An exact solution framework for multitrip vehicle-routing problems with time windows

Paradiso, Rosario; Roberti, Roberto; Lagana, Demetrio; Dullaert, Wout

published in
Operations Research
2020

DOI (link to publisher)
10.1287/OPRE.2019.1874

document version
Publisher's PDF, also known as Version of record

Operations Research

Publication details, including instructions for authors and subscription information:
http://pubsonline.informs.org

An Exact Solution Framework for Multitrip Vehicle-Routing Problems with Time Windows

Rosario Paradiso, Roberto Roberti, Demetrio Laganá, Wout Dullaert

To cite this article:
Rosario Paradiso, Roberto Roberti, Demetrio Laganá, Wout Dullaert (2020) An Exact Solution Framework for Multitrip Vehicle-Routing Problems with Time Windows. Operations Research 68(1):180-198. https://doi.org/10.1287/opre.2019.1874

http://pubsonline.informs.org/journal/opre

OPERATIONS RESEARCH
Vol. 68, No. 1, January–February 2020, pp. 180–198
ISSN 0030-364X (print), ISSN 1526-5463 (online)

Contextual Areas

An Exact Solution Framework for Multitrip Vehicle-Routing Problems with Time Windows

Rosario Paradiso,[a] Roberto Roberti,[b] Demetrio Laganá,[c] Wout Dullaert[b]

[a] Department of Mathematics and Computer Science, University of Calabria, 87036 Arcavacata di Rende CS, Italy; [b] Department of Supply Chain Analytics, Vrije Universiteit Amsterdam, 1081 HV Amsterdam, Netherlands; [c] Department of Mechanical, Energy and Management Engineering, University of Calabria, 87036 Arcavacata di Rende CS, Italy

Contact: rosario.paradiso@unical.it (RP); r.roberti@vu.nl, https://orcid.org/0000-0002-2987-1593 (RR); demetrio.lagana@unical.it, https://orcid.org/0000-0003-2618-6715 (DL); wout.dullaert@vu.nl (WD)

Received: December 20, 2018
Revised: March 20, 2019
Accepted: April 3, 2019
Published Online in Articles in Advance: January 2, 2020

Subject Classifications: programming: integer: algorithms: branch-and-bound; programming: integer: applications; transportation: vehicle routing
Area of Review: Transportation

https://doi.org/10.1287/opre.2019.1874

Abstract. *Multitrip vehicle-routing problems* (MTVRPs) generalize the well-known VRP by allowing vehicles to perform multiple trips per day. MTVRPs have received a lot of attention lately because of their relevance in real-life applications—for example, in city logistics and last-mile delivery. Several variants of the MTVRP have been investigated in the literature, and a number of exact methods have been proposed. Nevertheless, the computational results currently available suggest that MTVRPs with different side constraints require ad hoc formulations and solution methods to be solved. Moreover, solving instances with just 25 customers can be out of reach for such solution methods. In this paper, we proposed an exact solution framework to address four different MTVRPs proposed in the literature. The exact solution framework is based on a novel formulation that has an exponential number of variables and constraints. It relies on column generation, column enumeration, and cutting plane. We show that this solution framework can solve instances with up to 50 customers of four MTVRP variants and outperforms the state-of-the-art methods from the literature.

Supplemental Material: The online appendix is available at https://doi.org/10.1287/opre.2019.1874.

Keywords: multitrip vehicle routing • time windows • column generation • exact methods • dynamic programming

1. Introduction

Most of the literature on the *vehicle-routing problem* (VRP) addresses problems where each vehicle is limited to perform at most one trip per day. The first attempt to investigate VRPs where vehicles are allowed to perform multiple trips dates back to Fleischmann (1990). Since then, many contributions on *multitrip vehicle-routing problems* (MTVRPs) have been published, especially in the last decade, as observed in the recent survey of Cattaruzza et al. (2016b). Such an increasing interest in MTVRPs is due, for example, to the need of new practices in city logistics and last-mile delivery. The demand of limiting noise and pollution in city centers requires the usage of small vans, electric vehicles, and/or unmanned aerial vehicles (commonly known as drones) and forbids heavy large trucks from entering city centers. The limited capacity and autonomy of these small vehicles force them to perform multiple trips and to return to the depot to reload multiple times over the day.

In the literature, MTVRPs with different features are addressed in different papers, and a wide range of solution methods (both exact and heuristic) have been proposed. Nevertheless, we can identify a common underlying problem, the *capacitated MTVRP with time windows* (CMTVRPTW), that is a special case of the problems investigated in many papers, such as Hernandez et al. (2014, 2016), Cattaruzza et al. (2016a), and Cheng et al. (2018). The features of this CMTVRPTW are the following: (a) The goal is to minimize the routing costs; (b) all customers must be served; (c) multiple homogeneous vehicles are available; (d) vehicles are capacitated; and (e) time window constraints are imposed on the customer visits. Yet different papers consider additional side constraints on top of the CMTVRPTW, such as loading times to reload the vehicles at the depot (Hernandez et al. 2014, 2016), limited trip duration (Hernandez et al. 2014), release date on the availability of the goods to deliver (Cattaruzza et al. 2016a), drone battery capacity, and load-dependent battery consumption (Cheng et al. 2018). These four side constraints give rise to four different problems—namely, the *CMTVRPTW with loading times* (CMTVRPTW-LT), the *CMTVRPTW with limited trip duration* (CMTVRPTW-LD), the *CMTVRPTW with release dates* (CMTVRPTW-R),

and the *drone-routing problem* (DRP), respectively. For the sake of brevity, in the following, we will refer to these four problems as "the four variants of the CMTVRPTW."

The exact methods currently available for these four variants of the CMTVRPTW are based on several different mathematical formulations, which makes it unclear which formulation is most suitable to solve a CMTVRPTW. Moreover, in spite of the effort devoted to develop exact solution methods for CMTVRPTWs, the literature indicates that small instances with 25 customers cannot be consistently solved, and it could take a few hours of computing time to solve them.

With this paper, we aim at closing part of this research gap. We propose an *exact solution framework* (hereafter referred to as ESF) based on a novel mathematical model that can solve medium-sized instances with up to 50 customers of the four variants of the CMTVRPTW, significantly outperforming the state-of-the-art exact methods tailored for the single variants. We describe the ESF by focusing on the CMTVRPTW, and we show later how it can be tailored to solve the four variants by simply adapting one of its steps. The main contributions of this paper are the following: (a) We propose a novel mathematical model with an exponential number of variables and constraints that is valid for the CMTVRPTW and its four variants; (b) we describe two relaxations of this mathematical model that provide good lower bounds and can be efficiently computed; (c) we describe a seven-step ESF for the CMTVRPTW based on these two lower bounds and on a branch-and-cut algorithm, with an embedded branch-and-price to separate violated inequalities, to close the gap; (d) we illustrate how the ESF can be applied to solve each of the four variants of the CMTVRPTW by simply specializing one of the seven steps; and (e) we provide a computational proof that the ESF can solve instances with up to 50 customers and outperforms the state-of-the-art exact methods for the four individual variants of the CMTVRPTW.

The paper is organized as follows. Section 2 reviews the main contributions from the literature on exact methods for MTVRP with time windows. Section 3 formally introduces the CMTVRPTW and presents two column-generation-based formulations from the literature. Section 4 describes the novel formulation for the CMTVRPTW and its variants. Section 5 presents two lower bounds derived from the novel formulation. Section 6 provides an outline of the ESF and describes how to apply it to solve the CMTVRPTW. Section 7 describes the steps of the ESF in detail. Section 8 illustrates how the ESF can be tailored to solve each of the four variants of the CMTVRPTW. A computational analysis to show the effectiveness of the ESF and a comparison of its performance with the literature is provided in Section 9. Finally, conclusions are drawn in Section 10.

2. Literature Review

This section reviews the main exact methods proposed for MTVRPs with time windows. For the sake of brevity, we omit contributions on exact methods for MTVRPs without time windows and on heuristic methods. For a recent overview of the literature on these two topics, the reader is referred to Cattaruzza et al. (2016b).

In the remainder of the paper, we refer to a *structure* as a sequence of customers that can be visited consecutively by a vehicle between two visits at the depot, such that capacity constraints (and possibly other side constraints) are fulfilled and a departure time from the depot can be scheduled in such a way that all time windows are satisfied. Moreover, following the convention of Cattaruzza et al. (2016b), we refer to a *trip* as a structure with a fixed departure time from the depot, and we refer to a *journey* as a sequence of nonoverlapping trips assigned to the same vehicle. It is worth mentioning that, as stated in Cattaruzza et al. (2016b), different authors use different terms to refer to trips and journeys.

Azi et al. (2007) study a single-vehicle variant of the CMTVRPTW where it is not mandatory to serve all customers and the objective function is first to maximize the number of customers served and second to minimize the routing cost. Moreover, a limited trip-duration constraint (called deadline constraint) on the maximum time that the goods can stay on board before they are delivered at customers and a setup time to load the vehicle are considered. They propose a two-phase exact algorithm. In the first phase, all feasible nondominated trips are generated by complete enumeration, which is possible when time windows and limited trip-duration constraints are tight. In the second phase, feasible routes are generated by combining the trips generated in the first phase. The algorithm is tested on a set of instances adapted from the well-known Solomon instances for the VRPTW (Solomon 1987) with up to 100 customers. The results show that the algorithm is very sensitive to the limited trip-duration constraint. Indeed, if this constraint is not tight, it is impossible to enumerate all feasible trips in the first phase.

Azi et al. (2010) investigate the multivehicle version of the problem considered in Azi et al. (2007). The authors propose a branch-and-price algorithm based on a set-packing formulation, where each column represents a feasible journey. The pricing problem corresponds to an *elementary shortest-path problem with resource constraints* (ESPPRC). The algorithm is tested on instances with 25 and 40 customers derived from the Solomon VRPTW instances. The results show that instances with 25 customers can be routinely solved,

but the complexity of the problem strongly depends on the tightness of limited trip-duration constraints.

Macedo et al. (2011) study the same problem of Azi et al. (2010). They propose an iterative two-phase algorithm. In the first phase, all feasible trips are generated. The second phase corresponds to a pseudo-polynomial network flow model where nodes represent time instants and arcs represent feasible vehicle trips and is solved using a commercial solver. To create the network, a time discretization is needed. The model is iteratively solved until the current discretization provides a feasible solution. Computational results show that the proposed algorithm outperforms the two-phase method of Azi et al. (2010).

Hernandez et al. (2014) address the CMTVRPTW-LD, which differs from the problem considered in Azi et al. (2010) and Macedo et al. (2011) in that all customers must be served and the objective function aims at minimizing the total routing cost. They propose a two-phase exact algorithm. In the first phase, all feasible structures are enumerated by considering resource constraints. The second phase is a branch-and-price algorithm based on a set covering formulation with side constraints, where columns represent trips and side constraints guarantee that each vehicle is assigned nonoverlapping trips. The pricing problem can be solved in pseudo-polynomial time. The algorithm is tested on instances with 25 and 40 customers as in Azi et al. (2010) and Macedo et al. (2011). The computational results show that the algorithm of Hernandez et al. (2014) performs, on average, better than the method of Azi et al. (2010) and, on some instances, better than the method of Macedo et al. (2011).

The problem studied in Hernandez et al. (2016) differs from the problem of Hernandez et al. (2014) in that limited trip duration is not considered. Two branch-and-price exact algorithms are proposed, both based on a set covering formulation. In the first formulation, each column represents a journey, and the pricing problem is an ESPPRC, which can be solved by dynamic programming. In the second formulation, each column represents a trip, and additional side constraints guarantee the feasibility of the solutions; the pricing problem can again be solved by dynamic programming, where the concepts of group of labels and representative labels are exploited. Computational results on 25-customer instances show that the second branch-and-price, based on the trip-formulation, performs better than the first.

Recent contributions in the literature investigate the usage of truck–drone tandem systems and drone-only systems to deliver parcels. In particular, drones can perform multiple trips to deliver parcels by flying from/to trucks and depots where they can recharge their battery and pick up packages to deliver. The first significant contribution devoted to exact methods for drone-routing problems is owed to Cheng et al. (2018). They propose two mathematical formulations to solve the (multitrip) drone-routing problem. The objective function also takes energy consumption into account in the routing cost. The first formulation is based on drone-index variables, whereas the second formulation does not use such a drone index. Both formulations are strengthened by valid inequalities. The authors develop two branch-and-cut algorithms that are able to solve new benchmark instances with up to 50 customers. Their computational study shows that the formulation without the drone index outperforms the one with the drone index in terms of number of instances solved to optimality and computing time.

3. Description of the CMTVRPTW and Formulations from the Literature

In this section, we formally introduce the CMTVRPTW, which is used in the following sections to describe the ESF, and describe the two column-generation-based formulations from the literature.

The CMTVRPTW can be represented on a directed graph $G = (V, A)$. The vertex set $V = \{0\} \cup N$ consists of $n + 1$ vertices, where 0 represents the depot and $N = \{1, 2, \ldots, n\}$ represents a set of n customers to serve. For each customer $i \in N$, the demand q_i, the service time st_i, and a (hard) time window $[a_i, b_i]$ are given. A time window $[a_0, b_0]$ is associated with the depot. For the depot, we assume that $q_0 = 0$ and $st_0 = 0$. The arc set A is defined as $A = \{(i, j) \mid i, j \in V : a_i + t_{ij} + st_i \le b_j\}$, where t_{ij} is the travel time from vertex i to vertex j—without loss of generality, we assume that $t_{ij} \le t_{ik} + st_k + t_{kj}$, for all $i, j, k \in V$ such that $i \neq j \neq k$. A travel cost c_{ij} is also associated with each arc $(i, j) \in A$. Customers can be served by using a fleet of m vehicles, each one of capacity $\overline{q}$, that are located at the depot.

A trip h is represented as $h = (0, i_1, i_2, \ldots, i_{\mu_h}, 0)$, where μ_h is the number of visited customers. A journey r is represented as a sequence of nonoverlapping trips $r = (0, i_1, i_2, \ldots, i_{\mu_{r_1}}, 0, i_1, i_2, \ldots, i_{\mu_{r_2}}, 0, \ldots, 0)$.

The goal of the CMTVRPTW is to find a set of at most m journeys of minimum total cost such that each customer is visited exactly once.

In the literature, two formulations with an exponential number of variables have been proposed: a trip-based formulation (see Hernandez et al. 2016) and a journey-based formulation (see Hernandez et al. 2014, 2016).

Let $\mathcal{H}$ be the set of all feasible trips, and let c_h be the cost of trip $h \in \mathcal{H}$ given by the sum of the traversed arcs. A trip $h \in \mathcal{H}$ is described by coefficients α_{ih} indicating the number of times trip h visits customer $i \in N$ and by binary coefficients τ_{th} that are equal to 1 if trip h is *active* at time $t \in [a_0, b_0]$, where *active* means

that the vehicle performing trip h is either traveling between two vertices or serving/waiting at a customer. Let $x_h \in \{0,1\}$ be a binary variable equal to 1 if trip $h \in \mathcal{H}$ is selected (0 otherwise). The trip-based formulation, $F_{\mathcal{H}}$, is the following:

$$z(F_{\mathcal{H}}) = \min \sum_{h\in\mathcal{H}} c_h x_h, \tag{1a}$$

$$\text{s.t.} \sum_{h\in\mathcal{H}} \alpha_{ih} x_h = 1 \qquad i \in N, \tag{1b}$$

$$\sum_{h\in\mathcal{H}} \tau_{th} x_h \le m \quad t \in [a_0, b_0], \tag{1c}$$

$$x_h \in \{0,1\} \qquad h \in \mathcal{H}. \tag{1d}$$

The objective function (1a) aims at minimizing the cost of the selected trips. Constraints (1b) ensure that each customer is visited exactly once. Constraints (1c) guarantee that at most, m trips are active at any point in time. Constraints (1d) are integrality constraints.

Let $\mathcal{R}$ be the set of all feasible journeys. Moreover, let c_r be the cost of journey $r \in \mathcal{R}$ given by the sum of the costs of its individual trips, and let α_{ir} be a coefficient indicating the number of times journey $r \in \mathcal{R}$ visits customer $i \in N$. Let $x_r \in \{0,1\}$ be a binary variable equal to 1 if journey $r \in \mathcal{R}$ is selected (0 otherwise). The journey-based formulation, $F_{\mathcal{R}}$, is the following:

$$z(F_{\mathcal{R}}) = \min \sum_{r\in\mathcal{R}} c_r x_r, \tag{2a}$$

$$\text{s.t.} \sum_{r\in\mathcal{R}} \alpha_{ir} x_r = 1 \quad i \in N, \tag{2b}$$

$$\sum_{r\in\mathcal{R}} x_r \le m, \tag{2c}$$

$$x_r \in \{0,1\} \qquad r \in \mathcal{R}. \tag{2d}$$

The objective function (2a) aims at minimizing the cost of the selected journeys. Constraints (2b) ensure that each customer is visited exactly once. Constraint (2c) guarantees that at most m journeys are selected. Constraints (2d) are integrality constraints.

Both formulations $F_{\mathcal{H}}$ and $F_{\mathcal{R}}$ have an exponential number of variables that cannot be enumerated a priori, even for small instances. Therefore, both formulations can be used to solve the CMTVRPTW only if a column-generation framework is applied. Earlier contributions from the literature (see, e.g., Azi et al. 2010 and Hernandez et al. 2014, 2016) show that the lower bound provided by the linear relaxation of both formulations is of high quality. Nevertheless, solving the pricing problem of both formulations is particularly challenging, so it could take hours of computing time to find an optimal solution, even for instances with only 25 customers.

4. The Novel Structure-Based Formulation

Because of the computational complexity of using formulations $F_{\mathcal{H}}$ and $F_{\mathcal{R}}$, we introduce a novel structure-based formulation that has an exponential number of variables and constraints, but involves much fewer variables than both $F_{\mathcal{H}}$ and $F_{\mathcal{R}}$.

Let a *structure* $s = (0, i_1, i_2, \ldots, i_{\mu_s}, 0)$ be an ordered set of μ_s customers that can be visited by a vehicle in between two visits at the depot and can start from the depot within a time interval $[e_s, \ell_s]$, such that: (i) Capacity constraints are satisfied; (ii) the duration d_s and the cost c_s are constant for each departure time from the depot within $[e_s, \ell_s]$; and (iii) the duration d_s is the minimum duration to serve the set of customers in the given order. Let $\mathcal{S}$ be the set of all feasible structures, and let α_{is} be a coefficient equal to the number of times customer $i \in N$ is served by structure $s \in \mathcal{S}$. Let $x_s \in \{0,1\}$ be a binary variable equal to 1 if structure $s \in \mathcal{S}$ is selected (0 otherwise). The structure-based formulation, $F_{\mathcal{S}}$, is the following:

$$z(F_{\mathcal{S}}) = \min \sum_{s\in\mathcal{S}} c_s x_s, \tag{3a}$$

$$\text{s.t.} \sum_{s\in\mathcal{S}} \alpha_{is} x_s = 1 \qquad i \in N, \tag{3b}$$

$$\sum_{s\in\widehat{\mathcal{S}}} x_s \le \eta_m(\widehat{\mathcal{S}}) \qquad \widehat{\mathcal{S}} \subseteq \mathcal{S}, \tag{3c}$$

$$x_s \in \{0,1\} \qquad s \in \mathcal{S}, \tag{3d}$$

where $\eta_m(\widehat{\mathcal{S}})$ is the maximum number of structures of the set $\widehat{\mathcal{S}}$ that can be simultaneously in a solution given the number of vehicles m.

The objective function (3a) aims at minimizing the cost of the selected structures. Constraints (3b) ensure that each customer is visited exactly once. Constraints (3c) (hereafter called *structure feasibility constraints* (SFCs)) guarantee that the set of selected structures can be performed by the m vehicles. Constraints (3d) are integrality constraints.

Notice that the number of variables of $F_{\mathcal{S}}$ can be significantly lower than the number of variables of $F_{\mathcal{H}}$ and, in turn, of $F_{\mathcal{R}}$. Indeed, formulation $F_{\mathcal{H}}$ has a binary variable for each structure $s \in \mathcal{S}$ and each instant of time $t \in [e_s, \ell_s]$. Moreover, each feasible solution of $F_{\mathcal{S}}$ can correspond to multiple (possibly an infinite number of) solutions of formulation $F_{\mathcal{H}}$. Indeed, given the set of structures of a feasible solution of $F_{\mathcal{S}}$, it is possible to obtain a feasible solution of $F_{\mathcal{H}}$ by fixing the departure time from the depot of each of these structures within the corresponding time interval $[e_s, \ell_s]$.

5. Lower Bounds Based on the Structure-Based Formulation

In this section, we describe two lower bounds derived from formulation $F_{\mathcal{S}}$ that are used by the ESF and are described in the following sections. The first lower bound is based on a continuous relaxation of $F_{\mathcal{S}}$ that ignores SFC and therefore provides a lower bound

for the CVRPTW without the multitrip feature. The second lower bound is based on a relaxed version of SFC that can be enumerated by inspection.

5.1. First Lower Bound: CVRPTW Lower Bound

The first lower bound, which we call CVRPTW-LB in the following, corresponds to the optimal solution of the linear relaxation of formulation $F_{\mathcal{S}}$ without SFC—that is, it corresponds to the optimal value $z(P1_{\mathcal{S}})$ of the following problem $P1_{\mathcal{S}}$:

$$z(P1_{\mathcal{S}}) = \min \sum_{s\in\mathcal{S}} c_s x_s, \tag{4a}$$

$$\text{s.t.} \sum_{s\in\mathcal{S}} \alpha_{is} x_s = 1 \qquad i \in N, \tag{4b}$$

$$x_s \in \mathbb{R}_+ \qquad s \in \mathcal{S}. \tag{4c}$$

As $P1_{\mathcal{S}}$ is the continuous relaxation of a set-partitioning problem that does not consider SFC, it provides a lower bound to a VRP with capacity and time window constraints.

5.2. Second Lower Bound: Relaxed SFC Lower Bound

The second lower bound, which we call RSFC-LB in the following, is obtained from $F_{\mathcal{S}}$ by replacing SFC with a relaxed version and by adding a set of valid inequalities as follows.

Let $\tau_{hs} \in \{0,1\}$ be a binary coefficient defined for each structure $s \in \mathcal{S}$ and each instant of time $h \in [a_0, b_0]$ as follows:

$$\tau_{hs} = \begin{cases} 1 & \text{if } \ell_s < h < e_s + d_s \\ 0 & \text{otherwise;} \end{cases}$$

that is, τ_{hs} is equal to 1 if structure $s \in \mathcal{S}$ is active at time h for any possible departure time from the depot. A relaxed version of SFC (hereafter referred to as *relaxed SFC*, or, in short, RSFC) is therefore given by

$$\text{(RSFC)} \qquad \sum_{s\in\mathcal{S}} \tau_{hs} x_s \le m \quad h \in [a_0, b_0], \tag{5}$$

which indicate that, at any point in time, at most m structures (and thus vehicles) can be active.

To better explain constraints (5), let us consider two structures $s_1, s_2 \in \mathcal{S}$ with the following features: $[e_{s_1}, \ell_{s_1}] = [10, 20]$, $d_{s_1} = 15$, $[e_{s_2}, \ell_{s_2}] = [18, 22]$, $d_{s_2} = 10$. The only coefficients τ_{hs} equal to 1, for structure s_1, are τ_{21,s_1}, τ_{22,s_1}, and τ_{23,s_1}, τ_{24,s_1}, and, for structure s_2 τ_{23,s_2}, τ_{24,s_2}, τ_{25,s_2}, τ_{26,s_2}, and τ_{27,s_2}. Therefore, of all constraints (5) with $h \in [21, 27]$, the tightest constraints are those with $h = 23, 24$ that are $x_{s_1} + x_{s_2} \le m$.

RSFC-LB also uses a subset of the well-known *subset-row* (SR) inequalities, introduced by Jepsen et al. (2008) for the VRPTW. In particular, for each triplet of customers $\{i, j, k\} \in N$ and for each structure $s \in \mathcal{S}$, let β_{ijks} be a binary coefficient defined as

$$\beta_{ijks} = \begin{cases} 1 & \text{if } \alpha_{is} + \alpha_{js} + \alpha_{ks} \ge 2 \\ 0 & \text{otherwise,} \end{cases}$$

which is, β_{ijks} is equal to 1 if at least two of the customers of the set $\{i, j, k\}$ are visited by structure s (0 otherwise). SR inequalities are defined as

$$\sum_{s\in\mathcal{S}} \beta_{ijks} x_s \le 1 \quad \{i, j, k\} \in N. \tag{6}$$

RSFC-LB corresponds to the optimal value $z(P2_{\mathcal{S}})$ of the following problem $P2_{\mathcal{S}}$:

$$z(P2_{\mathcal{S}}) = \min \sum_{s\in\mathcal{S}} c_s x_s, \tag{7a}$$

$$\text{s.t.} \sum_{s\in\mathcal{S}} \alpha_{is} x_s = 1 \qquad i \in N, \tag{7b}$$

$$\sum_{s\in\mathcal{S}} \tau_{hs} x_s \le m \qquad h \in [a_0, b_0], \tag{7c}$$

$$\sum_{s\in\mathcal{S}} \beta_{ijks} x_s \le 1 \qquad \{i, j, k\} \in N, \tag{7d}$$

$$x_s \in \mathbb{R}_+ \qquad s \in \mathcal{S}. \tag{7e}$$

6. Outline of the ESF

This section provides an outline of the ESF described for the CMTVRPTW. We assume that a feasible solution of the CMTVRPTW exists. The algorithm uses the following notation:

- ub^* is the best upper bound (if any) found;
- ub_{guess} is a guess upper bound on the value of the optimal solution cost;
- gap_{guess} is a guess on the gap in percentage between the optimal CMTVRPTW solution cost and CVRPTW-LB;
- gap^0_{guess} is the initial value of gap_{guess};
- gap_{step} is the increase, at each iteration, of gap_{guess};
- *status* is the status of the solution process, which can take one of the following three values: *optimal* (i.e., an optimal solution of cost ub^* was found), *feasible* (i.e., a feasible solution of cost ub^* was found, but it may not be optimal), and *nil* (i.e., no solution has been found yet).

The algorithm consists of the following seven steps:

1. *Initialization*: Set $status = nil$ and $gap_{guess} = gap^0_{guess}$;
2. *CVRPTW-LB Computation* (see Section 5.1 for an overview and Section 7.1 for the details): Solve problem $P1_{\mathcal{S}}$ by column generation and compute its optimal solution cost $z(P1_{\mathcal{S}})$; set $ub_{guess} = z(P1_{\mathcal{S}}) * (1 + gap_{guess})$;
3. *Structure Enumeration* (see Section 7.2 for the details): Enumerate the set $\mathcal{S}_1$ of all the structures

having reduced costs not greater than $ub_{guess} - z(\mathrm{P1}_{\mathcal{S}})$ with respect to the dual solution corresponding to lower bound $z(\mathrm{P1}_{\mathcal{S}})$;

4. *RSFC-LB Computation* (see Section 5.2 for an overview and Section 7.3 for the details): Solve problem $\mathrm{P2}_{\mathcal{S}_1}$ obtained from $\mathrm{P2}_{\mathcal{S}}$ by replacing the set of structures $\mathcal{S}$ with its subset $\mathcal{S}_1$. Let $z(\mathrm{P2}_{\mathcal{S}_1})$ be the optimal cost of $\mathrm{P2}_{\mathcal{S}}$.

5. *Structure Reduction* (see Section 7.4): Compute the set of structures $\mathcal{S}_2 \subseteq \mathcal{S}_1$ obtained from $\mathcal{S}_1$ by removing all the structures having reduced cost greater than $ub_{guess} - z(\mathrm{P2}_{\mathcal{S}_1})$ with respect to the dual solution of cost $z(\mathrm{P2}_{\mathcal{S}_1})$ of problem $\mathrm{P2}_{\mathcal{S}_1}$.

6. *Branch-and-Cut to Close the Gap* (see Section 7.5): By using a branch-and-cut method, solve problem $\mathrm{F}_{\mathcal{S}_2}$ obtained from $\mathrm{F}_{\mathcal{S}}$ by replacing the set of structures $\mathcal{S}$ with its subset $\mathcal{S}_2$. If $\mathrm{F}_{\mathcal{S}_2}$ contains feasible solutions, let $z(\mathrm{F}_{\mathcal{S}_2})$ be the cost of an optimal solution.

7. *Iterative Step*: There are three possible cases:

(a) A feasible solution of $\mathrm{F}_{\mathcal{S}_2}$ exists

(a1) If $z(\mathrm{F}_{\mathcal{S}_2}) \le ub_{guess}$: such a solution is an optimal CMTVRPTW solution; set $status = optimal$, $ub^* = z(\mathrm{F}_{\mathcal{S}_2})$, and terminate;

(a2) If $z(\mathrm{F}_{\mathcal{S}_2}) > ub_{guess}$: such a solution is a valid upper bound to the CMTVRPTW; set $status = feasible$, $ub^* = z(\mathrm{F}_{\mathcal{S}_2})$, $ub_{guess} = z(\mathrm{F}_{\mathcal{S}_2})$, and go to Step 3;

(b) $\mathrm{F}_{\mathcal{S}_2}$ does not have any feasible solution: set $gap_{guess} = gap_{guess} + gap_{step}$, $ub_{guess} = z(\mathrm{P1}_{\mathcal{S}}) * (1 + gap_{guess})$, and go to Step 3.

7. Detailed Description of Steps 2–6 of the ESF

7.1. Step 2: CVRPTW-LB Computation

Step 2 of the ESF requires solving problem $\mathrm{P1}_{\mathcal{S}}$ to compute its optimal value $z(\mathrm{P1}_{\mathcal{S}})$. As the set of structures $\mathcal{S}$ increases exponentially with the number of customers, problem $\mathrm{P1}_{\mathcal{S}}$ must be solved via column generation. The column-generation procedure we propose initializes the master problem with the single-customer structures and solves it via a general-purpose solver. At each iteration, at most *col_iter* (where *col_iter* is a parameter) negative reduced-cost structures are added to the master problem. Such negative reduced-cost structures are priced out by using dynamic programming as follows.

Let $u_i \in \mathbb{R}$ be the dual variable associated with constraint (4b) of customer $i \in N$. Let $\tilde{c}_{ij}$ be the reduced cost of arc $(i,j) \in A$ with respect to $\boldsymbol{u} \in \mathbb{R}^n$ defined as

$$\tilde{c}_{ij} = \begin{cases} c_{ij} - \frac{1}{2}(u_i + u_j) & \text{if } i \neq 0 \text{ and } j \neq 0 \\ c_{ij} - \frac{1}{2}u_j & \text{if } i = 0 \\ c_{ij} - \frac{1}{2}u_i & \text{otherwise (i.e., if } j = 0). \end{cases} \tag{8}$$

Let $f = (0, i_1, \ldots, i_{\mu_f} = i^f)$ be an elementary forward path that starts from the depot at time a_0, visits the set of customers $N^f = \{i_1, \ldots, i^f\}$ each within its time window, ends at customer i^f at time t^f, such that the total demand q^f of the visited customers does not exceed the vehicle capacity. Let $\tilde{c}^f$ be the reduced cost of path f, defined as the sum of the reduced costs of the traversed arcs.

We associate a label $L^f = (N^f, q^f, t^f, i^f, \tilde{c}^f)$ with each path $f = (0, i_1, \ldots, i^f)$. Because of capacity and time window constraints, a label $L^f = (N^f, q^f, t^f, i^f, \tilde{c}^f)$ is feasible only if $N^f \subseteq N$, $q^f \le \overline{q}$, $t^f \in [a_{i^f}, b_{i^f}]$, and $i^f \in N^f$. Clearly, each path $f = (0, i_1, \ldots, i^f)$ such that $i^f = 0$ and $\tilde{c}^f < 0$ corresponds to a negative reduced-cost structure.

To generate such structures, the following initialization and extension are needed:

Initialization: A single label $L^f = (N^f, q^f, t^f, i^f, \tilde{c}^f)$ is generated, where $N^f = \emptyset$, $q^f = 0$, $t^f = a_0$, $i^f = 0$, and $\tilde{c}^f = 0$.

Extension: Extend each label $L^f = (N^f, q^f, t^f, i^f, \tilde{c}^f)$ such that either (a) $i^f \in N$ or (b) $i^f = 0$ and $N^f = \emptyset$ toward any vertex $j \in V \setminus N^f$ and generate label $L^{f'} = (N^{f'}, q^{f'}, t^{f'}, i^{f'}, \tilde{c}^{f'})$, where

$$\begin{cases} N^{f'} = N^f \cup \{j\} \text{ (if } j \in N) \text{ or } N^{f'} = N^f \text{ (if } j = 0) \\ q^{f'} = q^f + q_j \\ t^{f'} = \max\{a_j, t^f + st_{i^f} + t_{i^f j}\} \\ i^{f'} = j \\ \tilde{c}^{f'} = \tilde{c}^f + \tilde{c}_{i^f j}. \end{cases}$$

To speed up the solution process and limit the number of labels to generate, the following dominance rule can be applied: Given two labels $L^{f_1} = (N^{f_1}, q^{f_1}, t^{f_1}, i^{f_1}, \tilde{c}^{f_1})$ and $L^{f_2} = (N^{f_2}, q^{f_2}, t^{f_2}, i^{f_2}, \tilde{c}^{f_2})$, L^{f_2} is dominated if $N^{f_1} \subseteq N^{f_2}$, $q^{f_1} \le q^{f_2}$, $t^{f_1} \le t^{f_2}$, $i^{f_1} = i^{f_2}$, and $\tilde{c}^{f_1} \le \tilde{c}^{f_2}$. Moreover, we also apply two other well-known techniques—that is, *ng*-path relaxation (see Baldacci et al. 2011) and completion bounds (see Baldacci et al. 2012)—to further limit the number of labels to generate. For the sake of brevity, we omit the details on this matter.

Let $\boldsymbol{u}^1 = (u_1^1, u_2^1, \ldots, u_n^1)$ be the optimal dual solution of cost $z(\mathrm{P1}_{\mathcal{S}})$ achieved by Step 2.

7.2. Step 3: Structure Enumeration

Step 3 requires enumerating the set $\mathcal{S}_1$ all the structures having reduced costs not greater than $ub_{guess} - z(\mathrm{P1}_{\mathcal{S}})$ with respect to the dual solution $\boldsymbol{u}^1$ of cost $z(\mathrm{P1}_{\mathcal{S}})$. The set $\mathcal{S}_1$ can be generated with the following dynamic programming recursion, which is similar to the recursions proposed by Hernandez et al. (2014, 2016), and Tilk and Irnich (2017).

Let $b = (0, i_1, \ldots, i_{\mu_b} = i^b)$ be an elementary backward path that can start from the depot not earlier than g^b, visits the set of customers N^b each within its time window, ends at customer i^b not later than ℓ^b, has a

duration equal to d^b, and such that the total demand of the visited customers is equal to q^b. Let $\tilde{c}^b$ be the reduced cost of path b with respect of the dual solution $\boldsymbol{u}^1$, given by the sum of the reduced costs of the traversed arcs computed as in (8).

With each path $b = (0, i_1, \ldots, i^b)$, we associate a label $L^b = (N^b, q^b, \ell^b, g^b, d^b, i^b, \tilde{c}^b)$. A label $L^b = (N^b, q^b, \ell^b, g^b, d^b, i^b, \tilde{c}^b)$ is feasible if the following conditions hold:

$$\begin{cases} N^b \subseteq N \\ q^b \le \overline{q} \\ a_{i^b} \le \ell^b \le b_{i^b} \\ a_0 \le g^b \le b_0 \\ i^b \in N^b. \end{cases} \tag{9}$$

A structure $s \in \mathcal{S}$ corresponds to an elementary backward path $b = (0, i_1, \ldots, i^b)$ (and the corresponding label $L^b = (N^b, q^b, \ell^b, g^b, d^b, i^b, \tilde{c}^b)$), such that $i^b = 0$, and where $\ell_s = \ell^b$, $e_s = g^b - d^b$, $d_s = d^b$, $c_s = \tilde{c}^b + \sum_{i \in N^b} u_i^1$, and $\alpha_{is} = 1$, $\forall i \in N^b$ (0 if $i \in N \setminus N^b$).

To generate the set of structures $\mathcal{S}_1$, the following initialization and extension are needed:

Initialization: A single label $L^b = (N^b, q^b, \ell^b, g^b, d^b, i^b, \tilde{c}^b)$ is generated, where $N^b = \emptyset$, $q^b = 0$, $\ell^b = b_0$, $g^b = a_0$, $d^b = 0$, $i^b = 0$, and $\tilde{c}^b = 0$.

Extension: Extend each label $L^b = (N^b, q^b, \ell^b, g^b, d^b, i^b, \tilde{c}^b)$ such that either (*a*) $i^b \in N$ or (*b*) $i^b = 0$ and $N^b = \emptyset$ toward any vertex $j \in V \setminus N^b$ to generate label $L^{b'} = (N^{b'}, q^{b'}, \ell^{b'}, g^{b'}, d^{b'}, i^{b'}, \tilde{c}^{b'})$, where

$$\begin{cases} N^{b'} = N^b \cup \{j\} \text{ (if } j \in N) \text{ or } N^{b'} = N^b \text{ (if } j = 0) \\ q^{b'} = q^b + q_j \\ \ell^{b'} = \min\{b_j, \ell^b - t_{ji^b} - st_j\} \\ g^{b'} = \max\{a_j + d^b + t_{ji^b} + st_j,\ g^b\} \\ d^{b'} = \max\{d^b + t_{ji^b} + st_j,\ g^b - b_j\} \\ i^{b'} = j \\ \tilde{c}^{b'} = \tilde{c}^b + \tilde{c}_{ji^b}. \end{cases} \tag{10}$$

To speed up the enumeration phase and limit the number of generated labels, the following dominance rule can be applied: Given two labels $L^{b_1} = (N^{b_1}, q^{b_1}, \ell^{b_1}, g^{b_1}, d^{b_1}, i^{b_1}, \tilde{c}^{b_1})$ and $L^{b_2} = (N^{b_2}, q^{b_2}, \ell^{b_2}, g^{b_2}, d^{b_2}, i^{b_2}, \tilde{c}^{b_2})$, L^{b_2} is dominated if $N^{b_1} = N^{b_2}$ (and thus $q^{b_1} = q^{b_2}$), $\ell^{b_1} \ge \ell^{b_2}$, $g^{b_1} \le g^{b_2}$, $d^{b_1} \le d^{b_2}$, $i^{b_1} = i^{b_2}$, and $\tilde{c}^{b_1} \le \tilde{c}^{b_2}$.

The number of labels can further be limited by using completion bounds based on the *ng*-path relaxation, as described in Baldacci et al. (2011, 2012).

7.3. Step 4: RSFC-LB Computation

In Step 4, problem $\mathrm{P2}_{\mathcal{S}_1}$ is solved to compute its optimal solution value $z(\mathrm{P2}_{\mathcal{S}_1})$. Problem $\mathrm{P2}_{\mathcal{S}_1}$ is obtained from $\mathrm{P2}_{\mathcal{S}}$ by replacing the set of structures $\mathcal{S}$ with its subset $\mathcal{S}_1$ generated in Step 3. As the set of columns is known, no column generation is needed. At the beginning of Step 4, problem $\mathrm{P2}_{\mathcal{S}_1}$ is solved with a linear-programming solver without imposing constraints (7c) and (7d). These constraints are subsequently iteratively added in a cutting-plane fashion.

The separation of both (7c) and (7d) can easily be done by inspection, given the current optimal solution $\tilde{\boldsymbol{x}}$. Let $\tilde{\mathcal{S}} = \{s \in \mathcal{S} \mid \tilde{x}_s > 0\}$ be the set of structures in the solution $\tilde{\boldsymbol{x}}$. The violation of (7c) is checked only for instants of times $h \in [a_0, b_0]$ for which it exists $s \in \tilde{\mathcal{S}}$ such that $h = \ell_s$, whereas the violation of constraints (7d) is checked simply for all triplets of customers $\{i, j, k\} \in N$.

At the end of Step 4, an optimal dual solution $(\boldsymbol{u}^2, \boldsymbol{v}^2, \boldsymbol{w}^2)$ of $\mathrm{P2}_{\mathcal{S}_1}$ of cost $z(\mathrm{P2}_{\mathcal{S}_1})$ is available, where $\boldsymbol{u}^2$, $\boldsymbol{v}^2$, and $\boldsymbol{w}^2$ are the vectors of dual variables associated with constraints (7b), (7c), and (7d), respectively.

7.4. Step 5: Structure Reduction

Step 5 aims at deriving the set of structures $\mathcal{S}_2 \subseteq \mathcal{S}_1$ that is obtained from $\mathcal{S}_1$ by removing all structures with a reduced cost greater than $ub_{guess} - z(\mathrm{P2}_{\mathcal{S}_1})$ with respect to the dual solution $(\boldsymbol{u}^2, \boldsymbol{v}^2, \boldsymbol{w}^2)$ of cost $z(\mathrm{P2}_{\mathcal{S}_1})$ of problem $\mathrm{P2}_{\mathcal{S}_1}$ achieved at Step 4. The set $\mathcal{S}_2$ is easily derived by inspection of the set $\mathcal{S}_1$.

7.5. Step 6: Branch-and-Cut to Close the Gap

Step 6 attempts to find an optimal CMTVRPTW solution by applying a branch-and-cut method. In Steps 4 and 5, a limited set of structures $\mathcal{S}_2$ has been generated. This set contains optimal CMTVRPTW solutions under the assumption that ub_{guess} is a valid upper bound to the CMTVRPTW. Step 6 solves problem $\mathrm{F}_{\mathcal{S}_2}$ obtained from $\mathrm{F}_{\mathcal{S}}$ by replacing the set of structures $\mathcal{S}$ with its subset $\mathcal{S}_2$. It applies a branch-and-cut algorithm because all columns of $\mathrm{F}_{\mathcal{S}_2}$ are known, so only SFC constraints (3c) may be missing. The SFC constraints are added in a cutting-plane fashion as follows. Notice that as the separation problem is NP-hard; it is performed on integer solutions only.

Let us assume that we have an integer solution $\tilde{\boldsymbol{x}}$ of $\mathrm{F}_{\mathcal{S}_2}$, and we want to check its feasibility by separating violated SFC constraints. Let $\tilde{\mathcal{S}} \subseteq \mathcal{S}_2$ be the subset of structures in a solution (i.e., $\tilde{\mathcal{S}} = \{s \in \mathcal{S}_2 \mid \tilde{x}_s = 1\}$). The right-hand side of constraint (7c) corresponding to the set $\tilde{\mathcal{S}}$ is equal to the maximum number of structures of the set $\tilde{\mathcal{S}}$ that can be simultaneously in solution. The separation problem of determining if the structures of the set $\tilde{\mathcal{S}}$ represent a feasible CMTVRPTW solution can be then formulated as a *team orienteering problem with time windows* (TOPTW) (see, e.g., Vansteenwegen et al. 2009 and Archetti et al. 2014), where (a) each node is a structure to assign to a vehicle; (b) m vehicles are available; (c) each structure has a unit profit and a time window $[e_s, \ell_s]$; and (d) the goal is to maximize

the number of structures that can be assigned to the m vehicles, which is achieved by maximizing the profit of the TOPTW.

Therefore, the separation problem of SFC on the set $\tilde{\mathcal{S}}$ can be formulated on a support graph $\tilde{G} = (\tilde{V}, \tilde{A})$ defined as follows. The vertex set $\tilde{V}$ contains a node for each structure of $\tilde{\mathcal{S}}$ plus a dummy node o (i.e., $\tilde{V} = \{o\} \cup \tilde{\mathcal{S}}$). The arc set $\tilde{A}$ is defined as $\tilde{A} = \{(o,s) | s \in \tilde{\mathcal{S}}\} \cup \{(s_1,s_2) | s_1,s_2 \in \tilde{\mathcal{S}}, s_1 \neq s_2, e_{s_1} + d_{s_1} \leq \ell_{s_2}\} \cup \{(s,o) | s \in \tilde{\mathcal{S}}\}$. With each arc $(s_1,s_2) \in \tilde{A}$, we associate a travel time $\tilde{t}_{s_1 s_2}$ defined as $\tilde{t}_{s_1 s_2} = d_{s_1}$ if $s_1 \neq o$, and $\tilde{t}_{s_1,s_2} = 0$ if $s_1 = o$. For example, let us assume that $\tilde{S} = \{s_1, s_2, s_3, s_4, s_5, s_5, s_6\}$ and $e_{s_1} = \ell_{s_1} = 119.7$, $d_{s_1} = 145.6$; $e_{s_2} = 264$, $\ell_{s_2} = 276$, $d_{s_2} = 136.3$; $e_{s_3} = \ell_{s_3} = 415.5$, $d_{s_3} = 166.5$; $e_{s_4} = \ell_{s_4} = 607.4$, $d_{s_4} = 240.8$; $e_{s_5} = \ell_{s_5} = 142.1$, $d_{s_5} = 257.1$; $e_{s_6} = \ell_{s_6} = 437.9$, $d_{s_6} = 298.1$. The corresponding support graph $\tilde{G}$ is represented in Figure 1.

Let $\tilde{\mathcal{R}}$ be the set of all elementary journeys of graph $\tilde{G}$ that start and end at node o and such that the sequence of visited nodes/structures can be assigned to the same vehicle. For each journey $r = (s_1, s_2, \ldots, s_{\mu_r}) \in \tilde{\mathcal{R}}$, let α_{sr} be a binary coefficient equal to 1 if structure $s \in \tilde{\mathcal{S}}$ is assigned to journey r. Moreover, let $y_r \in \{0,1\}$ be a binary variable equal to 1 if journey $r \in \tilde{\mathcal{R}}$ is selected (0 otherwise). The separation problem of SFC can be formulated as

$$\eta_m(\tilde{\mathcal{S}}) = \max \sum_{r \in \tilde{\mathcal{R}}} \left(\sum_{s \in \tilde{\mathcal{S}}} \alpha_{sr} \right) y_r, \tag{11a}$$

$$\text{s.t.} \sum_{r \in \tilde{\mathcal{R}}} \alpha_{sr} y_r \leq 1 \qquad s \in \tilde{\mathcal{S}}, \tag{11b}$$

$$\sum_{r \in \tilde{\mathcal{R}}} y_r \leq m, \tag{11c}$$

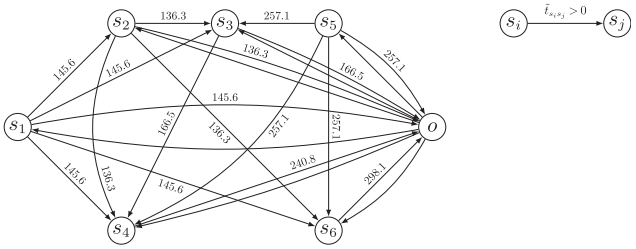
$$y_r \in \{0,1\} \qquad r \in \tilde{\mathcal{R}}. \tag{11d}$$

The objective function (11a) aims at maximizing the number of structures used in the selected journeys. Constraints (11b) ensure that each structure is not assigned more than once. Constraint (11c) guarantees that at most m journeys are selected. Constraints (11d) are integrality constraints.

The cardinality of the set $\tilde{\mathcal{R}}$ clearly increases exponentially with the number of structures of the set $\tilde{\mathcal{S}}$, so problem (11a)–(11d) can be solved by using column generation. The master problem corresponds to the linear relaxation of (11a)–(11d). The pricing problem corresponds to finding journeys of the set $\tilde{\mathcal{R}}$ having negative reduced cost with respect to the dual solution $(\boldsymbol{u}, v)$ of the master problem, where $\boldsymbol{u} \in \mathbb{R}_+^{|\tilde{\mathcal{S}}|}$ and $v \in \mathbb{R}_+$ are the dual variables associated with constraints (11b) and (11c), respectively.

7.5.1. Solving the Pricing Problem. The pricing problem can be solved by using dynamic programming as follows. Let $f = (o, s_1, \ldots, s_{\mu_f} = s^f)$ be an elementary

Figure 1. Example of a Support Graph $\tilde{G} = (\tilde{V}, \tilde{A})$ with Six Structures



forward path of graph $\tilde{G}$ that starts from node o, visits the set of nodes $\tilde{S}^f = \{s_1, \ldots, s^f\}$, and ends at node s^f at time $\tilde{e}^f$. Let $\tilde{c}^f$ be the reduced cost of path f with respect to the dual solution $(\boldsymbol{u}, v)$. We associate a label $L^f = (\tilde{S}^f, s^f, \tilde{e}^f, \tilde{c}^f)$ with each path $f = (o, s_1, \ldots, s^f)$. A label $L^f = (\tilde{S}^f, s^f, \tilde{e}^f, \tilde{c}^f)$ is feasible if $\tilde{S}^f \subseteq \tilde{V}$, $s^f \in \tilde{S}^f$, and $\tilde{e}^f \in [e_{s^f}, \ell_{s^f}]$. Each label $L^f = (\tilde{S}^f, s^f, \tilde{e}^f, \tilde{c}^f)$ with $s^f = o$ and $\tilde{c}^f < 0$ corresponds to a negative reduced-cost journey.

To generate such negative reduced-cost journeys, the following initialization and extension are needed:

Initialization: A label $L^f = (\tilde{S}^f, s^f, \tilde{e}^f, \tilde{c}^f)$ is generated, where $\tilde{S}^f = \varnothing$, $s^f = o$, $\tilde{e}^f = a_0$, $\tilde{c}^f = -v$.

Extension: Each label $L^f = (\tilde{S}^f, s^f, \tilde{e}^f, \tilde{c}^f)$, for which either (*a*) $s^f \in \tilde{\mathcal{S}}$ or (*b*) $s^f = o$ and $\tilde{S}^f = \varnothing$ is extended toward any node $s \in \tilde{V} \setminus \tilde{S}^f$ and generates label $L^{f'} = (\tilde{S}^{f'}, s^{f'}, \tilde{e}^{f'}, \tilde{c}^{f'})$, where

- $\tilde{S}^{f'} = \tilde{S}^f \cup \{s\}$,
- $s^{f'} = s$,
- $\tilde{e}^{f'} = \max\{\tilde{e}^f + \tilde{t}_{s^f s}, e_s\}$,
- $\tilde{c}^{f'} = \tilde{c}^f + 1 - u_s$ (if $s \in \tilde{\mathcal{S}}$) or $\tilde{c}^{f'} = \tilde{c}^f$ (if $s = o$).

To speed up the solution process and limit the number of labels to generate, the following dominance rule can be applied: Given two labels $L^{f_1} = (\tilde{S}^{f_1}, s^{f_1}, \tilde{e}^{f_1}, \tilde{c}^{f_1})$ and $L^{f_2} = (\tilde{S}^{f_2}, s^{f_2}, \tilde{e}^{f_2}, \tilde{c}^{f_2})$, L^{f_2} is dominated if $\tilde{S}^{f_1} = \tilde{S}^{f_2}$, $s^{f_1} = s^{f_2}$, $\tilde{e}^{f_1} \leq \tilde{e}^{f_1}$, and $\tilde{c}^{f_1} \leq \tilde{c}^{f_1}$.

7.5.2. Branching Scheme. Given a fractional solution $\tilde{\boldsymbol{y}}$ of problem (11a)–(11d) three types of branching are performed in a hierarchical way to find an integer solution.

Branching on Structures: A binary branching is performed on the structure s' that is closest to be used 0.5 times—that is, $s' = \arg\min_{s \in \tilde{\mathcal{S}}(\tilde{\boldsymbol{y}})} \{| \sum_{r \in \tilde{\mathcal{R}}} \alpha_{sr} \tilde{y}_r - 0.5|\}$, where $\tilde{\mathcal{S}}(\tilde{\boldsymbol{y}}) = \{s \in \tilde{\mathcal{S}} \,|\, 0 < \sum_{r \in \tilde{\mathcal{R}}} \alpha_{sr} \tilde{y}_r < 1\}$. In the first branch, structure s' must be used—that is, $\sum_{r \in \tilde{\mathcal{R}}} \alpha_{s'r} \tilde{y}_r = 1$. In the second branch, structure s' cannot used—that is, $\sum_{r \in \tilde{\mathcal{R}}} \alpha_{s'r} \tilde{y}_r = 0$;

Branching on Arcs: A binary branching is performed on the arc $(s,s') \in \tilde{A}$ that is closest to be traversed 0.5 times by the journeys in solution $\tilde{\boldsymbol{y}}$—that is, $(s,s') = \arg\min_{(s_1,s_2) \in \tilde{A}(\tilde{\boldsymbol{y}})} \{| \sum_{r \in \tilde{\mathcal{R}}} \gamma_{s_1 s_2 r} \tilde{y}_r - 0.5|\}$, where $\tilde{A}(\tilde{\boldsymbol{y}}) = \{(s_1,s_2) \in \tilde{A} \,|\, 0 < \sum_{r \in \tilde{\mathcal{R}}} \gamma_{s_1 s_2 r} \tilde{y}_r < 1\}$ and $\gamma_{s_1 s_2 r}$ is the number of times arc $(s_1,s_2) \in \tilde{A}$ is traversed by journey

$r \in \tilde{\mathcal{R}}$. In the first branch, arc (s, s') is removed from the arc set $\tilde{A}$. In the second branch, arc (s, s') is forced to be in solution if structures s and s' are visited, which is achieved by removing (*a*) all arcs $\{(s_1, s') \in \tilde{A} \,|\, s_1 \neq s\}$ and $\{(s, s_2) \in \tilde{A} \,|\, s_2 \neq s'\}$ if $s, s' \in \tilde{\mathcal{S}}$, (*b*) all arcs $\{(s_1, s') \in \tilde{A} \,|\, s_1 \neq o\}$ if $s = o$, and (*c*) all arcs $\{(s, s_2) \in \tilde{A} \,|\, s_2 \neq o\}$ if $s' = o$.

Branching on the Number of Vehicles: A binary branching is performed on the number of selected journeys—that is, on $\sum_{r \in \tilde{\mathcal{R}}} \tilde{y}_r$. In the first branch, constraint $\sum_{r \in \tilde{\mathcal{R}}} y_r \leq \lfloor \sum_{r \in \tilde{\mathcal{R}}} \tilde{y}_r \rfloor$ is added. In the second branch, constraint $\sum_{r \in \tilde{\mathcal{R}}} y_r \geq \lceil \sum_{r \in \tilde{\mathcal{R}}} \tilde{y}_r \rceil$ is added.

In the search tree, the best-bound-first policy is applied to identify the next node to branch on.

7.5.3. Acceleration Techniques. To speed up the solution process, we embed three more features in the branch-and-price to solve problem (11a)–(11d):

- Relaxed SFC inequalities (7c) are added in a cutting-plane fashion and separated both on integer and fractional solutions as described in Section 7.3.
- The pricing problem is relaxed by dropping resource $\tilde{S}^f$, so nonelementary journeys are allowed.
- The computation of $\eta_m(\tilde{\mathcal{S}})$ is stopped as soon as the highest upper bound (let us call this upper bound $\overline{\eta}_m(\tilde{\mathcal{S}})$) of all unexplored nodes in the search tree is strictly less than $|\tilde{\mathcal{S}}|$. Indeed, this condition is sufficient to add constraint $\sum_{s \in \tilde{\mathcal{S}}} x_s \leq \lfloor \overline{\eta}_m(\tilde{\mathcal{S}}) \rfloor$, which cuts off the integer solution $\tilde{\boldsymbol{x}}$.

8. Tailoring the ESF to Solve the Four Variants of the CMTVRPTW

As mentioned in Section 1, the CMTVRPTW can be specialized by adding side constraints that determine the feasibility of the structures and therefore define the set $\mathcal{S}$. Notice that CVRPTW-LB remains a valid lower bound, even if these side constraints are added. Notice also that, once the set of structures $\mathcal{S}_1$ is enumerated in Step 3, the remaining steps are not affected by the constraints that define the feasibility of a structure, but just rely on having the set of feasible structures $\mathcal{S}_1$. Therefore, the ESF can be tailored to solve the four variants of the CMTVRPTW by simply modifying the dynamic programming recursion of Step 3 to enumerate the structures in the gap $ub_{guess} - z(\text{P1}_{\mathcal{S}})$. In the following sections, we describe how Step 3 (described in Section 7.2 for the CMTVRPTW) can be modified to tackle each of the four variants.

8.1. Structure Enumeration for the CMTVRPTW-LT

The CMTVRPTW-LT proposed by Hernandez et al. (2016) considers the time to load the vehicle before it can depart from the depot. In particular, given the loading time lt_i for the goods requested by customer $i \in N$, the total loading time of a vehicle performing a given structure is equal to the sum of the individual loading times of the customers in the structure. For each label $L^b = (N^b, q^b, \ell^b, g^b, d^b, i^b, \tilde{c}^b)$, let us define its total loading time lt^b as $lt^b = \sum_{i \in N^b} lt_i$.

The structure enumeration for the CMTVRPTW-LT requires just two changes with respect to the one for the CMTVRPTW. The first change is in the feasibility of a label, and the second change is in the extension of a label to the depot.

A label $L^b = (N^b, q^b, \ell^b, g^b, d^b, i^b, \tilde{c}^b)$ with $i^b \neq 0$ is feasible if conditions (9) are satisfied and the following condition holds:

$$\ell^b - t_{0i^b} - lt^b \geq a_0 \,. \tag{12}$$

Indeed, condition (12) ensures that there is enough time to close the path by returning to the depot and to reload the vehicle.

When a label $L^b = (N^b, q^b, \ell^b, g^b, d^b, i^b, \tilde{c}^b)$ is extended to a customer, extension functions (10) are still valid. Whereas when a label $L^b = (N^b, q^b, \ell^b, g^b, d^b, i^b, \tilde{c}^b)$ is extended to the depot to generate label $L^{b'} = (N^{b'}, q^{b'}, \ell^{b'}, g^{b'}, d^{b'}, i^{b'}, \tilde{c}^{b'})$, functions (10) are replaced with the following functions

$$\begin{cases} N^{b'} = N^b \\ q^{b'} = q^b \\ \ell^{b'} = \ell^b - t_{0i^b} - lt^b \\ g^{b'} = \max\{a_0 + d^b + t_{0i^b} + lt^b,\ g^b\} \\ d^{b'} = \max\{d^b + t_{0i^b} + lt^b,\ g^b - b_0\} \\ i^{b'} = 0 \\ \tilde{c}^{b'} = \tilde{c}^b + \tilde{c}_{0i^b}, \end{cases}$$

which take into account the total loading time of a structure in computing the time to leave from the depot, return to the depot, and in the total duration.

8.2. Structure Enumeration for the CMTVRPTW-LD

In the CMTVRPTW-LD studied by Hernandez et al. (2014), a limited trip duration $\overline{d}$ is considered, which represents a maximum limit on the time that goods can be on board the vehicle before being delivered. This does not include loading times, the service time of the last customer of a structure, or the travel time to return to the depot. We accept the definition of limited trip duration given by Hernandez et al. (2014), even though we believe it is misleading and it should be called differently, for example, *maximum goods travel time*; indeed, we think it is more intuitive to call *limited trip duration* the maximum amount of time that the vehicle can be traveling in each trip from the departure from the depot to the return to it.

The structure enumeration of the CMTVRPTW-LD differs from the one for the CMTVRPTW because an additional resource is necessary, and feasibility

conditions and dominance rules must take the limited trip duration into account.

With each backward path $b = (0, i_1, \ldots, i^b)$, we associate a label $L^b = (N^b, q^b, \ell^b, g^b, d^b, i^b, i_1^b, \tilde{c}^b)$, where i_1^b represents the first customer visited by the backward path. Let dg^b be the goods travel time computed as $dg^b = d^b - t_{0i_1} - st_{i_1} - lt^b$. A label is feasible if it satisfies conditions (9) and (12) and the additional condition $dg^b \leq \overline{d}$.

Moreover, dominance rules need to be adjusted as follows. Given two labels $L^{b_1} = (N^{b_1}, q^{b_1}, \ell^{b_1}, g^{b_1}, d^{b_1}, i^{b_1}, i_1^{b_1}, \tilde{c}^{b_1})$ and $L^{b_2} = (N^{b_2}, q^{b_2}, \ell^{b_2}, g^{b_2}, d^{b_2}, i^{b_2}, i_1^{b_2}, \tilde{c}^{b_2})$, L^{b_2} is dominated if $N^{b_1} = N^{b_2}$ (and thus $q^{b_1} = q^{b_2}$), $\ell^{b_1} \geq \ell^{b_2}$, $g^{b_1} \leq g^{b_2}$, $d^{b_1} \leq d^{b_2}$, $i^{b_1} = i^{b_2}$, $i_1^{b_1} = i_1^{b_2}$, and $\tilde{c}^{b_1} \leq \tilde{c}^{b_2}$.

8.3. Structure Enumeration for the CMTVRPTW-R

Cattaruzza et al. (2016a) introduce the concept of *release dates* in the CMTVRPTW and study the CMTVRPTW-R. Each customer $i \in N$ has an associated release date rd_i that indicates the instant of time when the goods to deliver to customer i become available at the depot; therefore, the vehicle that serves customer i cannot depart from the depot before rd_i.

Similar to the case of the CMTVRPTW-LT, the structure enumeration for the CMTVRPTW-R requires two changes with respect to the one for the CMTVRPTW: the first in the feasibility conditions of a label and the second change in the extension of a label to the depot.

Let us indicate with rd^b the maximum release date of the customers serve by a backward path $b = (0, i_1, \ldots, i^b)$—that is, $rd^b = \max_{i \in N^b}\{rd_i\}$. The label $L^b = (N^b, q^b, \ell^b, g^b, d^b, i^b, \tilde{c}^b)$ associated with path $b = (0, i_1, \ldots, i^b)$ is feasible if it satisfies the following conditions

$$\begin{cases} N^b \subseteq N \\ q^b \leq \overline{q} \\ a_{i^b} \leq \ell^b \leq b_{i^b} \\ rd^b \leq g^b \leq b_0 \\ i^b \in N^b. \end{cases}$$

When a label $L^b = (N^b, q^b, \ell^b, g^b, d^b, i^b, \tilde{c}^b)$ is extended to the depot to generate label $L^{b'} = (N^{b'}, q^{b'}, \ell^{b'}, g^{b'}, d^{b'}, i^{b'}, \tilde{c}^{b'})$, functions (10) are replaced with the following extension functions:

$$\begin{cases} N^{b'} = N^b \\ q^{b'} = q^b \\ \ell^{b'} = \ell^b - t_{0i^b} \\ g^{b'} = \max\{\max\{rd^{b'}, a_0\} + d^b + t_{0i^b},\ g^b\} \\ d^{b'} = \max\{d^b + t_{0i^b},\ g^b - b_0\} \\ i^{b'} = 0 \\ \tilde{c}^{b'} = \tilde{c}^b + \tilde{c}_{0i^b}. \end{cases}$$

8.4. Structure Enumeration for the DRP

The drone-routing problem studied by Cheng et al. (2018) can be seen as a generalization of the CMTVRPTW, where vehicles are drones and specific features of the drones must be taken into account. In particular, unlike standard vehicles, drones have a limited battery capacity whose consumption depends on the load carried. Therefore, energy consumption and energy cost should be considered when optimizing drone-based distribution systems.

The energy consumed by a drone to traverse arc $(i, j) \in A$ depends both on the distance traveled and the weight transported while traversing the arc. Let en_{ijq} and ce_{ijq} be the energy consumption and the energy costs, respectively, associated with $(i, j) \in A$ when the drone is carrying weight q. Each drone has to return to the depot before running out of battery. Let us call $\overline{en}$ the drone battery capacity. Moreover, following Cheng et al. (2018), let us also assume that each drone starts each trip equipped with a fully charged battery.

Let us associate with each backward path $b = (0, i_1, \ldots, i^b)$ a label $L^b = (N^b, q^b, \ell^b, g^b, d^b, i^b, en^b, \tilde{c}^b)$, where en^b is the energy consumption of path b. A label $L^b = (N^b, q^b, \ell^b, g^b, d^b, i^b, en^b, \tilde{c}^b)$ is feasible if the following conditions hold:

$$\begin{cases} N^b \subseteq N \\ q^b \leq \overline{q} \\ a_{i^b} \leq \ell^b \leq b_{i^b} \\ a_0 \leq g^b \leq b_0 \\ en^b \leq \overline{en} \\ i^b \in N^b. \end{cases}$$

The enumeration is initialized with a single label $L^b = (N^b, q^b, \ell^b, g^b, d^b, i^b, en^b, \tilde{c}^b)$, where $N^b = \emptyset$, $q^b = 0$, $\ell^b = b_0$, $g^b = a_0$, $d^b = 0$, $en^b = 0$, and $\tilde{c}^b = 0$. Each label $L^b = (N^b, q^b, \ell^b, g^b, d^b, i^b, \tilde{c}^b)$ for which either (*a*) $i^b \in N$ or (*b*) $i^b = 0$ and $N^b = \emptyset$ is extended toward any vertex $j \in V \setminus N^b$ to generate label $L^{b'} = (N^{b'}, q^{b'}, \ell^{b'}, g^{b'}, d^{b'}, i^{b'}, en^{b'}, \tilde{c}^{b'})$, where

$$\begin{cases} N^{b'} = N^b \cup \{j\} \text{ (if } j \in N) \text{ or } N^{b'} = N^b \text{ (if } j = 0) \\ q^{b'} = q^b + q_j \\ \ell^{b'} = \min\{b_j, \ell^b - t_{ji^b} - st_j\} \\ g^{b'} = \max\{a_j + d^b + t_{ji^b} + st_j,\ g^b\} \\ d^{b'} = \max\{d^b + t_{ji^b} + st_j,\ g^b - b_j\} \\ en^{b'} = en^b + en_{ji^bq^b} \\ i^{b'} = j \\ \tilde{c}^{b'} = \tilde{c}^b + c_{ji^b} + ce_{ji^bq^b} - u_j^1 \text{ (if } j \in N) \text{ or} \\ \tilde{c}^{b'} = \tilde{c}^b + c_{ji^b} + ce_{ji^bq^b} \text{ (if } j = 0). \end{cases}$$

Finally, dominance rules should be adjusted as follows. Given two labels $L^{b_1} = (N^{b_1}, q^{b_1}, \ell^{b_1}, g^{b_1}, d^{b_1}, i^{b_1}, en^{b_1}, \tilde{c}^{b_1})$ and $L^{b_2} = (N^{b_2}, q^{b_2}, \ell^{b_2}, g^{b_2}, d^{b_2}, i^{b_2}, en^{b_2}, \tilde{c}^{b_2})$, L^{b_2} is dominated if $N^{b_1} = N^{b_2}$, $i^{b_1} = i^{b_2}$, $\ell^{b_1} \geq \ell^{b_2}$, $g^{b_1} \leq g^{b_2}$, $d^{b_1} \leq d^{b_2}$, $en^{b_1} \leq en^{b_2}$, and $\tilde{c}^{b_1} \leq \tilde{c}^{b_2}$.

9. Computational Results

In this section, we report a summary of the computational results achieved by the ESF on the CMTVRPTW and its four variants, and, when available, we compare its performance with the state-of-the-art exact methods. Detailed computational results are reported in the online appendix. The ESF is coded in C and compiled with Microsoft Visual C++ compiler. Cplex 12.8 is used to solve the master problem in Step 2, in Step 4, and for the branch-and-cut in Step 6. For Step 2 and Step 4, we set Cplex to use the primal and the dual simplex method, respectively; for all the other parameters, we use the default setting. For Step 6, we disable the presolve phase and the cutting-plane generation is embedded in the callback of the solver; all the other parameters are set to the default setting of Cplex. The tests are performed, in single-thread mode, on a Windows server equipped with six virtual CPUs running at 2.59 GHz and with 16 GB of RAM. A time limit of 3 hours per instance is set. All computing times are in seconds. The same parameter setting is used in all tests—namely, $col\_iter = 100$, $gap^0_{guess} = 0.05$, and $gap_{step} = 0.05$.

The first five subsections are devoted to the CMTVRPTW, CMTVRPTW-LD, CMTVRPTW-LT, CMTVRPTW-R, and DRP, respectively. In each subsection, we describe the instances first and then provide the results achieved by our solution method. The following subsections are devoted to the comparison with the literature. Finally, the impact of time windows' width on the performance of ESF is investigated in Section 9.7.

Tables 1–6 summarize the results by group of instances, and the following information is indicated: group of instances (Group), number of customers (n), number of instances (Inst), number of instances solved (Solved), average gap between $z(P1_{\mathcal{G}})$ and the optimal solution cost ($P1_{\mathcal{G}}\%$), average time to compute $z(P1_{\mathcal{G}})$ (T_{P1}), average cardinality of the set $\mathcal{S}_1$ ($|\mathcal{S}_1|$), average gap between $z(P2_{\mathcal{S}_1})$ and the optimal solution cost ($P2_{\mathcal{S}_1}\%$), average cardinality of the set $\mathcal{S}_2$ ($|\mathcal{S}_2|$), average gap between the root node relaxation of $z(F_{\mathcal{S}_2})$ and the optimal solution cost ($LF_{\mathcal{G}}\%$), average number of SFC added (nSFC), average number of nodes explored (Nds), average time spent to execute Step 6 ($T_{B\&C}$), and average total computing time (T_{tot}). Notice that, as Steps 3–7 may be iterated because of the conditions in Step 7, all information about Steps 3–7 refer to the last iteration. Table 3 has an additional column ($\bar{d}$) reporting the maximum trip duration, and Table 4 has an additional column (κ) indicating the type of release date (κ). The last row of each table indicates the total number of instances, the total number of instances solved, and the average values of each column computed over the instances solved only.

In the literature, different authors use different ways to compute travel times and costs, even when solving the same set of instances. Indeed, travel times and costs are either truncated or rounded to the first, second, or another decimal digit. The computational results in the literature show that this does not affect the complexity of the problem. Therefore, in the experiments reported in Tables 1–6, travel time and costs are truncated to one decimal digit, as done in Solomon (1987), who introduced the test instances used (or extended) by most papers on vehicle routing.

All test instances used in this section are available in the online appendix.

9.1. Computational Results on the CMTVRPTW

The benchmark set consists of 81 instances, derived from the type 2 Solomon instances by using a procedure similar to Hernandez et al. (2014, 2016). We consider type 2 instances only because the short planning horizon and the tight time windows of instances of type 1 prevent the vehicles from performing multiple trips. There are three groups of instances (C, R, and RC) differing in the customer distribution (i.e., clustered, random, and randomly clustered, respectively). We consider instances with 25, 40, and 50 customers by selecting the first customers from the original Solomon instances. Instances with 25 customers feature two vehicles, whereas instances with

Table 1. Summary of the Computational Results for the CMTVRPTW

Group	n	Instance	Solved	$P1_{\mathcal{G}}\%$	T_{P1}	$\lvert\mathcal{S}_1\rvert$	$P2_{\mathcal{S}_1}\%$	$\lvert\mathcal{S}_2\rvert$	$LF_{\mathcal{G}}\%$	nSFC	Nds	$T_{B\&C}$	T_{tot}
C	25	8	8	2.37	11.1	75,662	0.94	6,075	0.69	0	88	1.0	16.1
C	40	8	6	2.71	22.0	1,054,466	1.73	207,049	1.71	0	22,257	1,589.5	2,589.9
C	50	8	2	3.50	21.0	1,820,115	0.95	75,645	0.99	0	1,190	90.3	1,601.1
R	25	11	11	7.76	9.4	1,338,633	1.24	5,018	0.77	1	35	1.3	83.1
R	40	11	10	2.53	21.1	2,777,514	0.36	31,561	0.42	0	17	2.7	377.7
R	50	11	0										
RC	25	8	8	12.88	4.0	124,067	2.14	4,143	1.75	5,695	4,036	70.9	175.7
RC	40	8	8	11.95	9.4	3,935,229	0.45	97,816	0.66	304	4,241	479.7	1,223.7
RC	50	8	7	5.95	23.3	1,966,892	0.61	14,038	0.61	1,011	9,886	279.8	647.6
All		81	60	6.55	14.1	1,655,251	1.03	45,449	0.90	918	4,543	268.8	654.9

Table 2. Summary of the Computational Results for the CMTVRPTW-LT

Group	n	Instance	Solved	$P1_{\mathcal{G}}$%	T_{P1}	$\lvert\mathcal{S}_1\rvert$	$P2_{\mathcal{S}_1}$%	$\lvert\mathcal{S}_2\rvert$	$LF_{\mathcal{G}}$%	nSFC	Nds	$T_{B\&C}$	T_{tot}
C	25	8	8	3.08	13.3	73,298	1.35	5,575	0.73	1	71	1.0	19.0
C	40	8	7	2.71	29.4	1,323,350	1.55	190,787	1.51	0	10,877	933.9	2,170.0
C	50	8	3	3.99	35.0	2,429,725	1.35	78,585	1.41	3	5,980	494.7	3,576.6
R	25	11	11	8.00	11.2	1,483,675	0.91	2,992	0.78	2	27	0.8	114.8
R	40	11	10	2.74	22.5	2,821,178	0.31	27,430	0.41	0	7	2.6	417.9
R	50	11	0										
RC	25	8	8	13.39	4.7	117,308	2.30	4,658	1.91	7,614	10,379	733.9	879.7
RC	40	8	8	12.04	9.5	3,677,170	0.46	73,755	0.83	299	2,745	186.3	871.9
RC	50	8	7	6.37	17.6	1,965,379	0.60	7,269	0.59	843	2,936	26.5	311.7
All		81	62	6.76	16.2	1,706,204	1.05	41,956	0.96	1,117	3,557	251.8	769.5

40 and 50 customers feature four vehicles. The vehicle capacity is set equal to 100.

Table 1 summarizes computational results on the CMTVRPTW. It shows that the ESF can solve all 25-customer instances and all but three 40-customer instances. Instances with 50 customers represent the limit of the ESF, as only 9 out of 27 instances are solved. Overall, 60 of the 81 instances are solved in an average computing time of 11 minutes.

9.2. Computational Results on the CMTVRPTW-LT

Hernandez et al. (2016) introduce 27 instances of the CMTVRPTW-LT, generated by considering the first 25 customers of the Solomon instances of type 2. The fleet size is equal to two, and vehicle capacity equals 100. The loading time of each customer $i \in N$ is computed as $lt_i = 0.2st_i$. Following the procedure of Hernandez et al. (2016), we introduce 54 additional instances with 40 and 50 customers. We set capacity and loading times as in Hernandez et al. (2016), but we set the fleet size equal to four.

Table 2 summarizes the results on the 81 CMTVRPTW-LT instances. The ESF can solve all original 25-customer instances and all but two 40-customer instances. Ten of the 27 50-customer instances can be solved. All in all, the computational behaviour of the ESF is similar to that observed on the CMTVRPTW in Table 1.

9.3. Computational Results on the CMTVRPTW-LD

For the CMTVRPTW-LD, we use a set of instances based on the Solomon instances, with 25, 40, and 50 customers of groups C2, R2, and RC2, considering a short limited trip duration ($\overline{d} = \overline{d}_{\text{short}}$) and a long limited trip duration ($\overline{d} = \overline{d}_{\text{long}}$), for a total of 162 instances. As done in Azi et al. (2010), we set $\overline{d}_{\text{short}} = 75$ and $\overline{d}_{\text{long}} = 100$ for instances of type R2 and RC2, and we set $\overline{d}_{\text{short}} = 220$ and $\overline{d}_{\text{long}} = 250$ for instances of type C2. Travel time, distances, fleet size, and loading times are set as in the CMTVRPTW-LT instances of Table 2.

Table 3 shows that the ESF can solve all 162 instances of the data set with an average computing time of 16.8 seconds.

Table 3. Summary of the Computational Results for the CMTVRPTW-LD

Group	n	$\overline{d}$	Instance	Solved	$P1_{\mathcal{G}}$%	T_{P1}	$\lvert\mathcal{S}_1\rvert$	$P2_{\mathcal{S}_1}$%	$\lvert\mathcal{S}_2\rvert$	$LF_{\mathcal{G}}$%	nSFC	Nds	$T_{B\&C}$	T_{tot}
C	25	220	8	8	5.43	11.2	454	1.47	179	0.11	0	0	0.1	11.4
C	25	250	8	8	2.06	16.3	1,842	0.42	528	0.65	1	45	0.2	16.6
C	40	220	8	8	4.16	25.6	1,161	0.15	397	0.02	0	1	2.1	27.9
C	40	250	8	8	1.24	41.0	5,325	0.50	3,042	0.36	0	1,422	2.8	44.1
C	50	220	8	8	4.38	40.0	1,859	0.51	1,120	0.07	0	0	9.8	50.0
C	50	250	8	8	1.27	65.7	8,977	0.50	5,632	0.37	0	1,224	6.9	73.1
R	25	75	11	11	3.15	1.7	609	0.10	91	0.32	0	0	0.0	1.8
R	25	100	11	11	6.11	2.7	2,886	0.23	273	0.18	0	0	0.0	2.9
R	40	75	11	11	4.70	4.0	3,412	0.50	794	0.16	0	1	0.2	4.3
R	40	100	11	11	5.94	8.1	26,682	0.36	2,972	0.35	0	36	0.1	8.8
R	50	75	11	11	5.06	6.0	5,393	0.44	1,900	0.20	4	42	1.8	8.1
R	50	100	11	11	6.22	12.7	52,811	0.24	7,940	0.24	335	1,507	16.0	31.8
RC	25	75	8	8	15.70	1.4	507	4.52	222	0.22	16	31	0.1	1.8
RC	25	100	8	8	18.93	2.5	2,733	0.69	553	0.75	15	16	0.1	3.0
RC	40	75	8	8	11.86	2.6	631	3.28	356	0.42	0	1	0.7	3.6
RC	40	100	8	8	18.02	6.5	3,614	0.65	1,263	0.18	1	8	0.1	7.1
RC	50	75	8	8	11.04	4.2	1,295	2.64	774	0.47	163	406	5.7	10.1
RC	50	100	8	8	18.20	10.4	10,795	1.74	5,074	0.08	0	1	0.4	12.3
All			162	162	7.66	13.6	8,168	0.97	1,894	0.28	33	263	2.67	16.8

Table 4. Summary of the Computational Results for CMTVRPTW-R

Group	n	κ	Instance	Solved	$P1_{\mathcal{S}}$%	T_{P1}	$\lvert\mathcal{S}_1\rvert$	$P2_{\mathcal{S}_1}$%	$\lvert\mathcal{S}_2\rvert$	$LF_{\mathcal{S}}$%	nSFC	Nds	$T_{B\&C}$	T_{tot}
C	25	0.25	8	8	9.24	8.6	740,684	2.69	6,722	0.64	16	216	3.6	30.4
C	25	0.5	8	8	9.25	9.0	752,587	2.70	6,515	0.64	25	463	7.3	35.1
C	25	0.75	8	8	9.35	8.9	677,542	2.74	6,666	0.67	20	629	9.6	39.3
C	40	0.25	8	6	4.33	20.3	586,585	1.00	23,165	0.89	0	921	13.9	75.3
C	40	0.5	8	6	5.39	20.3	541,465	1.12	21,346	0.96	0	1,417	13.3	72.3
C	40	0.75	8	6	7.05	18.4	595,428	0.92	27,052	0.64	0	593	11.3	73.9
C	50	0.25	8	5	4.49	34.8	459,779	0.72	3,024	0.56	0	975	1.8	59.7
C	50	0.5	8	5	4.86	35.9	534,103	0.61	13,925	0.44	0	361	1.8	104.3
C	50	0.75	8	5	7.13	35.1	729,090	0.26	2,129	0.22	0	4	0.2	86.2
R	25	0.25	11	9	10.77	7.0	1,379,650	0.15	96	0.19	0	0	0.0	18.4
R	25	0.5	11	9	11.42	6.7	1,602,670	0.13	518	0.33	0	0	0.0	20.8
R	25	0.75	11	8	13.82	5.6	1,042,126	0.14	443	0.06	0	0	0.0	25.5
R	40	0.25	11	4	7.64	12.8	5,238,404	0.02	2,761	0.14	0	0	0.1	105.1
R	40	0.5	11	4	7.87	13.4	3,924,628	0.04	1,049	0.21	0	0	0.0	78.1
R	40	0.75	11	2	11.37	10.3	2,280,014	0.00	1,400	0.04	0	0	0.0	48.0
R	50	0.25	11	0										
R	50	0.5	11	0										
R	50	0.75	11	0										
RC	25	0.25	8	8	16.12	3.3	120,983	1.28	3,915	0.98	496	192	1.0	12.1
RC	25	0.5	8	8	16.19	3.1	118,613	1.18	4,287	0.91	443	258	0.9	10.8
RC	25	0.75	8	8	19.92	2.7	112,202	0.41	1,148	0.25	4	4	0.1	5.7
RC	40	0.25	8	7	16.34	8.2	1,750,114	0.44	19,208	0.65	13	9,115	108.1	217.5
RC	40	0.5	8	7	16.41	7.3	1,547,847	0.21	13,625	0.43	336	2,118	42.4	141.6
RC	40	0.75	8	7	21.24	6.9	1,096,531	0.46	2,898	0.47	0	935	1.4	59.7
RC	50	0.25	8	6	10.65	18.3	1,720,962	0.95	9,708	0.64	6,983	16,660	1,038.5	1,221.4
RC	50	0.5	8	5	13.25	17.0	1,768,256	0.43	9,591	0.20	1	5	0.6	173.4
RC	50	0.75	8	5	16.53	16.0	1,875,569	0.59	15,468	0.32	34	155	6.1	271.4
All			210	154	11.21	13.4	1,409,377	0.74	7,581	0.45	319	1334	48.1	118.0

9.4. Computational Results on the CMTVRPTW-R

The only CMTVRPTW-R instances available in the literature are provided by Cattaruzza et al. (2016a), who generated a test bed based on the Solomon instances. In particular, for each instance, they consider four types of release date (for details, see Cattaruzza et al. 2016a) and set the capacity equal to half of the original value. Nonetheless, all instances of this test bed feature 100 customers, which is out of reach for our method. Therefore, we generate a set of 210 instances starting from the instances of Cattaruzza et al. (2016a) by considering the first 25, 40, and 50 customers only and by considering three types of tighter and tighter release dates (0.25, 0.5, and 0.75). As for

Table 5. Summary of the Computational Results for the DRP (Type A Instances)

Group	n	Instance	Solved	$P1_{\mathcal{S}}$%	T_{P1}	$\lvert\mathcal{S}_1\rvert$	$P2_{\mathcal{S}_1}$%	$\lvert\mathcal{S}_2\rvert$	$LF_{\mathcal{S}}$%	nSFC	Nds	$T_{B\&C}$	T_{tot}
A1	10	5	5	11.07	0.2	105	3.43	52	0.59	6	4	0.1	0.5
A1	15	5	5	13.06	0.5	887	2.17	175	1.26	3	110	0.1	0.9
A1	20	5	5	8.69	0.5	930	1.05	229	0.60	58	163	1.2	1.9
A1	25	5	5	9.32	1.2	3,355	1.53	413	1.02	92	333	1.7	3.8
A1	30	5	5	8.25	1.3	9,884	0.83	1,091	0.77	140	323	1.5	4.4
A1	35	5	5	8.53	2.5	19,224	0.59	6,725	0.57	7	1,231	15.4	18.7
A1	40	5	5	8.07	2.8	16,630	0.35	1,525	0.26	71	174	4.3	7.5
A1	45	5	5	8.29	5.6	58,018	0.52	4,139	0.43	90	421	10.1	18.9
A2	10	5	5	11.15	0.1	54	1.92	22	0.21	3	0	0.0	0.4
A2	15	5	5	13.22	0.2	120	2.02	50	1.30	42	25	0.5	1.6
A2	20	5	5	11.28	0.4	432	1.72	125	1.12	73	120	1.2	2.0
A2	25	5	5	9.74	0.7	498	0.91	110	1.06	63	123	1.6	2.5
A2	30	5	5	10.48	1.6	1,798	0.73	578	0.26	25	45	1.0	2.9
A2	35	5	5	10.18	2.3	2,759	0.65	1,089	0.32	5	5	1.4	4.1
A2	40	5	5	9.17	3.3	3,648	0.51	1,279	0.25	89	163	5.8	9.4
A2	45	5	5	9.23	3.3	6,654	0.45	787	0.29	40	85	4.2	7.8
A2	50	5	5	8.82	5.8	6,706	0.53	1,905	0.34	239	573	48.8	54.9
All		85	85	9.91	1.9	7,747	1.17	1,194	0.63	62	229	5.8	8.4

Table 6. Summary of the Computational Results for DRP (Type B Instances)

Group	n	Instance	Solved	$P1_{\mathcal{G}}$%	T_{P1}	$\lvert\mathcal{S}_1\rvert$	$P2_{\mathcal{G}_1}$%	$\lvert\mathcal{S}_2\rvert$	$LF_{\mathcal{G}}$%	nSFC	Nds	$T_{B\&C}$	T_{tot}
C	25	8	8	8.97	0.5	10,377	1.98	577	1.32	0	568	0.4	1.2
C	40	8	8	7.54	1.4	63,146	0.77	6,989	0.44	3	203	1.7	4.9
R	25	11	11	7.11	0.3	5,305	0.02	200	0.13	0	0	0.0	0.5
R	40	11	11	7.04	0.8	43,020	0.03	2,889	0.03	0	0	0.5	1.9
RC	25	8	8	12.46	0.1	1,232	2.20	356	0.07	7	17	0.1	0.5
RC	40	8	8	10.51	0.5	2,105	1.39	1,503	0.14	200	538	3.4	4.2
All		54	54	8.73	0.6	21,231	0.95	2,026	0.32	31	196	0.9	2.1

the CMTVRPTW, instances with 25 customers feature two vehicles, and instances with 40 and 50 customers four vehicles.

Table 4 summarizes the results by instance group, number of customers, and type of release date. The ESF can solve 154 of the 210 instances. The complexity of the problem clearly increases with the number of customers. Instances of type R are more challenging than instances of types C and RC. The bottleneck of the ESF is clearly Step 3—that is, the structure enumeration. Indeed, instances cannot be solved when it is impossible to enumerate all structures, whereas instances can be solved, on average, in less than 2 minutes when all structures can be enumerated.

9.5. Computational Results on the DRP

We tested the ESF on the two sets (i.e., A and B) of DRP instances proposed by Cheng et al. (2018). Set A is created according to the framework presented in Solomon (1987) and Dorling et al. (2017); set B extends the Solomon instances to take into account the features of the drones. Type A instances consists of 85 instances divided in two groups (A1 and A2), which differ in the depot location; each group contains five instances with 10, 15, etc., up to 50 customers. Type B instances are generated by considering the first 25 and 40 customers of the type 2 (C2, R2, and RC2) Solomon instances, for a total of 54 instances. For further details about the instances, we refer the reader to Cheng et al. (2018).

Table 5 summarizes the results on DRP instances of type A. Results are grouped by instance group and number of customers. Table 5 shows that all 85 instances can be solved with an average computing time of 8.4 seconds.

Table 6 summarizes the results on the DRP instances of Type B. Results are grouped by customer distribution (C, R, and RC) and number of customers (25 and 40). Table 6 shows that all 54 can be solved, with an average computing time of 2.1 seconds.

9.6. Comparison with the Literature

In this section, we compare the performance of the ESF with the state-of-the-art exact methods from the literature. In particular, we compare the ESF with the methods of Hernandez et al. (2016) (hereafter Hern16) on the CMTVRPTW-LT, of Hernandez et al. (2014) (hereafter Hern14) on the CMTVRPTW-LD, and of Cheng et al. (2018) (hereafter Cheng18) on the DRP. Notice that these three methods have been designed and tailored on the problem at hand and have not been generalized to several CMTVRPTWs as our proposed ESF.

9.6.1. Comparison with Hernandez et al. (2016) on the CMTVRPTW-LT. Table 7 compares the performance of EFS and Hern16 on the 27 25-customer CMTVRPTW-LT instances proposed by Hernandez et al. (2016). In particular, as Hernandez et al. (2016) propose two exact branch-and-price algorithms, the comparison is done with the better of the two, which relies on a trip-based formulation. The branch-and-price of Hernandez et al. (2016) is run on an Intel Core i7 2670QM with 8 GB of RAM. A dash indicates that the instance was not solved.

Instances are grouped by customer distribution (C2, R2, and RC2). For each instance, we report the instance number (Inst); the best-known upper bound (UB); for ESF, the gap (Gap) between $LF_{\mathcal{G}}$ and the optimal solution cost, and the total computing time (Time); and for Hern16, the gap (Gap) between the linear relaxation of the corresponding trip-based formulation and the optimal solution cost, and the total computing time (Time).

Table 7 shows that the ESF can solve all 27 instances, whereas Hern16 cannot solve instances RC204 and RC208. Moreover, the ESF is on average several times faster than Hern16, mainly because of the smaller gaps provided by the lower bounding procedure.

9.6.2. Comparison with Hernandez et al. (2014) on the CMTVRPTW-LD. Tables 8 and 9 compare the results achieved by the ESF and by the branch-and-price of Hernandez et al. (2014) on the instances having a feasible solution used by Hernandez et al. (2014) and introduced in Azi et al. (2010). The branch-and-price of Hernandez et al. (2014) was run on an Intel Core 2 Duo 2.10 GHz with 2 GB of RAM. Table 8 reports the results on 27 instances with 25 customers, each one solved with a short and a long limited trip duration.

Table 7. Comparison with Hernandez et al. (2016) on the CMTVRPTW-LT

	C2					R2					RC2				
		ESF		Hern16			ESF		Hern16			ESF		Hern16	
Instance	Upper bound	Gap	Time	Gap	Time	Instance	Upper bound	Gap	Time	Gap	Time	Instance	Upper bound	Gap	Time	Gap	Time
1	380.8	0.45	8.7	3.46	7.2	1	554.6	0.71	154.0	3.19	13.8	1	660.0	0.19	11.0	3.52	5.9
2	368.6	0.11	19.2	1.71	245.7	2	485.0	0.81	482.1	2.49	53.2	2	596.8	9.91	6,891.4	8.98	19,737.7
3	361.7	0.48	23.7	0.55	37.6	3	444.2	0.10	108.9	2.00	198.2	3	530.1	1.82	66.0	7.16	11,904.5
4	358.8	0.11	45.0	0.76	180.7	4	407.5	2.02	158.9	4.70	2,242.8	4	518.0	1.34	12.0	—	—
5	377.2	2.84	11.2	4.60	65.4	5	448.4	0.00	37.1	1.19	13.3	5	605.3	0.02	22.7	3.84	86.3
6	367.2	1.53	13.1	2.87	56.7	6	413.9	0.00	43.3	0.44	69.7	6	575.1	0.78	13.7	5.41	526.4
7	359.1	0.04	17.6	1.76	154.4	7	400.1	2.62	18.4	2.53	842.4	7	528.2	1.13	12.3	3.54	7,764.0
8	360.9	0.27	13.7	2.20	112.6	8	394.3	0.09	21.5	3.47	1,049.0	8	506.4	0.07	8.2	—	—
						9	418.3	1.26	48.6	1.54	62.0						
						10	448.3	0.07	124.2	1.83	79.2						
						11	400.1	0.94	65.6	3.14	2,481.7						
Average		0.73	19.0	2.24	107.5			0.78	114.8	2.41	645.9			1.91	1,169.5	5.41	6,670.8
	8 instances		Solved: 8		Solved: 8		11 instances		Solved: 11		Solved: 11		8 inst.		Solved: 8		Solved: 6

Table 9 reports the results on 18 instances with 40 customers, each one solved with a short and a long limited trip duration. Columns of these two tables have the same meaning of the columns of Table 7.

Table 8 shows that the ESF can solve all 25-customer instances, whereas Hern14 cannot solve five of them. Moreover, the ESF is, on average, much faster than Hern14 on both instances with short and long limited duration, in particular on the latter ones.

Table 9 indicates that the ESF can solve all 35 instances, which is 10 more than Hern14. The ESF is more efficient in particular on instances with a long limited duration. Also, in terms of computing time, the ESF is clearly superior.

9.6.3. Comparison with Cheng et al. (2018) on the DRP.

Table 10 compares the results achieved by the ESF and by the branch-and-cut of Cheng et al. (2018) based on the mathematical formulation without a drone index, referred to as $(R+E)_e|nk$ in Cheng et al. (2018). The branch-and-cut of Cheng et al. (2018) is run on a cluster of Intel Xeon X5650 CPUs with 2.67 GHz and 24 GB of RAM under Linux 6.3.

For each instance type (Type), we report the instance group (Group), the number of customers (n), and the number of instances (Inst). Columns under heading ESF report the number of instances solved (Solved), the average gap (Gap) between $LF_{\mathcal{G}}$% and the optimal solution cost, and the average computing time (Time) of the ESF. Columns under heading Cheng18 report the number of instances solved (Solved), the gap (Gap) between the root node relaxation of $(R+E)_e|nk$ and the optimal solution cost, and the average computing time (Time) of Cheng18. The average values reported in columns Gap and Time are computed over the instances solved only.

Table 10 shows that the ESF can solve all 139 instances, whereas Cheng18 can solve 45 instances only. Moreover, the ESF is on average much faster than Cheng18, thanks to the smaller gaps provided by the lower bounding procedure.

9.7. Impact of the Width of the Time Windows

In this section, we investigate the impact of the time window's width on the performance of ESF. We test ESF on the CMTVRPTW instances studied in Section 9.1 by enlarging the original time windows. For each of the 81 instances, we consider three additional instances: the first featuring time windows that are 50% larger, the second featuring time windows twice as large as the initial one, and the third without time windows. In particular, time windows $[a'_i, b'_i]$, $i \in N$, of these new instances are defined as

$$a'_i = \lceil \max\{a_0 + t_{0i}, a_i - (b_i - a_i)\delta\} \rceil \qquad i \in N$$
$$b'_i = \lfloor \min\{b_0 - t_{i0} - st_i, b_i + (b_i - a_i)\delta\} \rfloor \qquad i \in N,$$

Table 8. Comparison with Hernandez et al. (2014) on the CMTVRPTW-LD (Instances with 25 Customers)

	Short limited duration					Long limited duration				
		ESF		Hern14			ESF		Hern14	
Inst	Upper bound	Gap	Time	Gap	Time	Upper bound	Gap	Time	Gap	Time
C201	659.02	0.00	5.3	1.90	1.3	540.90	0.00	3.6	0.00	0.1
C202	653.37	0.00	14.2	2.85	49.3	533.43	0.00	18.6	1.54	51.4
C203	646.40	0.22	20.1	3.15	265	532.77	0.21	25.6	1.70	335.7
C204	602.46	0.01	28.7	1.73	248	525.46	0.86	35.6	2.29	4734.4
C205	636.39	0.00	7.5	4.48	38.1	529.94	1.49	9.4	1.49	0.9
C206	636.39	0.00	8.3	5.19	692.4	527.84	0.04	12.7	2.21	123.9
C207	603.22	0.00	10.2	2.39	104.7	525.46	1.86	18.4	1.86	31.1
C208	613.20	0.00	9.6	2.59	41.4	525.46	1.86	14.6	1.86	4.7
R201	762.43	0.61	1.2	0.61	0.1	698.18	0.00	1.3	0.99	0.8
R202	645.78	0.00	1.6	0.00	0.6	617.53	0.00	2.9	0.14	4.1
R203	621.97	0.29	2.4	0.16	2.0	577.74	0.00	4.6	0.11	11.6
R204	579.68	0.32	2.9	0.70	4.9	483.3	0.00	4.0	0.54	33.6
R205	634.09	1.12	1.2	1.20	1.0	559.14	0.03	2.2	0.64	3.7
R206	596.74	0.00	1.9	0.00	0.8	523.64	0.00	3.3	0.00	5.7
R207	585.74	0.00	2.7	0.34	3.5	512	0.73	4.1	2.85	418.9
R208	579.68	0.32	3.0	0.70	7.2	483.3	0.49	4.2	1.30	97.8
R209	602.39	0.62	1.7	0.71	1.7	517.69	0.00	2.9	1.11	14.1
R210	636.15	0.00	1.6	2.49	8.5	547.23	0.00	2.8	0.00	2.6
R211	575.91	0.23	2.0	1.28	27.6	474.49	0.00	4.1	0.93	80.4
RC201	988.05	0.15	1.2	0.37	0.9	849.33	2.79	14.3	4.15	3.6
RC202	881.49	0.00	2.2	4.98	24.5	679.86	0.00	3.0	0.00	3.6
RC203	749.15	0.00	2.3	5.87	62.3	593.56	0.07	4.5	0.10	13.1
RC204	744.72	0.00	3.2	—	—	587.22	0.01	5.0	—	—
RC205	840.35	0.45	2.0	3.78	3.7	702.49	0.54	2.5	0.54	2.6
RC206	761.03	0.08	1.5	4.59	35.7	604.12	0.00	1.9	0.65	2.9
RC207	738.87	0.35	2.1	—	—	514.81	0.00	3.2	0.84	45.5
RC208	727.99	5.85	2.8	—	—	502.18	0.00	4.6	—	—
Average		0.39	5.3	2.17	67.7		0.41	7.9	1.11	241.1
	27 instances		Solved: 27		Solved: 24	27 instances		Solved: 27		Solved: 25

and parameter δ is set equal to 0.25, 0.5, and ∞ to obtain the three instances. Notice that the time window of the depot remains unchanged.

Table 11 compares the results of ESF on the original instances (i.e., with $\delta = 0$) with the results on the instances obtained by setting $\delta = 0.25$, 0.50, and ∞. We report the instances group (Group), the number of customers (n), and the number of instances (Inst). For each value of δ, we indicate the number of instances solved (Solved), the average gaps between the lower bounds and the optimal solution costs ($P1_{\mathcal{G}}$%, $P2_{\mathcal{G}_1}$%, $LF_{\mathcal{G}}$%), and the average total computing time (T_{tot}).

Table 11 shows that the performance of ESF does not deteriorate when time windows are enlarged. Indeed, ESF can solve 73 instances with $\delta = 0.25$, 71 with $\delta = 0.5$, and 70 instances with $\delta = \infty$, whereas 60 instances were solved with $\delta = 0$. Therefore, we cannot conclude that the width of the time windows has a consistent impact on the performance of ESF.

Nevertheless, we can observe a general trend in the quality of the three lower bounds: By increasing the width of the time windows, the three lower bounds are often on average better. This is probably why instances with larger time windows are not always more difficult to solve in spite of the larger number of feasible structures involved and the higher complexity of the pricing problems and enumeration phase.

10. Conclusions

We have presented an *exact solution framework* for multitrip vehicle-routing problems with time windows. The ESF relies on a novel mathematical model with an exponential number of variables and constraints. Unlike the other mathematical models from the literature, the proposed model is based on the concept of *structure*. The novel formulation is used to derive two lower bounds that can be efficiently computed. These lower bounds are exploited in the ESF to generate a reduced set of columns containing any optimal MTVRP solution, which is then found by using a branch-and-cut. The computational results show that the ESF can efficiently solve instances, with up to 50 customers of the CMTVRPTW significantly outperforming the state-of-the-art exact methods. We have also shown that the ESF can easily be adapted to solve four generalizations of the CMTVRPTW, and

Table 9. Comparison with Hernandez et al. (2014) on the CMTVRPTW-LD (Instances with 40 Customers)

	Short limited duration					Long limited duration				
		ESF		Hern14			ESF		Hern14	
Instance	Upper bound	Gap	Time	Gap	Time	Upper bound	Gap	Time	Gap	Time
C201	1,168.83	1.95	1,068.5	3.79	31.3	966.70	2.38	194.9	3.32	90.4
C202	1,111.15	0.10	218.1	1.19	67.4	919.85	0.00	50.3	0.63	84.2
C203	1,088.55	0.00	43.0	1.04	186.5	915.04	1.27	81.1	—	—
C204	1,039.16	0.00	59.9	0.45	145.3	908.24	1.13	115.9	—	—
C205	1,083.81	0.00	14.0	0.64	34.1	921.19	0.00	19.6	1.02	66.3
C206	1,081.37	0.00	16.2	0.67	184	919.05	0.25	131.3	0.87	1,539.1
C207	1,055.04	0.12	307.3	1.16	1,491.5	910.43	0.54	46.3	—	—
C208	1,071.99	0.00	22.3	0.75	52.6	915.41	0.00	31.0	0.73	2,673.7
R202				Infeasible	961.33	0.38	17.0	—	—	
R203	962.22	0.49	7.0	—	—	816.51	0.34	67.9	1.07	15.5
R204	858.22	0.00	6.7	2.14	4,049.2	708.98	0.76	13.9	—	—
R205	1,017.84	2.98	684.3	2.48	1,193.4	873.22	3.68	2313	2.25	2,429.0
R206	927.22	1.05	12.3	0.28	171.5	812.31	0.46	14.5	1.11	926.4
R207	886.22	0.00	5.6	0.39	68.9	764.39	0.27	11.5	—	—
R208	858.22	0.66	6.7	2.14	4,954.8	708.01	0.95	14.5	—	—
R209	935.81	0.02	4.0	1.00	198.2	768.84	0.44	15.3	1.12	1,511.4
R210	952.92	0.83	272.3	0.98	246.5	822.78	1.50	115.5	—	—
R211	869.75	2.32	54.5	2.18	5,093.9	728.94	3.06	549.0	—	—
RC204	1,362.34	0.00	8.2	—	—	985.98	1.34	13.0	—	—
Average		0.58	156.2	1.33	1,135.6	0.99	200.8	1.35	1,037.3	
	17 instances		Solved: 17	Solved: 16	18 instances			Solved: 18	Solved: 9	

the corresponding computational results are significantly better than those achieved by the best exact methods from the literature.

As the bottleneck of the ESF is represented by the enumeration of all structures in the gap given the CVRPTW lower bound, future research can be directed

Table 10. Comparison with Cheng et al. (2018) on the DRP

				ESF			Cheng18		
Type	Group	n	Inst	Solved	Gap	Time	Solved	Gap	Time
A	1	10	5	5	0.59	0.5	5	4.07	0.6
A	1	15	5	5	1.26	0.9	1	7.92	22.5
A	1	20	5	5	0.60	1.9	2	1.28	13.6
A	1	25	5	5	1.02	3.8	0	—	—
A	1	30	5	5	0.77	4.4	0	—	—
A	1	35	5	5	0.57	18.7	0	—	—
A	1	40	5	5	0.26	7.5	0	—	—
A	1	45	5	5	0.43	18.9	0	—	—
A	2	10	5	5	0.21	0.4	5	4.92	0.3
A	2	15	5	5	1.30	1.6	3	4.65	4.2
A	2	20	5	5	1.12	2.0	3	5.29	21.5
A	2	25	5	5	1.06	2.5	0	—	—
A	2	30	5	5	0.26	2.9	0	—	—
A	2	35	5	5	0.32	4.1	0	—	—
A	2	40	5	5	0.25	9.4	0	—	—
A	2	45	5	5	0.29	7.8	0	—	—
A	2	50	5	5	0.34	54.9	0	—	—
B	C	25	8	8	1.32	1.2	8	1.85	35.5
B	C	40	8	8	0.44	4.9	1	1.64	43.8
B	R	25	11	11	0.13	0.5	11	2.82	66.6
B	R	40	11	11	0.03	1.9	3	4.56	5,543.3
B	RC	25	8	8	0.07	0.5	3	4.42	95.2
B	RC	40	8	8	0.14	4.2	0	—	—
Average					0.51	5.9		3.55	402.4
Solved			139	139			45		

Table 11. Summary of the Computational Results with Time Windows of Increasing Width

			$\delta = 0$					$\delta = 0.25$					$\delta = 0.50$					$\delta = \infty$				
Group	n	Instance	Solved	P1$_{LF}$%	P2$_{LF_1}$%	LF$_{LF}$%	T_{tot}	Solved	P1$_{LF}$%	P2$_{LF_1}$%	LF$_{LF}$%	T_{tot}	Solved	P1$_{LF}$%	P2$_{LF_1}$%	LF$_{LF}$%	T_{tot}	Solved	P1$_{LF}$%	P2$_{LF_1}$%	LF$_{LF}$%	T_{tot}
C	25	8	8	2.37	0.94	0.69	16.1	8	2.36	1.16	0.38	13.8	8	1.93	1.00	0.28	14.1	8	1.81	1.15	0.89	13.9
C	40	8	6	2.71	1.73	1.71	2,589.9	8	2.59	1.65	1.28	428.7	8	2.49	1.65	1.41	289.0	8	2.39	1.48	1.44	174.0
C	50	8	2	3.50	0.95	0.99	1,601.1	7	2.76	0.97	0.93	1,362.5	8	2.70	1.02	0.97	1,242.8	8	2.67	0.88	0.84	1,507.6
R	25	11	11	7.76	1.24	0.77	83.1	11	5.38	1.27	0.35	28.5	11	5.49	1.53	0.90	78.0	11	5.16	1.88	0.00	16.8
R	40	11	10	2.53	0.36	0.42	377.7	11	2.07	0.09	0.06	69.9	11	2.28	0.23	0.16	83.0	11	1.65	0.00	0.00	39.4
R	50	11	0					4	3.11	0.94	0.92	480.0	2	2.86	0.86	0.86	279.3	0				
RC	25	8	8	12.88	2.14	1.75	175.7	8	9.79	1.11	0.93	57.0	8	8.37	0.63	0.52	11.5	8	6.87	0.00	0.00	4.1
RC	40	8	8	11.95	0.45	0.66	1,223.7	8	10.79	0.13	0.06	164.2	8	10.24	0.07	0.00	165.9	8	9.98	0.00	0.00	44.4
RC	50	8	7	5.95	0.61	0.61	647.6	8	4.76	0.43	0.24	256.3	7	3.07	0.09	0.05	29.9	8	2.64	0.00	0.00	24.7
Average				6.55	1.03	0.90	654.9		4.88	0.84	0.52	272.6		4.48	0.80	0.55	229.9		4.08	0.70	0.36	211.0
Solved		81	60					73					71					70				

toward studying valid inequalities to tighten such a lower bound without affecting the complexity of the pricing problem.

Moreover, given the performance of the ESF on a well-studied class of vehicle-routing problems, we consider it promising to explore the usage of similar mathematical models with an exponential number of variables and constraints for other combinatorial optimization problems.

Acknowledgments

The authors thank the associate editor and the three referees for their valuable comments and Diego Cattaruzza for kindly providing the upper bounds for the new CMTVRPTW-R instances.

References

Archetti C, Speranza M, Vigo D (2014) Vehicle routing problems with profits. Toth P, Vigo D, eds. *Vehicle Routing: Problems, Methods, and Applications*, 2nd ed., MOS-SIAM Series on Optimization, vol. 18 (Society for Industrial and Applied Mathematics, Philadelphia), 273–298.

Azi N, Gendreau M, Potvin JY (2007) An exact algorithm for a single-vehicle routing problem with time windows and multiple routes. *Eur. J. Oper. Res.* 178(3):755–766.

Azi N, Gendreau M, Potvin JY (2010) An exact algorithm for a vehicle routing problem with time windows and multiple use of vehicles. *Eur. J. Oper. Res.* 202(3):756–763.

Baldacci R, Mingozzi A, Roberti R (2011) New route relaxation and pricing strategies for the vehicle routing problem. *Oper. Res.* 59(5):1269–1283.

Baldacci R, Mingozzi A, Roberti R (2012) New state-space relaxations for solving the traveling salesman problem with time windows. *INFORMS J. Comput.* 24(3):356–371.

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

Cattaruzza D, Absi N, Feillet D (2016b) Vehicle routing problems with multiple trips. *4OR* 14(3):223–259.

Cheng C, Adulyasak Y, Rousseau LM (2018) Formulations and exact algorithms for drone routing problem. Technical Report CIRRELT-2018-31, CIRRELT, Polytechnique Montréal, Montréal.

Dorling K, Heinrichs J, Messier GG, Magierowski S (2017) Vehicle routing problems for drone delivery. *IEEE Trans. Systems Man Cybernetics Systems* 47(1):70–85.

Fleischmann B (1990) The vehicle routing problem with multiple use of vehicles. Technical report, Fachbereich Wirtschaftswissenschaften, Universität Hamburg, Hamburg, Germany.

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

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

Jepsen M, Petersen B, Spoorendonk S, Pisinger D (2008) Subset-row inequalities applied to the vehicle-routing problem with time windows. *Oper. Res.* 56(2):497–511.

Macedo R, Alves C, De Carvalho JMV, Clautiaux F, Hanafi S (2011) Solving the vehicle routing problem with time windows and multiple routes exactly using a pseudo-polynomial model. *Eur. J. Oper. Res.* 214(3):536–545.

Solomon MM (1987) Algorithms for the vehicle routing and scheduling problems with time window constraints. *Oper. Res.* 35(2): 254–265.

Tilk C, Irnich S (2017) Dynamic programming for the minimum tour duration problem. *Transportation Sci.* 51(2):549–565.

Vansteenwegen P, Souffriau W, Vanden Berghe G, Van Oudheusden D (2009) Iterated local search for the team orienteering problem with time windows. *Comput. Oper. Res.* 36(12): 3281–3290.

Rosario Paradiso is a PhD student in the Department of Mathematics and Computer Science of the University of Calabria in Italy. His studies and research interests are focused on optimization methods for supply chain and logistics decision-making problems.

Roberto Roberti is an associate professor at the Department of Supply Chain Analytics of the Vrije Universiteit Amsterdam in Netherlands. His main research interests include exact solution methods for large-scale optimization, in particular in the field of logistics and supply chain management.

Demetrio Laganá is an assistant professor of operations research at the Department of Mechanical, Energy and Management Engineering of University of Calabria, Italy. His research interests include supply chain optimization, arc-routing and inventory-routing problems, exact and heuristic algorithms, and approximate dynamic programming algorithms. Recently his research has focused on mathematical programming methods to solve rich vehicle-routing problems and integrated logistics problems.

Wout Dullaert is a full professor of supply chain logistics at the Department of Supply Chain Analytics of the Vrije Universiteit Amsterdam, Netherlands. His research focuses on supply chain analysis and management, in particular for tactical and operational distribution problems.