Press, 2007.

S. Parragh, K. Doerner, and R. Hartl. A survey on pickup and delivery problems. Part I: Transportation between customers and depot. *Journal für Betriebswirtschaft*, 58: 21–51, 2008a.

S. Parragh, K. Doerner, and R. Hartl. A survey on pickup and delivery problems. Part II: Transportation between pickup and delivery locations. *Journal für Betriebswirtschaft*, 58:81–117, 2008b.

R. J. Petch and S. Salhi. A multi-phase constructive heuristic for the vehicle routing problem with multiple trips. *Discrete Applied Mathematics*, 133(1):69–92, 2003.

H. Petersen and S. Ropke. The pickup and delivery problem with cross-docking opportunity. *Computational Logistics*, pages 101–113, 2011.

G. Righini and M. Salani. Symmetry helps: Bounded bi-directional dynamic programming for the elementary shortest path problem with resource constraints. *Discrete Optimization*, 3(3):255–273, 2006.

G. Righini and M. Salani. New dynamic programming algorithms for the resource constrained elementary shortest path problem. *Networks*, 51(3):155–170, 2008.

G. Righini and M. Salani. Decremental state space relaxation strategies and initialization heuristics for solving the orienteering problem with time windows with dynamic programming. *Computers & Operations Research*, 36(4):1191–1203, 2009.

S. Ropke and D. Pisinger. A unified heuristic for a large class of vehicle routing problems with backhauls. *European Journal of Operational Research*, 171(3):750–775, 2006.

M. Salani and I. Vacca. Branch and price for the vehicle routing problem with discrete split deliveries and time windows. *European Journal of Operational Research*, 213(3): 470–477, 2011.

S. Salhi and R. Petch. A ga based heuristic for the vehicle routing problem with multiple trips. *Journal of Mathematical Modelling and Algorithms*, 6(4):591–613, 2007.

F. A. Santos, G. R. Mateus, and A. S. da Cunha. A branch-and-price algorithm for a vehicle routing problem with cross-docking. *Electronic Notes in Discrete Mathematics*, 37:249–254, 2011.

F. A. Santos, G. R. Mateus, and A. S. Da Cunha. The pickup and delivery problem with cross-docking. *Computers & Operations Research*, 40(4):1085–1093, 2013.

M. W. P. Savelsbergh and M. M. Solomon. The general pickup and delivery problem. *Transportation Science*, 29:17–29, 1995.

M. Schneider, A. Stenger, and J. Hof. An adaptive vns algorithm for vehicle routing problems with intermediate stops. *OR Spectrum*, 37(2):353–387, 2015.

A. Subramanian, L. M. d. A. Drummond, C. Bentes, L. S. Ochi, and R. Farias. A parallel heuristic for the vehicle routing problem with simultaneous pickup and delivery. *Computers & Operations Research*, 37(11):1899–1911, 2010.

E. D. Taillard, G. Laporte, and M. Gendreau. Vehicle routeing with multiple use of vehicles. *Journal of the Operational research society*, 47(8):1065–1070, 1996.

C. D. Tarantilis. Adaptive multi-restart tabu search algorithm for the vehicle routing problem with cross-docking. *Optimization letters*, 7(7):1583–1596, 2013.

C. D. Tarantilis, E. E. Zachariadis, and C. T. Kiranoudis. A hybrid guided local search for the vehicle-routing problem with intermediate replenishment facilities. *INFORMS Journal on Computing*, 20(1):154–168, 2008.

P. Toth and D. Vigo. An exact algorithm for the vehicle routing problem with backhauls. *Transportation science*, 31(4):372–385, 1997.

P. Toth and D. Vigo, editors. *The Vehicle Routing Problem*. SIAM Monographs on Discrete Mathematics and Applications. SIAM, Philadelphia, PA, 2002.

P. Toth and D. Vigo, editors. *Vehicle Routing: Problems, Methods, and Applications*. MOS-SIAM Series on Optimization. SIAM, Philadelphia, PA, 2014.

T. Vidal, T. G. Crainic, M. Gendreau, and C. Prins. Heuristics for multi-attribute vehicle routing problems: A survey and synthesis. *European Journal of Operational Research*, 231(1):1–21, 2013.

T. Vidal, T.G. Crainic, M. Gendreau, and C. Prins. A Unified Solution Framework for Multi-Attribute Vehicle Routing Problems. *European Journal of Operational Research*, 234(3):658–673, 2014.

T. Vidal, M. Battarra, A. Subramanian, and G. Erdogan. Hybrid metaheuristics for the clustered vehicle routing problem. *Computers & Operations Research*, 58:87–99, 2015.

N. Wassan. Reactive tabu adaptive memory programming search for the vehicle routing problem with backhauls. *Journal of the Operational Research Society*, 58(12):1630–1641, 2007.

N. Wassan, N. Wassan, G. Nagy, and S. Salhi. The multiple trip vehicle routing problem with backhauls: Formulation and a two-level variable neighbourhood search. *Computers & Operations Research*, 78:454–467, 2017.

M. Wen, J. Larsen, J. Clausen, J.-F. Cordeau, and G. Laporte. Vehicle routing with cross-docking. *Journal of the Operational Research Society*, 60(12):1708–1718, 2009.

A Notation

g	the main depot
m	number of vehicles
Q	vehicle capacity
F	vehicle fixed cost
T	planning horizon
$\mathcal{S}$	set of supply points
$[t(s)-\eta, t(s)]$	time window of supply point $s \in \mathcal{S}$
$\varphi(s)$	time for loading/unloading operations at $s \in \mathcal{S}$
$\mathcal{C}^D$	set of delivery-customer demands
$\mathcal{C}^P$	set of pickup-customer demands
$\mathcal{C}^D_s$	subset of delivery-customer demands served by $s \in S$ (delivery service zone of s)
$\mathcal{C}^P_s$	subset of pickup-customer demands served by $s \in S$ (pickup service zone of s)
q_i	quantity to be delivered or picked up at customer demand $i \in \mathcal{C}^D \cup \mathcal{C}^P$
$\delta(i)$	service time at customer demand $i \in \mathcal{C}^D \cup \mathcal{C}^P$
$[e_i, l_i]$	time window of customer $i \in \mathcal{C}^D \cup \mathcal{C}^P$
$\mathcal{W}$	set of waiting stations
σ	unit waiting-time cost
$\mathcal{G} = (\mathcal{V}, \mathcal{A})$	directed graph
c_{ij}	routing cost and time of arc $(i,j) \in \mathcal{A}$
$\mathcal{L}$	set of feasible starting, ending and inter-facility route legs
$\mathcal{L}^+(s)$	set of route legs starting from $s \in \mathcal{S} \cup \{g\}$
$\mathcal{L}^-(s)$	set of route legs ending at $s \in \mathcal{S} \cup \{g\}$
$\mathcal{L}^+(g)$	set of route legs starting from g
a_{il}	binary coefficient assuming value 1 if customer demand $i \in \mathcal{C}^D \cup \mathcal{C}^P$ is visited by leg $l \in \mathcal{L}$
x_l	binary variable assuming value 1 if leg $l \in \mathcal{L}$ belongs to a work assignment
$\mathcal{V}_l$	set of nodes visited by leg $l \in \mathcal{L}$
$\mathcal{V}^C_l$	set of customer demand nodes visited by leg $l \in \mathcal{L}$
$\mathcal{V}^S_l$	set of supply point nodes visited by leg $l \in \mathcal{L}$
$\mathcal{V}^W_l$	set of waiting station nodes visited by leg $l \in \mathcal{L}$
$\mathcal{A}_l$	set of arcs belonging to leg $l \in \mathcal{L}$
T_l	set of arrival time instants at each node $i \in \mathcal{V}_l$, $l \in \mathcal{L}$
θ_i	arrival time instant at node $i \in \mathcal{V}_l$
c_l	routing cost of leg $l \in \mathcal{L}$
w_l	waiting cost of leg $l \in \mathcal{L}$
π_l	routing and waiting cost of leg $l \in \mathcal{L}$
λ_i	dual variables associated with covering constraints, $i \in \mathcal{C}^D \cup \mathcal{C}^P$
μ	dual variable associated with the fleet size constraint
γ_s	dual variables associated with flow conservation constraints, $s \in \mathcal{S}$
$\mathcal{C}_3$	set of all subsets of customer demands of cardinality 3
$\mathcal{L}(C)$	set of legs visiting at least two customer demands in $C \in \mathcal{C}_3$
ρ_C	dual variables associated with zone-based subset-row inequalities, $C \in \mathcal{C}_3$
$\mathcal{C}_3^{intra,\mathcal{P}}(s)$	set of subsets of pickup customer demands of cardinality 3 that induce intra-zone inequalities, $s \in \mathcal{S}$
$\mathcal{C}_3^{intra,\mathcal{D}}(s)$	set of subsets of delivery customer demands of cardinality 3 that induce intra-zone inequalities, $s \in \mathcal{S}$
$\mathcal{C}_3^{inter}(s,s')$	set of subsets of customer demands of cardinality 3 that induce inter-zone inequalities, $s, s' \in \mathcal{S}$
$\xi_{\mathcal{P}_s}$	dual variables associated with rounded capacity inequalities on a pickup service zone
$\xi_{\mathcal{D}_s}$	dual variables associated with rounded capacity inequalities on a delivery service zone
τ	time consumption associated with a label
ϕ	forward shift associated with a label
RC	reduced cost associated with a label
χ	vehicle load associated with a label
Ψ	set of visited customers associated with a label