The Vehicle Routing Problem with Drones and Time Windows: Minimizing Route Duration

Jeanette Schmidt[a,*]

[a] *Chair of Logistics Management, Gutenberg School of Management and Economics, Johannes Gutenberg University Mainz, Jakob-Welder-Weg 9, D-55128 Mainz, Germany.*

Abstract

The vehicle routing problem with drones and time windows (VRP-DTW) is a generalization of the vehicle routing problem with drones, in which each customer is associated with a predefined time window that specifies the delivery time for the customer's shipment. Most literature on the VRP-DTW considers the minimization of the total routing cost as objective function, which is the standard objective for vehicle routing problems. However, the presence of time windows gives rise to the analysis of alternative objective functions, such as minimizing the completion time or route duration. We present a branch-price-and-cut algorithm to solve the VRP-DTW with the objective to minimize the sum of route duration over all routes. The column-generation pricing problem is modeled as a shortest path problem with resource constraints (SPPRC) that can handle the implications of non-robust cuts and branching decisions. The SPPRC is solved using an effective dynamic programming labeling algorithm on an artificial network. Non-robust capacity cuts and dynamic neighborhood extensions are used to strengthen the linear relaxation. In a computational study, we investigate the algorithmic components and show that the proposed algorithm can solve instances with up to 30 customers to proven optimality within two hours of computation time. The gap over all considered instances is less than 0.5% on average. We also present managerial insights that analyze the impact of combined truck-and-drone routing compared to classical truck routing, and show that minimizing the route duration leads to considerable time savings compared to minimizing the completion time.

Keywords: routing, drone deliveries, time windows, route duration, branch-price-and-cut

1. Introduction

In synchronized truck-and-drone routing problems, one or several trucks are equipped with a single or multiple *unmanned aerial vehicles* (UAVs, also referred to as *drones*) to fulfill a set of transportation requests. Drones can usually reach customers faster, easier, and more environmentally friendly than a truck, but they have the disadvantages of a limited flying range and payload, i.e., they cannot serve long-haul transportation requests or deliver heavy packages (Joerss *et al.*, 2016; Goodchild and Toy, 2018). However, the combination of two different types of vehicles with complementary characteristics allows the reduction of delivery times and transportation costs, can simplify access to certain customers, and increases the efficiency of performing transportation requests (Wang *et al.*, 2017; Poikonen *et al.*, 2017; Carlsson and Song, 2018). Thus, this topic attracts the research community (Otto *et al.*, 2018; Chung *et al.*, 2020; Macrina *et al.*, 2020; Moshref-Javadi and Winkenbach, 2021; Madani and Ndiaye, 2022) as well as several delivery companies such as Amazon or UPS (Amazon Prime Air, 2023; Federal Aviation Administraion, 2023).

The (basic) *vehicle routing problem with drones* (VRP-D) is one example of a synchronized truck-and-drone routing problem. Herein, a fleet of homogeneous trucks –each equipped with a single drone– is

*Corresponding author.
Email address: sjeanett@uni-mainz.de (Jeanette Schmidt)

stationed at a depot and can be used to serve a given set of customers. Therefore, a truck and its drone can operate either together or separately. When operating *together*, the drone is inside or on top of the truck, and the truck is responsible for serving the customer. Whenever it is beneficial, the truck can release its drone (at a customer location or the depot) so that both vehicles can serve customers in parallel. While the drone is airborne, the truck continues its route *alone* to serve one or several customers. Meanwhile, the drone serves exactly one customer and *returns* to the truck. In the basic version, it is assumed that the assignment of trucks and drones is fixed, i.e., a drone always has to return to the truck from which it was released. Additionally, a drone is not allowed to *loop*, i.e., the truck must continue its journey while the drone is airborne. The VRP-D aims to find a cost-minimal set of truck-and-drone routes that start and end at the depot, visit each customer exactly once, and respect the capacity of the truck and drone on each route (Schmidt *et al.*, 2023). The basic VRP-D can be generalized in several ways, e.g., by relaxing the loop constraint (Zhou *et al.*, 2023), by relaxing the fixed assignment of trucks and drones (Bakir and Tiniç, 2020), or by limiting the flying range of the drones (Zhen *et al.*, 2023).

In this paper, we consider the *vehicle routing problem with drones and time windows* (VRP-DTW), which is also a generalization of the VRP-D where the service at each customer must start within a predefined time interval, called a *time window*. An additional time window at the depot spans the planning horizon and limits the departure and arrival times for truck and drone. Time windows are present in many routing problems, as they can represent real-life situations such as working hours, opening hours, or agreed delivery times. The number of publications concerning a routing problem with time windows is huge, but there are only a few publications considering time windows in a VRP-D variant. Almost all of these publications aim at solving the VRP-DTW with the objective to minimize the total routing cost, including different cost components such as fixed and variable costs for trucks and/or drones, costs for waiting times, or CO_2 emissions. However, the introduction of time windows allows the investigation of alternative objective functions such as minimizing the *completion time* or the *route duration* (Savelsbergh, 1992). When considering the completion time of a route, truck and drone leave the depot directly upon opening the time window to start the delivery process, generating an *as-early-as-possible schedule*. In contrast, when considering the route duration, the departure time at the depot is not fixed. Both types of vehicles can depart at any time within the depot's time window to deliver the customers, creating an *as-late-as-possible schedule*. Minimizing the completion time may not always be appropriate for truck-and-drone routing, as it may result in long waiting times at customer locations. Especially, if drones have a limited flying range (in the sense that they cannot be separated from the truck for a specific amount of time) long waiting times can make a drone flight infeasible.

We develop a BPC algorithm to solve the VRP-DTW with the objective to minimize the sum of route duration over all routes. Our BPC algorithm is based on a set-partitioning formulation, as usually done in BPC approaches for vehicle routing problems. The column-generation pricing problem is modeled as a shortest path problem with resource constraints (SPPRC, Irnich and Desaulniers, 2005) and solved by means of a dynamic programming labeling algorithm. The labeling algorithm runs on a *multi-digraph*, originally proposed by Roberti and Ruthmair (2021), in which vertices represent possible truck-and-drone positions and arcs represent potential movements of truck and drone. To track Pareto-optimal schedules, we use resources that are pairwise interdependent and coupled with a max-term (Irnich, 2008; Tilk and Irnich, 2017).

Our contributions are as follows:

- To the best of our knowledge, we developed the first exact algorithm to solve the VRP-DTW with the objective to minimize the sum of route duration over all routes contained in a solution. We equip our algorithm with non-robust capacity cuts (Baldacci *et al.*, 2008) and dynamic neighborhood extensions (Roberti and Mingozzi, 2014; Bode and Irnich, 2015) to strengthen the linear relaxation of the set-partitioning formulation. Integer solutions are obtained by means of a four-stage branching strategy. Our SPPRC pricing problem provides the flexibility to handle all implications of the non-robust cuts and branching decisions.
- Synchronizing truck and drone to obtain a route with minimal duration is a non-trivial task. Neither the as-late-as-possible schedule nor the as-early-as-possible schedule are appropriate. We propose a method on how to properly synchronize both types of vehicles to obtain a route with minimal duration.

- We present an extensive computational study on newly generated instances that evaluates the algorithmic components and demonstrates the efficacy of the overall BPC algorithm. Finally, our algorithm is able to solve instances with up to 30 customers to proven optimality within a time limit of two hours. The remaining gap over all instances is less than 0.5% on average.
- We also provide managerial insights that analyze the impact of combined truck-and-drone routing compared to classical truck routing, as well as the impact of different types of drones that differ only in their flying range. In addition, we analyze the time savings that can be achieved by minimizing the route duration rather than the completion time.

The remainder of this paper is organized as follows: At first, we provide a brief overview of the related scientific literature in Section 2. Afterwards, we formally define the VRP-DTW in Section 3. Section 4 presents the BPC algorithm to solve the VRP-DTW with a specific focus on the solution of the pricing problem. Computational results and managerial insights are presented in Section 5. Finally, in Section 6, we conclude and suggest future research directions.

2. Literature Review

Synchronized truck-and-drone routing is a popular research topic. The review articles by Otto *et al.* (2018); Chung *et al.* (2020); Macrina *et al.* (2020); Moshref-Javadi and Winkenbach (2021); Madani and Ndiaye (2022) show the huge amount of publications in the last years and new publications continue to appear. It would go far beyond the scope of this paper to list them all. Therefore, we focus only on publications that consider a variant of the VRP-DTW. For a review of VRP-D variants, we refer the reader to Schmidt *et al.* (2023), and for a more general overview of (synchronized) truck-and-drone routing, we refer to one of the review articles mentioned above.

The first work considering time windows in a VRP-D variant was published by Pugliese and Guerriero in 2017. The authors developed a mixed-integer linear program (MIP) to solve the VRP-DTW with multiple drones per truck with the objective of minimizing the total routing cost. The commercial MIP solver CPLEX can solve small instances of five and 10 customers in a matter of seconds. Their work was later extended by Pugliese *et al.* (2020), in which the authors investigated several variants of the problem, such as relaxed time window constraints for each customer served by a drone, or a limit on the number of drones associated with each truck. In addition to a MIP formulation, the authors developed a heuristic algorithm that embeds a two-phase strategy in a multi-start framework. The MIP can be solved to proven optimality for instances with up to 15 customers within a computation time of 30 minutes. The multi-start heuristic can find almost all optimal solutions obtained by the MIP in a considerably shorter computation time. In addition, large instances with up to 100 customers are solved heuristically. In a second follow-up paper, Pugliese *et al.* (2021) focused on different energy consumption functions in case of adverse weather conditions while a drone is airborne. They proposed a Benders' decomposition approach to solve the VRP-DTW under uncertain energy consumption and presented results on instances with 10 and 15 customers. Coindreau *et al.* (2021) investigated a version of the VRP-DTW that also allows a drone to loop. They developed a MIP formulation and an adaptive large neighborhood search (ALNS) to minimize the total routing cost. CPLEX solves instances with up to 20 customers within a time limit of 10 hours, while the proposed ALNS solves instances with up to 100 customers in less than a minute of computation time. In the publication by Das *et al.* (2021), each customer can have multiple, prioritized time windows that are assumed to be soft. They developed an ant colony optimization heuristic and a genetic algorithm, both aimed at solving a multi-objective function. More precise, the heuristics aim to minimize the total routing cost and maximize the customer service level in terms of on-time deliveries. Both heuristics can solve instances with 25 and 50 customers from the Solomon benchmark set (Solomon, 1987) in less than 10 minutes of computation time. Kuo *et al.* (2022) presented another MIP formulation and a variable neighborhood search (VNS) to solve the VRP-DTW with the objective to minimize the total routing cost. The Gurobi MIP solver solves instances with a maximum of 10 customers within a computation time of two hours. The VNS is used to tackle larger instances with up to 50 customers within a time limit of 10 minutes.

To the best of our knowledge, Li and Wang (2022) developed the first exact BPC algorithm to solve the VRP-DTW with the objective to minimize the total routing cost. It solves instances with up to 50

customers to proven optimality within a time limit of 20 hours. They also combine the BPC algorithm with an ALNS to tackle large-sized instances with up to 100 customers. The time limit for the combined BPC algorithm is set to 24 hours. Another exact algorithm was proposed by Yin *et al.* (2023). They consider a variant of the VRP-DTW in which the drone is allowed to serve more than one customer, as long as it does not exceed its capacity or maximum flying range. They developed a full-fledged BPC algorithm that uses a bidirectional labeling algorithm to solve the pricing problem to minimize the total routing cost. Instances with up to 35(45) customers can be solved within a computation time of two(three) hours.

Li *et al.* (2020) and Zhou *et al.* (2023) studied another variant of the VRP-DTW, in which each truck is equipped with more than one drone. Whenever the truck releases one or several drones, it must wait until all the launched drones have returned before it can serve the next customer. In such a setting, the synchronization constraints between a truck and its drones are different compared to all VRP-DTW variants mentioned above. Li *et al.* (2020) presented a MIP formulation together with an ALNS. Small instances with up to 12 customers can be solved with CPLEX within four hours of computation time. For the same set of instances, the proposed ALNS finds the optimal solution in less than 10 seconds. Additionally, the authors run their ALNS algorithm on instances with up to 100 customers and obtain solutions in less than 1500 seconds. Zhou *et al.* (2023) developed a BPC algorithm that uses a bidirectional labeling algorithm to solve the pricing problem. Instances with up to 35 customers can be solved to proven optimality within a time limit of three hours.

The works of Chen *et al.* (2021a) and Chen *et al.* (2021b) consider delivery robots instead of drones. A matheuristic (Chen *et al.*, 2021b) and an ALNS (Chen *et al.*, 2021a) were developed to solve the problems heuristically.

3. The Vehicle Routing Problem with Drones and Time Windows

The VRP-DTW is defined on a complete directed graph $G = (V, A)$ with vertex set V and arc set A. The vertex set $V = N \cup \{0, 0'\}$ comprises the set of *customers* N and the *start* and *end depot* $\{0, 0'\}$. Associated with each customer $v \in N$ is an integer *demand* $q_v > 0$ and a *time window* $[e_v, l_v]$ in which the *service* of length $\sigma_v > 0$ has to start. We consider the time windows to be hard, i.e., if a vehicle (independent of its type) arrives earlier than e_v to serve customer $v \in N$ the start of the service is delayed to time e_v. It is not allowed to start the service later than l_v. The time windows at the start and end depot $[e_0, l_0] = [e_{0'}, l_{0'}]$ represent the planning horizon. We assume that there is no service and demand at the depot, i.e., $q_0 = q_{0'} = \sigma_0 = \sigma_{0'} = 0$. The depot houses a homogeneous fleet of K trucks. Each truck has a *capacity* Q and is equipped with a single drone. The *capacity of a drone* is limited by Q^{dr} and in the sense that it can only serve one customer before returning to the truck. Each drone has a maximum *flying range* δ, i.e., the drone cannot remain airborne for more than δ time units. We assume that a drone always waits on the ground rather than hovering, so there is no energy consumption that affects its flying range. Both types of vehicles can be used to serve the set of customers. For this purpose, each truck and its drone can operate together or separately. To operate *together*, the drone is either inside or on top of the truck while the truck *serves* the customer. To operate *separately*, the truck can release a drone at any customer location (or at the start depot) $v \in N \cup \{0\}$. While the drone is airborne, the truck continues its journey *alone* to serve one or several customers. After serving exactly one customer, the drone must *return* to the same truck from which it was released, at the truck's current location $w \in N \cup \{0'\}$ with $w \neq v$. We assume that take-off and landing of a drone is completely autonomous and does not require any interaction from the truck driver. This allows for take-off and landing at any time, even while the truck is serving a customer. In addition, take-off and landing are independent of the customer time windows $[e_v, l_v]$ and $[e_w, l_w]$ at the release and landing positions v and w with $w \neq v$, as they are only relevant for the truck serving that customers. The drone can take off as soon as the truck arrives at a customer, without waiting for the truck to finish its service or for the time window to open.

Associated with each arc $(v, w) \in A$ is a *travel time* $t_{vw} > 0$ for the truck and $t^{\mathrm{dr}}_{vw} > 0$ for the drone. It is assumed that the triangle inequality holds for all travel times t_{vw} and t^{dr}_{vw}. Times for take-off and landing are considered to be negligible.

A *route* $r = (P, D)$ is defined as a pair of a *truck path* P and a sequence of (possibly empty) *drone subpaths* D together with corresponding *time schedules* $S = (T(P), U(D))$. A truck path $P = (0, v_1, \ldots, v_m, 0')$ with $m \geq 1$ is a path in G, starting at the start depot 0, visiting a sequence of customers $N(P) = \{v_1, \ldots, v_m\}, N(P) \subseteq N$ and returning to the end depot $0'$. The drone subpaths $D = (D^1, \ldots, D^b)$ specify unique drone flights along the truck path P. Each drone subpath $D^s = \langle v^s, k^s, w^s \rangle$ for all $s \in \{1, \ldots, b\}$ is represented by a triplet: at $v^s \in \{0\} \cup N$ the drone leaves the truck, $k^s \in N$ specifies the customer that is served by the drone, and $w^s \in N \cup \{0'\}$ indicates where truck and drone meet each other again.
For each drone subpath s, we define two indices $g_s, h_s \in \{0, 1, \ldots, m, m+1\}$ with $g_s < h_s$. For convenience, the set $I(s) = \{g_s, g_s + 1, \ldots, h_s - 1\}$ defines the indices of the truck path P where the drone leaves the truck g_s and the truck serves a customer alone $g_s + 1, \ldots, h_s - 1$. Note that the position where the drone returns to the truck is not included. For two drone subpaths $s, s' \in \{1, \ldots, b\}$ with $s < s' : g_s < h_s \leq g_{s'} < h_{s'}$. For each truck-and-drone path, there exist a corresponding time schedule. The schedule $T(P) = (T_0, T_1, \ldots, T_m, T_{0'})$ with $m \geq 1$ represents the service start times at each customer $N(P)$ together with the departure and arrival times at the start and end depot, respectively. Additionally, there exists a time schedule $U(D) = (U(D^1), \ldots, U(D^b))$ for each drone subpath. Each drone time schedule $U(D^s) = \langle U_{v^s}, U_{k^s}, U_{w^s} \rangle$ for all $s \in \{1, \ldots, b\}$ is represented by an additional triplet: the drone takes off at time U_{v^s} from its current position v^s, the service at customer k^s starts at U_{k^s}, and the drone meets the truck at time U_{w^s}. As a reminder, U_{v^s} and U_{w^s} are not restricted by the time windows of both customers v^s and w^s.

A route $r = (P, D)$ is *elementary* if all elements in $N(P)$ and all k^s for all $s \in \{1, \ldots, b\}$ are different and it is *feasible* if the following conditions hold:

i) $\sum_{p=1}^{m} q_{v_p} + \sum_{p=s}^{b} q_{k^s} \leq Q$;

ii) $q_{k^s} \leq Q^{\mathrm{dr}}$ for all $s \in \{1, \ldots, b\}$;

iii) $T_p \in [e_{v_p}, l_{v_p}]$ for all $p \in \{0, \ldots, m+1\}$;

iv) $U_{k^s} \in [e_{k^s}, l_{k^s}]$ for all $s \in \{1, \ldots, b\}$;

v) $T_{p-1} + \sigma_{p-1} + t_{v_{p-1}, v_p} \leq T_p$ for all $p \in \{1, \ldots, m+1\}$;

vi) $U_{v^s} + t^{\mathrm{dr}}_{v^s, k^s} \leq U_{k^s}$ and $U_{k^s} + \sigma_{k^s} + t^{\mathrm{dr}}_{k^s, w^s} \leq U_{w^s}$ for all $s \in \{1, \ldots, b\}$;

vii) $t^{\mathrm{dr}}_{v^s, k^s} + t^{\mathrm{dr}}_{k^s, w^s} \leq \delta$ for all $s \in \{1, \ldots, b\}$;

viii) $I(s) \cap I(s') = \varnothing$ for $1 \leq s < s' \leq b$ with $b \geq 2$;

ix) $T_{g_{s-1}} + \sigma_{g_{s-1}} + t_{g_{s-1} g_s} \leq U_{v^s}$ for all $s \in \{1, \ldots, b\}$;

x) $U_{v^s} \leq T_{g_{s+1}} - t_{g_s g_{s+1}}$ for all $s \in \{1, \ldots, b\}$;

xi) $U_{w^s} \leq T_{v_{h+1}} - t_{v_h, v_{h+1}}$ for all $s \in \{1, \ldots, b\}$;

These conditions ensure that each route respects the capacity of the truck and the drone (i, ii), the customer time windows (iii, iv), and travel times (v, vi). Condition (vii) ensures that a drone is never airborne longer than allowed. Additionally, constraints (viii) – (xi) ensure a synchronization of truck and drone in space (viii) and time (ix – xi).

The *route duration* is defined by the *time span* between the earliest departure x of truck or drone at the start depot, i.e., $x = \min(T_0, U_{v^1})$, and the arrival of the latest vehicle at the end depot, i.e., $y = \max(T_{0'}, U_{w^d})$. Therefore, the duration of a route r is defined as $d_r = (y - x)$.

The VRP-DTW asks for a set Ω of at most K feasible routes such that each customer is served by a single vehicle and the summarized duration over all routes $\sum_{r \in \Omega} d_r$ is minimized.

Example 1. *In the following, we consider a feasible route* $r = (P, D)$ *with* $P = (0, 1, 2, 3, 5, 0')$ *and* $D = (\langle 2, 4, 3\rangle, \langle 3, 6, 0'\rangle)$. *Time windows* $[e_v, l_v]$ *along with travel times* t_{vw} *and* t^{dr}_{vw} *are given in Table 1 (columns 2+3 and 7+8). Further, we assume a demand* $q_v = 2$ *for all* $v \in N = \{1, \ldots, 6\}$ *and a capacity* $Q = 12$ *for the truck. The drone's capacity is set to* $Q^{dr} = 5$ *and its flying range to* $\delta = 5$ *so that each customer can be served by a drone. The service time for each customer* $v \in N$ *is set to* $\sigma_v = 1$.

P	$[e_v, l_v]$	$t_{v,v+1}$	$T(P)_{alap}$	$T(P)_{aeap}$	D	$[e_v, l_v]$	$t^{\mathrm{dr}}_{v,v+1}$	$U(D)_{alap}$	$U(D)_{aeap}$
0	$[0, 25]$	1	5	0	2	—	2	10	7
1	$[3, 6]$	3	6	3	4	$[10, 12]$	3	12	10
2	$[8, 10]$	3	10	8	3	—	—	16	14
3	$[12, 14]$	3	14	12					
5	$[17, 20]$	3	19	17	3	—	2	16	14
$0'$	$[0, 25]$	—	23	21	6	$[18, 20]$	3	18	18
					$0'$	—	—	22	22

Table 1: VRP-DTW instance with a feasible solution.

The columns $T(P)_{alap}$ *and* $U(D)_{alap}$ *show the truck-and-drone schedules for the given route according to an as-late-as-possible schedule. The truck leaves the depot at time 5 for customer 1. It arrives there at time 6, which is the latest possible time to start service. After completing the service, the truck drives towards customer 2, where it arrives at time 10, which is again the latest possible time to start the service. Now, truck and drone separate. The truck continues its journey alone to customer 3, arrives at time 14 and performs service until time 15. In parallel, the drone takes off from customer 2, serves customer 4, and meets the truck at customer 3 at time 16. Note that the drone is allowed to meet the truck later than* $l_3 = 14$ *because it is not restricted to that time window. The truck has to wait for the drone, before they separate again. The truck continues serving customer 5 at time 19 and arrives at the depot at time 23. Meanwhile, the drone serves customer 6 and arrives at the depot* $0'$ *at time 22. The duration of this route is* $\max(23, 22) - 5 = 18$.

In contrast, an as-early-as-possible schedule can be determined as follows: The truck leaves the depot immediately at time $e_0 = 0$ *and arrives at customer 1 at time 1. It must wait until time 3 to start service. The truck then continues to customer 2, where it has to wait for one time unit to start service. While waiting for service, the drone is permitted to take off to customer 4. Note that the drone is allowed to do so because it is not restricted to the time window at customer 2. Starting at time 7, serving customer 4 at time 10 (including one unit of waiting time), the drone lands at customer 3 at time 14. While the drone is airborne, the truck serves customers 2 and 3 and waits for one time for the drone to arrive. After both vehicles are synchronized, the truck continues to customer 5 and* $0'$*, where it arrives at time 21. The drone arrives at customer 6 at time 16 and has to wait for 2 time units to start service at time 18. After serving the customer, the drone reaches the depot at time 22. The completion time, according to an as-early-as-possible schedule is* $\max(21, 22) = 22$. *The example shows, that starting later than* e_0 *reduces the delivery time by 4 time units, which is a time saving of approx. 18%.*

4. Branch-Price-and-Cut Algorithm

This section describes the BPC algorithm for the solution of the VRP-DTW. Section 4.1 presents a straightforward set-partitioning formulation that will be used as *master problem* (MP) in the column-generation process. Section 4.2 discusses the modeling and solution of the column-generation subproblem. In particular, we present the underlying network in Section 4.2.1, discuss the labeling algorithm in Sections 4.2.2 and 4.2.3, and present effective dominance rules in Section 4.3. Afterwards, Section 4.4 shows implemented acceleration techniques for the overall BPC algorithm. Valid inequalities used to strengthen the linear relaxation of the MP are presented in Section 4.5. Finally, Section 4.6 discusses the branching rules to obtain integer solutions.

4.1. Route-based Formulation

Let Ω denote the set of all VRP-DTW-routes that are feasible with respect to the capacity, time window, and synchronization constraints. Further, let d_r denote the duration of a route r as defined in Section 3 and λ_r be a binary variable equal to 1 if and only if the route $r \in \Omega$ is selected in the solution. The coefficient a_{vr} indicates the number of times customer $v \in N$ is served by route r. The VRP-DTW can be stated as follows:

$$\min \quad \sum_{r \in \Omega} d_r \lambda_r \tag{1a}$$

$$\text{subject to} \quad \sum_{r \in \Omega} a_{vr} \lambda_r = 1 \qquad \forall v \in N \tag{1b}$$

$$L \leq \sum_{r \in \Omega} \lambda_r \leq K \tag{1c}$$

$$\lambda_r \in \{0, 1\} \qquad \forall r \in \Omega \tag{1d}$$

The objective function (1a) minimizes the summarized duration over all routes. The set-partitioning constraints (1b) ensure that each customer is served exactly once. Due to (1c), the number of trucks in use ranges between a lower bound (L, see Section 4.6) and the fleet size K. The domain of all route variables is specified in (1d).

The model defined by (1) contains a huge number of feasible routes so that it is almost impossible (and not necessary) to generate all such routes beforehand. Instead, the linear relaxation of this model, i.e., $\lambda_r \geq 0$, is solved by means of *column generation* (Desaulniers *et al.*, 2005). Its basic idea is to work with a *restricted master problem* (RMP) that considers only a (small) subset of feasible routes. It can be obtained by replacing Ω with a subset $\bar{\Omega} \subset \Omega$ of generated routes. Missing routes are generated by solving one or several *subproblems*, also called *pricing problems*. A solution of the linear relaxation of the MP can be obtained by iteratively re-optimizing the RMP and solving the pricing problem(s), until no more beneficial routes, i.e., routes with negative reduced cost, can be added to the RMP.

4.2. Column Generation

The column-generation pricing problem aims to generate beneficial routes or to prove that no such route exists. A route is beneficial if it is feasible and its reduced cost $\tilde{c}_r$ is negative. Let $(\pi_v)_{v \in N}$ denote the dual prices of the set-partitioning constraints (1b) and let μ be the dual price of fleet-size constraint (1c). The *reduced cost* of an arbitrary route $r \in \Omega$ can be calculated as

$$\tilde{c}_r = d_r - \sum_{v \in N} a_{vr} \pi_v - \mu.$$

This pricing problem can be modeled as *shortest path problem with resource constraints* (SPPRC, Irnich and Desaulniers, 2005) and solved with a labeling algorithm.

4.2.1. Artificial Network

We run our labeling algorithm on an artificial network originally introduced by Roberti and Ruthmair (2021) to solve the *traveling salesman problem with drone* (TSP-D). The work by Schmidt *et al.* (2023) shows that this network is also suitable to solve the multiple-vehicle version of this problem, the VRP-D. For the sake of completeness, we explain the most important characteristics of this network and refer the reader to Roberti and Ruthmair (2021) or Schmidt *et al.* (2023) for further details. From now on, we stay close to the notation used in Schmidt *et al.* (2023).

The *artificial network* $\mathcal{N} = (\mathcal{W}, \mathcal{A})$ contains a set of vertices $\mathcal{W}$ and arcs $\mathcal{A}$. The vertex set

$$\mathcal{W} = \{(0,0) \cup (\{0\} \cup N) \times (N \cup \{0'\}) \cup (0', 0')\} \subset V \times V$$

represents all possible truck-and-drone positions. Each vertex $i = (i^{\text{tr}}, i^{\text{dr}}) \in \mathcal{W}$ contains two information: the current truck position i^{tr} and the current drone position i^{dr}.

The arc set $\mathcal{A}$ represents movements of truck and drone between two vertices $i = (i^{\text{tr}}, i^{\text{dr}})$ and $j = (j^{\text{tr}}, j^{\text{dr}})$ with $i, j \in \mathcal{W}, i \neq j$ together with an (possible) additional service by the drone, denoted by $k \in N^{dr} = \{v \in N : q_v \leq Q^{\text{dr}}\} \cup \{\perp\}$, where $\perp$ represents that no feasible service can be performed. Since the artificial network is a directed *multi-graph*, each arc $\mathcal{A}$ can be uniquely described by a triplet $[i, j, k] = [(i^{\text{tr}}, i^{\text{dr}}), (j^{\text{tr}}, j^{\text{dr}}), k]$. All arcs of the artificial network $\mathcal{N}$ can be divided into three disjoint categories:

(i) *together arcs*

$$\mathcal{A}^{tog} = \left\{[i, j, \perp] \in \mathcal{W} \times \mathcal{W} \times \{\perp\} : i^{\text{tr}} = i^{\text{dr}} \neq j^{\text{tr}} = j^{\text{dr}}\right\}$$

represent that truck and drone are moving together from $i^{\text{tr}} = i^{\text{dr}}$ to a new position $j^{\text{tr}} = j^{\text{dr}}$;

(ii) *truck alone arcs*

$$\mathcal{A}^{alone} = \left\{[i, j, \perp] \in \mathcal{W} \times \mathcal{W} \times \{\perp\} : i^{\text{dr}} = j^{\text{dr}} \text{ and } i^{\text{tr}} \neq j^{\text{tr}} \text{ and } i^{\text{dr}} \neq j^{\text{tr}}\right\}$$

represent movements of the truck from customer i^{tr} to another customer j^{tr} while the drone is airborne (release position $i^{\text{dr}} = j^{\text{dr}}$); and

(iii) *drone arcs*

$$\mathcal{A}^{drone} = \left\{[i, j, k] \in \mathcal{W} \times \mathcal{W} \times N^{dr} : i^{\text{tr}} = j^{\text{tr}} = j^{\text{dr}} \neq i^{\text{dr}} \text{ and } k \notin \{i^{\text{dr}}, i^{\text{tr}}, \perp\}\right\}$$

represent drone movements. Here, the drone starts from its release position i^{dr} to serve customer k and meets the truck at the truck's current position $j^{\text{tr}} = j^{\text{dr}} \neq i^{\text{dr}}$.

Every path in $\mathcal{N}$, starting at $(0, 0)$, visiting a sequence of vertices, and ending at $(0', 0')$ represents a joint truck-and-drone route. Note that this route is not necessarily elementary or feasible. The labeling algorithm presented in Section 4.2.2 ensures final elementary and feasibility.

Pre-Processing. Before running the labeling algorithm on $\mathcal{N}$, the graph can be pre-processed by removing all capacity and time window infeasible arcs that cannot be part of a feasible solution. First, if customers i and j are both served by a truck, we check if serving customer j after customer i is feasible with respect to their time windows. Otherwise the arc $[i, j, \perp]$ can be eliminated from $\mathcal{A}$ so that

$$\mathcal{A}^{tog} \subset \{[i, j, \perp] \in \mathcal{W} \times \mathcal{W} \times \{\perp\} : e_{i^{\text{tr}}} + \sigma_{i^{\text{tr}}} + t_{i^{\text{tr}} j^{\text{tr}}} \leq l_{j^{\text{tr}}}\} \text{ and}$$

$$\mathcal{A}^{alone} \subset \{[i, j, \perp] \in \mathcal{W} \times \mathcal{W} \times \{\perp\} : e_{i^{\text{tr}}} + \sigma_{i^{\text{tr}}} + t_{i^{\text{tr}} j^{\text{tr}}} \leq l_{j^{\text{tr}}}\}.$$

Unfortunately, it is not possible to eliminate drone arcs in the same way, since the drone is allowed to start earlier than $e_{i^{\text{dr}}}$ at its release position i^{dr}. Second, if the demand of all customers along an arc $[i, j, k] \in \mathcal{A}$ is greater than the truck's capacity, the arc can be eliminated due to capacity restrictions, i.e.,

$$\mathcal{A}^{tog} \subset \{[i, j, \perp] \in \mathcal{W} \times \mathcal{W} \times \{\perp\} : q_{i^{\text{tr}}} + q_{j^{\text{tr}}} \leq Q\},$$

$$\mathcal{A}^{alone} \subset \{[i, j, \perp] \in \mathcal{W} \times \mathcal{W} \times \{\perp\} : q_{i^{\text{tr}}} + q_{j^{\text{tr}}} \leq Q\}, \text{ and}$$

$$\mathcal{A}^{drone} \subset \left\{[i, j, k] \in \mathcal{W} \times \mathcal{W} \times N^{dr} : q_{i^{\text{tr}}} + q_{j^{\text{tr}}} + q_k \leq Q\right\}.$$

In addition, all drone arcs that do not respect the drone's flying range δ can also be eliminated, i.e.,

$$\mathcal{A}^{drone} \subset \left\{[i, j, k] \in \mathcal{W} \times \mathcal{W} \times N^{dr} : t^{\text{dr}}_{i^{\text{dr}}, k} + t^{\text{dr}}_{k, j^{\text{dr}}} \leq \delta\right\}.$$

4.2.2. Labeling Algorithm

In our labeling algorithm for the VRP-DTW, each label $\mathcal{L}_i$ represents a partial path $\mathcal{P}_i = ((0,0), \ldots, i = (i^{tr}, i^{dr}))$ in $\mathcal{N}$ that starts at the depot $(0,0)$ and ends at some artificial vertex $i = (i^{tr}, i^{dr}) \in \mathcal{N}$. Each element stored in a label is called an *attribute* R_i and represents the resource consumption along the partial path. Starting from an initial label $\mathcal{L}_0$ at the depot $0 = (0,0)$, a labeling algorithm propagates labels toward the depot $(0', 0')$ over all three types of arcs $\mathcal{A} = \mathcal{A}^{tog} \cup \mathcal{A}^{alone} \cup \mathcal{A}^{drone}$ with the help of *resource extension functions* (REFs, Irnich, 2008). To avoid enumerating all feasible paths, some labels can be eliminated by a dominance criterion.

A label $\mathcal{L}_i$ at an artificial vertex $i = (i^{\mathrm{tr}}, i^{\mathrm{dr}})$ comprises the following attributes:

R_i^{rdc}: the accumulated *dual prices* along $\mathcal{P}$;
R_i^{load}: the accumulated *load* along $\mathcal{P}$;
$R_i^{tme,ar}$: the earliest *arrival time* at vertex i;
$R_i^{tme,dp}$: the earliest *departure time* at vertex i, i.e., including possible waiting times and service;
$R_i^{dur,ar}$: the minimum *route duration* along $\mathcal{P}$ up to vertex i such that each customer in $\mathcal{P}$ is served within its time window;
$R_i^{dur,dp}$: the minimum *route duration* along $\mathcal{P}$, when leaving vertex i;
R_i^{strt}: the negative of the *latest possible departure time* at $(0,0)$ such that each customer in $\mathcal{P}$ can be served within its time window; and
$R_i^{cust,n}$: the number of times that customer $n \in N$ is served along the path.

The attributes $R_i^{rdc}, R_i^{load}, R_i^{tme,ar}, R_i^{tme,dp}$, and $R_i^{cust,n}$ are standard resources in a *vehicle routing problem with time windows* (VRPTW, Desaulniers *et al.*, 2014), while the resources $R_i^{dur,ar}, R_i^{dur,dp}$, and R_i^{strt} are used to generate the as-late-as-possible schedule (Tilk and Irnich (2017); Irnich (2008)). In contrast to the VRPTW, we explicitly distinguish between the arrival and the departure at each vertex i to model the situation where truck and drone have different departure times at a vertex i, e.g., whenever the truck arrives at customer i^{tr}, its earliest departure time is at $e_{i^{\mathrm{tr}}} + \sigma_{i^{\mathrm{tr}}}$, while the drone can leave as soon as the truck arrives (even before the time window opens). We do not distinguish between arrival and departure for the resource R_i^{strt} since the latest possible departure time at the depot is independent from possible waiting and service times at the current vertex i. Note that the resource R_i^{strt} is defined to be negative to get a non-decreasing REF (Tilk and Irnich, 2017). To properly synchronize truck and drone at a later vertex $j = (j^{\mathrm{tr}}, j^{\mathrm{dr}}) \in \mathcal{W}$ with $j^{\mathrm{tr}} = j^{\mathrm{dr}}$ it is important to remember the resource consumption whenever truck and drone separate. Therefore, we use an additional set of attributes $\mathscr{R}_i$ whenever the truck operates alone $\mathscr{R}_i = (\mathscr{R}_i^{tme,ar}, \mathscr{R}_i^{tme,dp}, \mathscr{R}_i^{dur,ar}, \mathscr{R}_i^{dur,dp}, \mathscr{R}_i^{strt})$. The meaning of all attributes in $\mathscr{R}_i$ is the same as for the attributes in R_i, e.g., $\mathscr{R}_i^{tme,ar}$ is the earliest arrival time at vertex i when the *truck travels alone*. The initial label at the depot $0 = (0,0)$ is given by

$$\begin{aligned}
\mathcal{L}_0 &= (R_0^{rdc}, R_0^{load}, R_0^{tme,ar}, R_0^{tme,dp}, R_0^{dur,ar}, R_0^{dur,dp}, R_0^{strt}, (R_0^{cust,n})_{n\in N}, \mathscr{R}_0) \\
&= (0, 0, e_0, e_0, 0, 0, -l_0, \mathbf{0}, \mathscr{R}_0) \text{ with} \\
\mathscr{R}_0 &= (\mathscr{R}_0^{tme,ar}, \mathscr{R}_0^{tme,dp}, \mathscr{R}_0^{dur,ar}, \mathscr{R}_0^{dur,dp}, \mathscr{R}_0^{strt}) \\
&= (e_0, e_0, 0, 0, -l_0)
\end{aligned}$$

Extending an arbitrary label $\mathcal{L}_i$ over an arc $a = [i, j, k] \in \mathcal{A}$ creates a new label $\mathcal{L}_j$ for the corresponding partial path $\mathcal{P}_j = ((0,0), \ldots, i, j = (j^{\mathrm{tr}}, j^{\mathrm{dr}}))$ depending on the particular arc types $\mathcal{A}^{tog}, \mathcal{A}^{alone}$, and $\mathcal{A}^{drone}$. Thus, we distinguish three different types of REFs depending on each arc type:

(i) propagating a label $\mathcal{L}_i$ over a *together arc* $a = [i, j, \perp] \in \mathcal{A}^{tog}$ results in label $\mathcal{L}_j$ with

$$R_j^{rdc} = R_i^{rdc} - \pi_{j^{\mathrm{tr}}}, \tag{2a}$$
$$R_j^{load} = R_i^{load} + q_{j^{\mathrm{tr}}}, \tag{2b}$$

$$R_j^{tme,ar} = R_i^{tme,dp} + t_{i^{\mathrm{tr}}j^{\mathrm{tr}}}, \tag{2c}$$
$$R_j^{tme,dp} = \max\{R_j^{tme,ar}, e_{j^{\mathrm{tr}}}\} + \sigma_{j^{\mathrm{tr}}}, \tag{2d}$$
$$R_j^{dur,ar} = R_i^{dur,dp} + t_{i^{\mathrm{tr}}j^{\mathrm{tr}}}, \tag{2e}$$
$$R_j^{dur,dp} = \max\{R_j^{dur,ar}, R_i^{strt} + e_{j^{\mathrm{tr}}}\} + \sigma_{j^{\mathrm{tr}}}, \tag{2f}$$
$$R_j^{strt} = \max\{R_j^{dur,ar} - l_{j^{\mathrm{tr}}}, R_i^{strt}\}, \tag{2g}$$
$$R_j^{cust,n} = \begin{cases} R_i^{cust,n} + 1, & \text{if } n = j^{\mathrm{tr}} \\ R_i^{cust,n}, & \text{otherwise.} \end{cases} \tag{2h}$$

Additionally, all attributes in $\mathscr{R}_j$ are set to zero: $\mathscr{R}_j^{tme,ar} = \mathscr{R}_j^{tme,dp} = \mathscr{R}_j^{dur,ar} = \mathscr{R}_j^{dur,dp} = \mathscr{R}_j^{strt} = 0$;

(ii) propagating a label $\mathcal{L}_i$ over a *truck alone arc* $a = [i, j, \perp] \in \mathcal{A}^{alone}$ results in label $\mathcal{L}_j$ with

$$R_j^{rdc} = R_i^{rdc} - \pi_{j^{\mathrm{tr}}}, \tag{3a}$$
$$R_j^{load} = R_i^{load} + q_{j^{\mathrm{tr}}}, \tag{3b}$$
$$\mathscr{R}_j^{tme,ar} = \begin{cases} R_i^{tme,dp} + t_{i^{\mathrm{tr}}j^{\mathrm{tr}}}, & \text{if } \mathscr{R}_i^{tme,dp} = 0 \\ \mathscr{R}_i^{tme,dp} + t_{i^{\mathrm{tr}}j^{\mathrm{tr}}}, & \text{if } \mathscr{R}_i^{tme,dp} > 0 \end{cases} \tag{3c}$$
$$\mathscr{R}_j^{tme,dp} = \max\{\mathscr{R}_j^{tme,ar}, e_{j^{\mathrm{tr}}}\} + \sigma_{j^{\mathrm{tr}}}, \tag{3d}$$
$$\mathscr{R}_j^{dur,ar} = \begin{cases} R_i^{dur,dp} + t_{i^{\mathrm{tr}}j^{\mathrm{tr}}}, & \text{if } \mathscr{R}_i^{tme,dp} = 0 \\ \mathscr{R}_i^{dur,dp} + t_{i^{\mathrm{tr}}j^{\mathrm{tr}}}, & \text{if } \mathscr{R}_i^{tme,dp} > 0 \end{cases} \tag{3e}$$
$$\mathscr{R}_j^{dur,dp} = \begin{cases} \max\{\mathscr{R}_j^{dur,ar}, R_i^{strt} + e_{j^{\mathrm{tr}}}\} + \sigma_{j^{\mathrm{tr}}}, & \text{if } \mathscr{R}_i^{tme,dp} = 0 \\ \max\{\mathscr{R}_j^{dur,ar}, \mathscr{R}_i^{strt} + e_{j^{\mathrm{tr}}}\} + \sigma_{j^{\mathrm{tr}}}, & \text{if } \mathscr{R}_i^{tme,dp} > 0 \end{cases} \tag{3f}$$
$$\mathscr{R}_j^{strt} = \begin{cases} \max\{\mathscr{R}_j^{dur,ar} - l_{j^{\mathrm{tr}}}, R_i^{strt}\}, & \text{if } \mathscr{R}_i^{tme,dp} = 0 \\ \max\{\mathscr{R}_j^{dur,ar} - l_{j^{\mathrm{tr}}}, \mathscr{R}_i^{strt}\}, & \text{if } \mathscr{R}_i^{tme,dp} > 0 \end{cases} \tag{3g}$$
$$R_j^{cust,n} = \begin{cases} R_i^{cust,n} + 1, & \text{if } n = j^{\mathrm{tr}} \\ R_i^{cust,n}, & \text{otherwise.} \end{cases} \tag{3h}$$

Whenever truck and drone separate, the attributes in $\mathscr{R}_j$ are used and the attributes in R_j remain unchanged, i.e., $R_j^{tme,ar} = R_i^{tme,ar}$, $R_j^{tme,dp} = R_i^{tme,dp}$, $R_j^{dur,ar} = R_i^{dur,ar}$, $R_j^{dur,dp} = R_i^{dur,dp}$, and $R_j^{strt} = R_i^{strt}$. Thus, the resource consumption up to the drone's release position is stored and whenever the drone returns to the truck, we resume this status; and

(iii) propagating a label $\mathcal{L}_i$ over a *drone arc* $a = [i, j, k] \in \mathcal{A}^{drone}$ results in label $\mathcal{L}_j$ with

$$R_j^{rdc} = R_i^{rdc} - \pi_k, \tag{4a}$$
$$R_j^{load} = R_i^{load} + q_k, \tag{4b}$$
$$R_j^{tme,ar} = \max\left\{\max\{R_i^{tme,ar} + t_{i^{\mathrm{dr}}k}, l_k\} + \sigma_k + t_{kj^{\mathrm{dr}}}, \mathscr{R}_j^{tme,ar}\right\}, \tag{4c}$$
$$R_j^{tme,dp} = \max\left\{\max\{R_i^{tme,ar} + t_{i^{\mathrm{dr}}k}, l_k\} + \sigma_k + t_{kj^{\mathrm{dr}}}, \mathscr{R}_j^{tme,dp}\right\}, \tag{4d}$$
$$R_j^{dur,ar} = \max\left\{\max\{R_i^{dur,ar} + t_{i^{\mathrm{dr}}k}, R_i^{strt} + e_k\} + \sigma_k + t_{k,j^{\mathrm{dr}}}, \mathscr{R}_j^{dur,ar}\right\}, \tag{4e}$$
$$R_j^{dur,dp} = \max\left\{\max\{R_i^{dur,ar} + t_{i^{\mathrm{dr}}k}, R_i^{strt} + e_k\} + \sigma_k + t_{kj^{\mathrm{dr}}}, \mathscr{R}_i^{dur,dp}\right\}, \tag{4f}$$
$$R_j^{strt} = \max\left\{\max\{R_i^{dur,ar} + t_{i^{\mathrm{dr}}k} - l_k, R_i^{strt}\}, \mathscr{R}_i^{strt}\right\}, \tag{4g}$$

$$R_j^{cust,n} = \begin{cases} R_i^{cust,n} + 1, & \text{if } n = k \\ R_i^{cust,n}, & \text{otherwise.} \end{cases} \tag{4h}$$

While propagating over a drone arc, the drone returns to the truck and all resources in $\mathscr{R}_j$ are set to zero again.

We remark that the resources $R^{tme,dp}$ and $\mathscr{R}^{tme,dp}$ are not necessary in the labeling algorithm. They do not affect the feasibility of a path (feasibility is ensured by $R^{tme,ar}$ and $\mathscr{R}^{tme,ar}$, respectively) and can always be calculated from the resources $R^{tme,ar}$ and $\mathscr{R}^{tme,ar}$. They are only considered for the sake of a simpler notation.

A new label $\mathcal{L}_j$ is feasible, if the capacity (5a), time-window (5b)–(5d), and elementary (5e) constraints are fulfilled, i.e.,

$$R_j^{load} \leq Q \qquad \text{if } a \in \mathcal{A}^{alone} \cup \mathcal{A}^{tog} \cup \mathcal{A}^{drone} \tag{5a}$$

$$R_j^{tme,ar} \leq l_{j^{\mathrm{tr}}}, \qquad \text{if } a \in \mathcal{A}^{tog} \tag{5b}$$

$$\mathscr{R}_j^{tme,ar} \leq l_{j^{\mathrm{tr}}}, \qquad \text{if } a \in \mathcal{A}^{alone} \tag{5c}$$

$$R_i^{tme,ar} + t_{i^{\mathrm{dr}}k} \leq l_k, \qquad \text{if } a \in \mathcal{A}^{drone} \tag{5d}$$

$$R_j^{cust,n} \leq 1, \qquad \text{for all } n \in N. \tag{5e}$$

4.2.3. Synchronizing trucks and drones

Synchronizing the duration of truck and drone according to REFs (4e)–(4g) does not necessarily result in a partial path with minimum duration, especially not when truck and drone leave the depot 0 at different times, i.e., $\max\{R_i^{dur,ar} + t_{i^{\mathrm{dr}}k} - l_k, R_i^{strt}\} \neq \mathscr{R}_i^{strt}$ in REF (4g). Whenever the truck or the drone has to leave earlier because of (4g), the REFs (4e)–(4f) do not consider the additional duration for the corresponding vehicle. However, adding the difference of $\max\{R_i^{dur,ar} + t_{i^{\mathrm{dr}}k} - l_k, R_i^{strt}\}$ and $\mathscr{R}_i^{strt}$ to the corresponding duration in (4e)–(4f) does also not always lead to a proper duration. Since both vehicles are scheduled according to an as-late-as-possible schedule, shifting the departure to an earlier point in time can lead to unnecessary waiting times at one or several customers along the path when the service still starts at the latest feasible point in time. Example 2 visualizes both situations.

Example 2. *Given a partial route* $r = (P = (0, 1, 3, \ldots), D = (\langle 0, 2, 3\rangle), \ldots))$ *with time windows* $[e_0, l_0] = [0, 25], [e_1, l_1] = [10, 13], [e_2, l_2] = [6, 8]$, *and* $[e_3, l_3] = [15, 20]$. *The service times are assumed to be* $\sigma_1 = \sigma_2 = 1$ *and* $\sigma_3 = 3$. *All travel times for truck and drone are set to 2.*

Figure 1 depicts two timelines that represent the truck-and-drone movements in route r according to an as-late-as-possible schedule. The bar above each timeline, represents the truck path P and the bar below depicts the drone subpath in D. For the sake of clarity, both paths distinguish between travel times (colored in dark blue), service times (colored in light blue) and waiting times (colored in red). The customer time windows are sketched by curly brackets.

The first timeline shows that using REFs (4e)–(4f) would lead to an incorrect duration of the path $R_3^{dur,dp} = 8$ *that corresponds only to the trucks' duration. It ignores the additional duration caused by leaving the depot at time 6 (shown as a red dashed arc). The second timeline shows that only shifting the truck, i.e., letting the truck start earlier by the difference of* $\max\{R_i^{dur,ar} + t_{i^{\mathrm{dr}}k} - l_k, R_i^{strt}\}$ *and* $\mathscr{R}_i^{strt}$, *leads to unnecessary waiting at customer 1. Although the truck arrives before the time window opens, the service is performed at the latest possible time, resulting in a path duration of* $R_3^{dur,dp} = 13$.

On each vertex j along a path, it is possible to shift the latest possible departure time (for truck or drone) in the direction to e_0 for a specific amount of time without increasing the duration of the path. To be

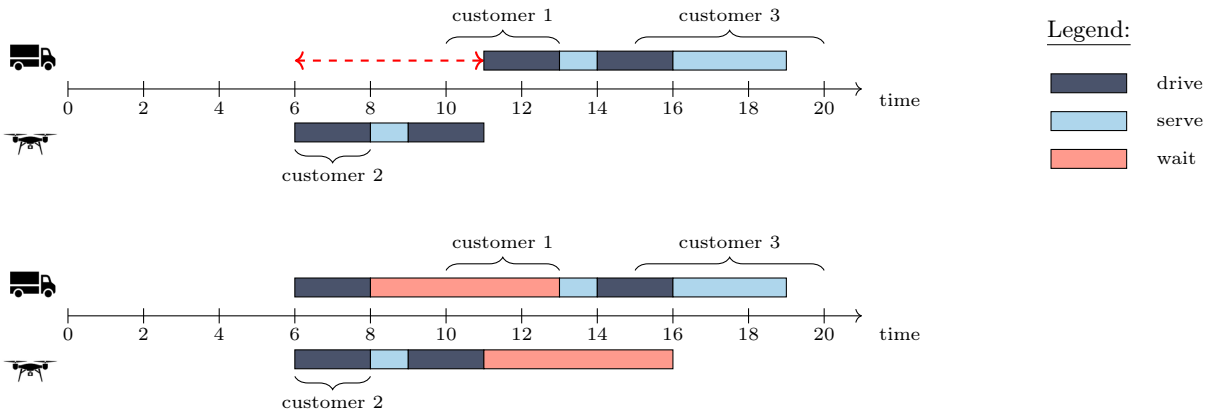

Figure 1: Incorrect synchronization of tuck and drone resulting in an incorrect duration.

more precise, it is possible to shift the departure of a vehicle by the difference between the as-late-as-possible schedule and the as-early-as-possible schedule. Each additional shift in time results in additional waiting times within the path and increases the path duration. Algorithm 1 shows the calculation of the additional duration. Whenever the truck has to leave earlier, the duration on departure as well as the duration on arrival needs to be adjusted. The values Δ and Δ^{ar} represent those values. Whenever the drone has to leave earlier, only the duration on departure Δ needs to be considered, since the duration on arrival at drone customer k is not relevant.

For the sake of simplicity, we define the variable $\alpha^{\mathrm{dr}} = \max\{R_i^{dur,ar} + t_{i^{\mathrm{dr}}k} - l_k, R_i^{strt}\}$ as the latest possible departure time for the drone at node j just before synchronizing with the truck.

Algorithm 1: Determining the additional duration for truck or drone.

Data: $\mathcal{L}_i, a = [i, j, k] \in \mathcal{A}$
Result: additional duration Δ and Δ^{ar}

Initialization: $\Delta = 0, \Delta^{ar} = 0$
$\alpha^{\mathrm{dr}} = \max\{R_i^{dur,ar} + t_{i^{\mathrm{tr}}k} - l_k, R_i^{strt}\}$
`// Drone leaves earlier than truck`
if $\mathscr{R}_i^{strt} < \alpha^{dr}$ **then**
  $\Delta = \left(|\mathscr{R}_i^{strt} - \alpha^{\mathrm{dr}}| - (-\mathscr{R}_i^{strt} - \mathscr{R}_i^{tme,dp} + \mathscr{R}_i^{dur,dp})\right)^+$
  $\Delta^{ar} = \left(|\mathscr{R}_i^{strt} - \alpha^{\mathrm{dr}}| - (-\mathscr{R}_i^{strt} - \mathscr{R}_i^{tme,ar} + \mathscr{R}_i^{dur,ar})\right)^+$
`// Truck leaves earlier than drone`
else if $\mathscr{R}_i^{strt} > \alpha^{dr}$ **then**
  $\Delta = \left(|\mathscr{R}_i^{strt} - \alpha^{\mathrm{dr}}| - (-\alpha^{\mathrm{dr}} - \max\{R_i^{tme,ar} + t_{i^{\mathrm{tr}}k}^{\mathrm{dr}}, l_k\} + \max\{R_i^{dur,ar} + t_{i^{\mathrm{tr}}k}^{\mathrm{dr}}, R_i^{strt} + e_k\})\right)^+$
else
  $\Delta = 0$

After determining the additional duration, we can update the REFs (4e)–(4f) as follows:

$$R_j^{dur,ar} = \begin{cases} \max\left\{\max\{R_i^{dur,ar} + t_{i^{\mathrm{dr}}k}, R_i^{strt} + e_k\} + \sigma_k + t_{k,j^{\mathrm{dr}}} + \Delta, \mathscr{R}_j^{dur,ar}\right\}, & \text{if } \mathscr{R}_i^{strt} > \alpha^{\mathrm{dr}} \\ \max\left\{\max\{R_i^{dur,ar} + t_{i^{\mathrm{dr}}k}, R_i^{strt} + e_k\} + \sigma_k + t_{k,j^{\mathrm{dr}}}, \mathscr{R}_j^{dur,ar} + \Delta^{ar}\right\}, & \text{if } \mathscr{R}_i^{strt} < \alpha^{\mathrm{dr}} \end{cases} \quad (4e')$$

$$R_j^{dur,dp} = \begin{cases} \max\left\{\max\{R_i^{dur,ar} + t_{i^{\mathrm{dr}}k}, R_i^{strt} + e_k\} + \sigma_k + t_{kj^{\mathrm{dr}}} + \Delta, \mathscr{R}_i^{dur,dp}\right\}, & \text{if } \mathscr{R}_i^{strt} > \alpha^{\mathrm{dr}} \\ \max\left\{\max\{R_i^{dur,ar} + t_{i^{\mathrm{dr}}k}, R_i^{strt} + e_k\} + \sigma_k + t_{kj^{\mathrm{dr}}}, \mathscr{R}_i^{dur,dp} + \Delta\right\}, & \text{if } \mathscr{R}_i^{strt} < \alpha^{\mathrm{dr}} \end{cases} \tag{4f'}$$

Example 3. *(continued from Example 2) The new timeline in Figure 2 depicts the situation when synchronizing truck and drone according to REFs (4e') and (4f').*

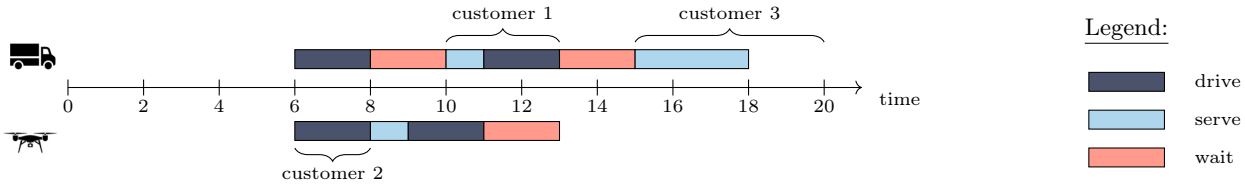

Figure 2: Synchronization of truck and drone resulting in a correct duration.

It shows that customer 1's service now starts immediately when the time window opens for service. This reduces the truck's waiting time by three time units (compared to the second timeline in Figure 1). In addition, the earlier arrival of the truck at customer 3 also reduces the time that the drone waits for the truck. The new duration of the partial route r when leaving vertex 3 is $R_3^{dur,dp} = 12$.

4.3. Dominance

A *dominance procedure* can be used to eliminate all provable redundant labels during the labeling process. Following the ideas of Roberti and Ruthmair (2021), we present three dominance rules that depend on the truck-and-drone positions. Whenever truck and drone are at the same position, a label $\mathcal{L}_1$ dominates a label $\mathcal{L}_2$ at the same vertex $i = (i^{\mathrm{tr}}, i^{\mathrm{dr}}) \in \mathcal{N}$ with $i^{\mathrm{tr}} = i^{\mathrm{dr}}$ if the following dominance rule hold true:

Dominance Rule 1. *$i^{tr} = i^{dr}, R_1^{rdc} \leq R_2^{rdc}, R_1^{load} \leq R_2^{load}, R_1^{tme,ar} \leq R_2^{tme,ar}, R_1^{dur,ar} \leq R_2^{dur,ar}, R_1^{dur,dp} \leq R_2^{dur,dp}, R_1^{strt} \leq R_2^{strt}$, and $R_1^{cust,n} \leq R_2^{cust,n}$ for all $n \in N$.*

Dominance Rule 1 follows the *standard dominance rules*, i.e., a simple pairwise $\leq$-comparison, since all REFs considered are non-decreasing (Irnich, 2008). Since truck and drone are at the same position, all attributes in $\mathscr{R}_1$ and $\mathscr{R}_2$ are zero according to REFs (2) and (4). Consequently, the standard dominance rules are always fulfilled for this set of attributes.

Whenever truck and drone are separated, i.e., $i^{\mathrm{tr}} \neq i^{\mathrm{dr}}$, stronger dominance criteria can be applied for all attributes that have an equivalent attribute in $\mathscr{R}_i$. A label $\mathcal{L}_1$ dominates a label $\mathcal{L}_2$ at the same vertex $i = (i^{\mathrm{tr}}, i^{\mathrm{dr}}) \in \mathcal{N}$ with $i^{\mathrm{tr}} \neq i^{\mathrm{dr}}$ if the following dominance rules hold true:

Dominance Rule 2. *$i^{tr} \neq i^{dr}, R_1^{rdc} \leq R_2^{rdc}, R_1^{load} \leq R_2^{load}, R_1^{tme,ar} \leq R_2^{tme,ar}, R_1^{tme,ar} + \mathscr{R}_1^{tme,ar} \leq R_2^{tme,ar} + \mathscr{R}_2^{tme,ar}, R_1^{dur,ar} \leq R_2^{dur,ar}, R_1^{dur,ar} + \mathscr{R}_1^{dur,ar} \leq R_2^{dur,ar} + \mathscr{R}_2^{dur,ar}, R_1^{dur,dp} \leq R_2^{dur,dp}, R_1^{dur,dp} + \mathscr{R}_1^{dur,dp} \leq R_2^{dur,dp} + \mathscr{R}_2^{dur,dp}, R_1^{strt} \leq R_2^{strt}, R_1^{strt} + \mathscr{R}_1^{strt} \leq R_2^{strt} + \mathscr{R}_2^{strt}$ and $R_1^{cust,n} \leq R_2^{cust,n}$ for all $n \in N$.*

Dominance Rule 3. *$i^{tr} \neq i^{dr}, R_1^{rdc} \leq R_2^{rdc}, R_1^{load} \leq R_2^{load}, R_1^{cust,n} \leq R_2^{cust,n}$ for all $n \in N$,*

$$
\begin{aligned}
R_1^{tme,ar} &\geq R_2^{tme,ar}, \\
R_1^{tme,ar} + \mathscr{R}_1^{tme,ar} &\leq R_2^{tme,ar} + \mathscr{R}_2^{tme,ar}, \\
R_1^{tme,ar} + \max_{k \in N^{dr}, w \in N \cup \{0'\}: k \neq w} \{l_k + \sigma_k + t_{kw}^{dr}\} &\leq R_2^{tme,ar} + \mathscr{R}_2^{tme,ar}, \\
R_1^{dur,ar} &\geq R_2^{dur,ar}, \\
R_1^{dur,ar} + \mathscr{R}_1^{dur,ar} &\leq R_2^{dur,ar} + \mathscr{R}_2^{dur,ar}, \\
R_1^{dur,ar} + \max\{\mathscr{R}_1^{dur,ar} + t_{i^{dr}k^*}^{dr}, \mathscr{R}_1^{strt} + e_{k^*}\} + \sigma_{k^*} + t_{k^*w^*}^{dr} &\leq R_2^{dur,ar} + \mathscr{R}_2^{dur,ar}, \\
R_1^{dur,dp} &\geq R_2^{dur,dp}, \\
R_1^{dur,dp} + \mathscr{R}_1^{dur,dp} &\leq R_2^{dur,dp} + \mathscr{R}_2^{dur,dp} \\
R_1^{dur,dp} + \max\{\mathscr{R}_1^{dur,dp} + t_{i^{dr}k^*}^{dr}, \mathscr{R}_1^{strt} + e_{k^*}\} + \sigma_{k^*} + t_{k^*w^*}^{dr} &\leq R_2^{dur,dp} + \mathscr{R}_2^{dur,ar}, \\
R_1^{strt} &\geq R_2^{strt}, \\
R_1^{strt} + \mathscr{R}_1^{strt} &\leq R_2^{strt} + \mathscr{R}_2^{strt}, \text{ and} \\
R_1^{strt} + \max\{R_1^{dur,ar} + t_{k^*w^*}^{dr} - l_{k^*}, R_1^{strt}\} &\leq R_2^{strt} + \mathscr{R}_2^{strt}
\end{aligned}
$$

with $(k^*, w^*) = \arg\max_{k \in N^{dr}, w \in N \cup \{0'\}: k \neq w} \{l_k + \sigma_k + t_{kw}^{dr}\}$.

For example, in Dominance Rule 2, the equation $R_1^{tme,ar} + \mathscr{R}_1^{tme,ar} \leq R_2^{tme,ar} + \mathscr{R}_2^{tme,ar}$ is only valid if $R_1^{tme,ar} \leq R_2^{tme,ar}$ and $\mathscr{R}_1^{tme,ar} \leq \mathscr{R}_2^{tme,ar}$ are fulfilled at the same time. Dominance Rule 3 works in a similar way: When $R_1^{tme,ar} \geq R_2^{tme,ar}$ holds true, the equation $R_1^{tme,ar} + \mathscr{R}_1^{tme,ar} \leq R_2^{tme,ar} + \mathscr{R}_2^{tme,ar}$ can only be true if $\mathscr{R}_1^{tme,ar} \leq \mathscr{R}_2^{tme,ar}$. Additionally, the maximum term guarantees that the resource consumption of the first path is never larger than that of the second path, when truck and drone meets each other at a later position $j = (j^{\text{tr}}, j^{\text{dr}})$ with $j^{\text{tr}} = j^{\text{dr}}$ in the path.

4.4. Acceleration Techniques

In this section we sketch techniques to accelerate the proposed BPC algorithm.

Lower Bound on the Number of Trucks. We compute a lower bound on the number of trucks needed to serve all customers based on their demand. Therefore, we solve a bin-packing problem with a bin size equal to the capacity of the truck Q and item weights equal to the demand q_v of each customer $v \in N$. This bin-packing problem is modeled as an arc-flow formulation (Valério de Carvalho, 1999). Using a lower bound on the number of trucks prevents the branch-and-bound tree from solving infeasible nodes, while branching on the number vehicles used.

ng-Route Relaxation and Dynamic Neighborhood Extension. Instead of solving an *elementary* SPPRC algorithm, it is common to relax its elementary requirement and only solve a SPPRC algorithm. On the one hand, this leads to an easier/faster solution of the pricing problem, but on the other hand, it comes at the cost of a weaker linear relaxation of the corresponding MP. A prominent method to control this trade-off is the *ng*-route relaxation, originally invented by Baldacci *et al.* (2011). It introduces a parametrized neighborhood $N_v \subset N$ for all $v \in N$ to effectively avoid non-elementary paths. A non-elementary cycle $(v, v_1, \ldots, v_m, v)$ over vertex v with $m \geq 1$ is only feasible if $v \notin N_{v_l}$ for some $l \in \{1, \ldots, m\}$. The quality of the lower bound and the efficiency of the pricing problem now depends on the size of N_v, i.e., larger neighborhoods lead to tighter bounds, while smaller neighborhoods lead to a faster labeling algorithm. To adapt the *ng*-route relaxation to the artificial network $\mathcal{N}$ we assign each artificial vertex $(i^{\text{tr}}, i^{\text{dr}}) \in \mathcal{N}$ with the neighborhood of the truck vertex $N_{i^{\text{tr}}}$. In addition, we strengthen the *ng*-route relaxation with *dynamic neighborhood extensions* (DNE, Roberti and Mingozzi, 2014; Bode and Irnich, 2015) in the same fashion as proposed in Schmidt *et al.* (2023). The REFs (2h), (3h), and (4h) are replaced by $R_j^{cust,n} = 0$ if $n \notin N_j$.

Heuristic Pricing. The pricing subproblem does not necessarily have to be solved to optimality, as long as negative reduced-cost columns can be generated. Therefore, we use a hierarchy of pricing heuristics (Gamache *et al.*, 1999) that rely on reduced networks to accelerate the labeling procedure. The first(second) heuristic only includes 6(10) outgoing truck arcs ($\mathcal{A}^{alone} \cup \mathcal{A}^{tog}$) and at most 3(5) drone arcs ($\mathcal{A}^{drone}$). Only if none of these heuristics has generated a reduced-cost column, the pricing problem is solved exactly. We choose the arcs according to the lowest reduced cost in each pricing iteration.

MIP-based heuristic. To provide an early upper bound, we solve a MIP-based heuristic at the first and second level of the branch-and-bound tree. Therefore, we solve the RMP with all columns generated up to this point as an integer model. Whenever the BPC terminates without finding an upper bound within the given time limit, the final RMP is solved as a MIP-based heuristic again.

Bidirectional Labeling. Developing a bidirectional labeling algorithm on the asymmetric network is a non-trivial task. Nevertheless, the work of Blufstein *et al.* (2024) and Schmidt *et al.* (2023) have shown that a bidirectional labeling algorithm on this network is possible for TSP-D and VRP-D variants. Pre-tests have shown, that it is also possible to develop a bidirectional labeling for the VRP-DTW, but the number of generated labels in the backward labeling increases rapidly compared to the forward labeling. To prevent the backward labeling from this *combinatorial explosion*, even stronger dominance rules or other techniques are necessary. Despite the general success of bidirectional labeling (Righini and Salani, 2006; Tilk *et al.*, 2017), we do not employ it here because it unfortunately does not pay off.

4.5. Valid Inequalities

In our BPC algorithm, we implement *capacity cuts* (CC, Baldacci *et al.*, 2008) to strengthen the linear relaxation of the RMP. Let $C \subseteq N$ be any subset of customers and $K(C)$ a lower bound on the number of trucks needed to serve all customers in C according to their demand. The valid inequality reads

$$\sum_{r \in \Omega_C} \lambda_r \geq K(C), \tag{7}$$

where $\Omega_C = \{r \in \Omega : r \cap C \neq \varnothing\}$ represents the subset of routes that contain at least one customer from subset C. Instead of solving a bin packing problem (as done in Section 4.4), we compute the lower bound $K(C) = \lceil \sum_{v \in C} q_v / Q \rceil$.

Since this variant of CC is non-robust, they affect the structure of the pricing problem, i.e., we have to adjust the pricing problem to handle the dual prices of each inequality. The adjustments are as follows: Let $C \in \mathcal{C}$ denote all active CCs and $\gamma_C > 0$ their corresponding positive dual prices. Further, we define an additional binary resource $(R^{cc,C})_{C \in \mathcal{C}}$, initially set to zero. When propagating along an arc $a = [i, j, k] \in \mathcal{A}$, we update the resource according to the arc type used:

$$R_j^{cc,C} = \begin{cases} 1, & \text{if } a = [i, j, \perp] \in \mathcal{A}^{tog} \text{ and } j^{\text{tr}} \in C \\ 1, & \text{if } a = [i, j, \perp] \in \mathcal{A}^{alone} \text{ and } j^{\text{tr}} \in C \\ 1, & \text{if } a = [i, j, k] \in \mathcal{A}^{drone} \text{ and } k \in C \\ R_i^{cc,C}, & \text{otherwise.} \end{cases} \tag{8}$$

The dual price γ_C has to be subtracted from the reduced cost, whenever a partial path serves a customer in C at the first time. Therefore, we adjust the REFs in (2a), (3a), and (4a) as follows:

$$R_j^{rdc} = \begin{cases} R_i^{rdc} - \pi_{j^{\text{tr}}} - \sum\limits_{C \in \mathcal{C}: R_i^{cc,C} = 0, j^{\text{tr}} \in C} \gamma_C & \text{if } a = [i, j, \perp] \in \mathcal{A}^{tog} \quad \text{(2a')} \\ R_i^{rdc} - \pi_{j^{\text{tr}}} - \sum\limits_{C \in \mathcal{C}: R_i^{cc,C} = 0, j^{\text{tr}} \in C} \gamma_C & \text{if } a = [i, j, \perp] \in \mathcal{A}^{alone} \quad \text{(3a')} \\ R_i^{rdc} - \pi_k - \sum\limits_{C \in \mathcal{C}: R_i^{cc,C} = 0, k \in C} \gamma_C & \text{if } a = [i, j, k] \in \mathcal{A}^{drone} \quad \text{(4a')} \end{cases}$$

The dominance Rules (1)-(3) are modified according to Baldacci *et al.* (2008). To identify violated inequalities (7), we use the MIP-based separation procedure by Martinelli *et al.* (2013).

4.6. Branching

To obtain an integer solution of formulation (1), we use a four-stage branching scheme. Let λ_r^* be a fractional solution of the RMP defined over $\bar{\Omega}$. At the first stage, we branch on the number of trucks in use, whenever $K^* = \sum_{r\in\Omega} \lambda_r^*$ is fractional. Therefore, we set the bounds in (1c) to $L = \lfloor K^* \rfloor$ and $K = \lceil K^* \rceil$. Second, we branch on the number of times each customer is served by a drone (Roberti and Ruthmair, 2021). Let y_{kr} be the number of times a customer k is served by route r. Whenever $y_k^* = \sum_{r\in\Omega} y_{kr}\lambda_r^*$ is fractional for a customer $k \in N^{dr}$, we create two branches $y_{k^*} = 0$ (i.e., customer k^* has to be served by the truck) and $y_{k^*} = 1$ (i.e., customer k^* has to be served by a drone) for customer k^*. We choose k^* according to a value $y_{k^*}^* - \lfloor y_{k^*}^* \rfloor$ closest to 0.5.
At the third stage, we consider the truck flow between two customers $v, w \in N$ with $v \neq w$. Let f_{ar}^{tr} be the flow value on a route r along a truck arc $a \in \mathcal{A}_{vw}^{tr} = \{\{[i,j,\perp] \in \mathcal{A}^{tog} : i^{\text{tr}} = i^{\text{dr}} = v, j^{\text{tr}} = j^{\text{dr}} = w\} \cup \{[i,j,\perp] \in \mathcal{A}^{alone} : i^{\text{tr}} = v, j^{\text{tr}} = w, i^{\text{dr}} = j^{\text{dr}} \in V\}\}$. Whenever, $f_{ar}^{tr*} = \sum_{r\in\Omega}\sum_{a\in\mathcal{A}_{vw}^{tr}} f_{ar}^{tr}\lambda_r^*$ is fractional for a customer pair v and w, we choose v^*, w^* such that the value $f_{v^*w^*}^{tr*} - \lfloor f_{v^*w^*}^{tr*} \rfloor$ is closest to 0.5. We create the two branches $\sum_{r\in\Omega}\sum_{a\in\mathcal{A}_{v^*w^*}^{tr}} f_{ar}^{tr}\lambda_r = 0$, i.e., arc $(v^*, w^*) \in A$ must not be traversed by a truck and $\sum_{r\in\Omega}\sum_{a\in\mathcal{A}_{v^*w^*}^{tr}} f_{ar}^{tr}\lambda_r = 1$, i.e., arc $(v^*, w^*) \in A$ must be traversed by a truck.

Finally, we branch on the flow of a drone subpath $\langle v, k, w\rangle$, similar to the branching strategy on stage three. Therefore, let f_{ar}^{dr} be the flow value on a route r along a drone subpath $a \in \mathcal{A}_{\langle v,k,w\rangle}^{dr} = \{[(w,v),(w,w),k] \in \mathcal{A}^{drone}\}$. Again, whenever $f_{ar}^{\text{dr}*} = \sum_{r\in\Omega}\sum_{a\in\mathcal{A}_{\langle v,k,w\rangle}^{\text{dr}}} f_{ar}^{\text{dr}}\lambda_r^*$ is fractional for a drone subpath $\langle v, k, w\rangle$ with $v \neq k \neq w, v \neq w$ we choose the subpath $\langle v^*, k^*, w^*\rangle$ according to a value $f_{\langle v^*,k^*,w^*\rangle}^{\text{dr}} - \lfloor f_{\langle v^*,k^*,w^*\rangle}^{\text{dr}} \rfloor$ closest to 0.5. The two branches are created as follows: $\sum_{r\in\Omega}\sum_{a\in\mathcal{A}_{\langle v^*,k^*,w^*\rangle}^{dr}} f_{ar}^{\text{dr}}\lambda_r = 0$, i.e., the subpath $\langle v^*, k^*, w^*\rangle$ must not be traversed by a drone and $\sum_{r\in\Omega}\sum_{a\in\mathcal{A}_{\langle v^*,k^*,w^*\rangle}^{dr}} f_{ar}^{\text{dr}}\lambda_r = 1$, i.e., the subpath $\langle v^*, k^*, w^*\rangle$ must be traversed by a drone. Unique truck and drone paths are implied by the branching decisions on stages three and four. Since both types of decisions fully determine a solution, the presented branching scheme is complete.

We remark that all branching decisions on stages two to four can be implemented directly on the artificial network by removing a subset of arcs and/or vertices. Thus, it is not necessary to add the constraints to the RMP and handle their dual prices in the pricing problem. We use a *best-first* strategy to explore the branch-and-bound tree with the primal goal to improve the dual bound.

5. Computational Results

Our BPC algorithm was coded in `C++` and compiled into a 64-bit single-thread executable with GCC 11.2.0 in release mode. CPLEX 20.1.0 with default parameters (except for allowing only a single thread) is used to (re)optimize the RMPs, solve the primal MIP-based heuristic, solve the bin-packing problem, and separate CCs. The computation time is limited to 7200 seconds per instance plus an additional time of 60 seconds to solve the MIP-based heuristic after time out. The computations were carried out on the high-performance computing cluster `MOGON II` of Johannes Gutenberg University Mainz. All used nodes are equipped with Intel Xeon E5-2630v4 (Broadwell) processors running at 2.2 GHz. Thus, its performance (on a single node) is slightly worse compared to a standard desktop processor.

5.1. Instances

Since there is no commonly used instance set publicly available, we generate a set of VRP-DTW instances to test our BPC algorithm. We create a set of 20 instances with $15, 20, 25$, and 30 customers, respectively, together with a single depot. Each customer location is placed at a randomly distributed location on a 50×50 grid. The depot is always centrally located at $(25, 25)$ and houses a homogeneous fleet of trucks, each with a capacity of $Q = 100$. The fleet size K is equal to the number of customers considered in

the instance, so the fleet is assumed to be unconstrained. A drone can carry goods of at most $Q^{\text{dr}} = 25$. Each customer $v \in N$ has a demand q_v randomly drawn from $q_v \in \{10; 15; 20; 25; 30\}$. The service time is set to 10 for all customers, i.e., $\sigma_v = 10$ for all $v \in N$. We assume that there is no service at the depot, i.e., $\sigma_0 = \sigma_{0'} = 0$. We assume a planning horizon of 540 units, so we set the time window at the depots accordingly: $[e_0, l_0] = [e_{0'}, l_{0'}] = [0, 540]$. For each customer, we randomly draw a time window from the following options: $[0, 240], [240, 480]$, or a randomly generated time window with a width of 180 within the interval of the planning horizon. As is common practice in the literature on truck-and-drone routing, we assume that drone travel times are lower than truck travel times. Given two customers v and w with their corresponding positions on the grid (x_v, y_v) and (x_w, y_w), we compute the travel time for trucks as Manhattan distance rounded down to one decimal, i.e., $t_{vw} = \lfloor 10\,(|x_v - x_w| + |y_v - y_w|)\rfloor$. The travel time for a drone is computed as Euclidean distance divided by a factor β representing the speed of the drone (also rounded down to one decimal), i.e, $t^{\text{dr}}_{vw} = \lfloor 10\,(\sqrt{(x_v - x_w)^2 + (y_v - y_w)^2}/\beta)\rfloor$. We initially set $\beta = 3$, i.e., a drone is 3 times faster than a truck. We add a small offset ϵ to all travel times such that the triangle inequality holds.

5.2. Evaluation of Algorithmic Components

We now present algorithmic results for our BPC algorithm. At first, we show the results on a pure branch-and-price (BaP) algorithm, i.e. without considering CC. In a second step, we successively add CCs and strength the *ng*-route relaxation with DNE, to show their contribution to the solution process. Table 2 shows the results for the BaP algorithm grouped by the number of customers considered ($|N|$). For the *ng*-route relaxation, we consider a neighborhood N_v that contains vertex v itself and the next five closest customers that can possibly reached within their time windows. The columns of the table have the following meaning:

#Opt: the number of instances (out of 20) solved to proven optimality within the given time limit;
Gap$_r$: the average gap between the optimal solution (*opt*) and the lower bound at the root node (LB_r) in percent, i.e., $100 \cdot (opt - LB_r)/LB_r$;
Gap: the average remaining gap between the upper bound (UB) and lower bound (LB) after time out in percent, i.e., $100 \cdot (UB - LB)/LB$;
#B&B: the average number of solved branch-and-bound nodes; and
Time: the average solution time (in seconds).

Since the algorithm computed an UB and LB for all 80 instances, the Gap considers all instances. The Gap$_r$ does only consider instances with a known optimal solution *opt*.

$\|N\|$	#Opt	Gap$_r$	Gap	#B&B	Time
15	20	1.97	0.00	27.5	81.0
20	19	1.40	<0.01	101.0	1,869.6
25	12	1.16	0.36	164.6	4,810.7
30	1	0.32	1.14	61.3	6,954.9
Total	52	1.54	0.36	88.6	3,429.1

Table 2: Computational results for the branch-and-price algorithm.

The BaP algorithm can solve 51 out of 60 instances with up to 25 customers to proven optimality. For instances with 30 customers, only one instance can be solved to proven optimality. Since the average solution times and the number of solved branch-and-bound nodes are quite high, we implement CC to strengthen the linear relaxation at the root node (see Section 4.5). Additionally, we reduce the size of all neighborhoods N_v to 3 customers (including the vertex v itself and its both closest neighbors) and extend the neighborhoods dynamically (see Section 4.4). The dynamic extension is stopped when all routes are elementary or 20 task cycles have been eliminated.

Table 3 presents the results of both BPC algorithms, including CC and DNE, again grouped by the number of customers considered ($|N|$). The columns have the same meaning as in Table 2. The additional

column Gap_{r+c} represents the average percentage gap at the root node after adding CC and at most 20 iterations of dynamic neighborhood extensions. Analogous to Gap_r, it is only calculated for instances that were solved to proven optimality.

	BPC with CC						BPC with CC and DNE					
$\|N\|$	#Opt	Gap_r	Gap_{r+c}	Gap	#B&B	Time	#Opt	Gap_r	Gap_{r+c}	Gap	#B&B	Time
15	20	1.97	0.89	0.00	6.8	114.3	20	2.85	0.89	0.00	7.8	84.8
20	19	1.40	0.69	0.03	21.5	1,969.0	20	1.75	0.69	0.00	25.2	1,630.0
25	12	1.30	0.55	0.38	43.1	4,621.8	18	1.66	0.81	0.21	88.5	3,386.8
30	2	0.59	0.13	1.10	14.0	6,776.4	4	1.13	0.63	1.18	58.5	6,618.6
Total	53	1.56	0.71	0.36	21.3	3,370.4	62	2.04	0.79	0.34	45.0	2,930.0

Table 3: Computational results for the branch-price-and-cut algorithm with CC and DNE.

Table 3 shows that the addition of CC reduces the number of solved branch-and-bound nodes by around 75% compared to the BaP algorithm. Also, the average gap at the root node (after adding cuts) can be reduced by approximately 50%. Unfortunately, CC increase the computation times for small instances with 15 and 20 customers, but they reduce the computation times as soon as more than 15 customers are considered in an instance. Nevertheless, only one additional instance with 30 customers can be solved to proven optimality in comparison to the BaP algorithm. Only the addition of DNE can finally decrease the computation times by approximately 15% over all instances. Additionally, nine more instances can be solved so that all small-sized instances with 15 and 20 customers are now solved to proven optimality. For all other instance sizes, there remains a relatively small gap of 0.2% and 1.2%, respectively.

In several VRPTW variants, the non-robust *subset-row inequalities* (SRI, Jepsen *et al.*, 2008) turned out to be a successful method to further strengthen the linear relaxation. Pre-tests have shown that adding SRIs to our VRP-DTW-specific BPC algorithm makes the labeling algorithm significantly more difficult and leads to a considerable increase in computation time. Therefore, we do not consider SRIs.

5.3. Managerial Insights

We now present three analyses on (i) the impact of combined truck-and-drone routing compared to classical truck routing, (ii) the impact of different drones that only differ in their flying range, and (iii) the comparison of an as-late-as-possible schedule with an as-early-as-possible schedule.

Impact of combined truck-and-drone routing. We analyze the impact of combined truck-and-drone routing in comparison to classical truck routing. By removing all truck alone and drone arcs from the artificial network, the BPC algorithm solves a VRPTW with the objective to minimize the summarized route duration. Of course, this problem is computationally much easier to solve than the VRP-DTW so that all 80 instances were solved to proven optimality within approximately 160 seconds of computation time on average over all customer sizes. Since we are only interested in an upper bound (or optimal solution) to the objective function, we do not present any further computational results for the VRPTW solutions.

Figure 3 shows the average summarized route duration for classical truck routing (VRPTW) compared to truck-and-drone routing with different types of drones, meaning that each type of drone has a different speed relative to the truck. If trucks and drones have the same speed ($\beta = 1$), the average summarized route duration can be reduced by approximately 30% compared to classical truck routing consistent over all instance sizes. Whenever the drone is faster than the truck ($\beta = 3$), the summarized duration can be further decreased by approximately 15 percentage points. Increasing the drone speed to $\beta = 5$ or above does result in considerably additional time savings, consistent with the findings in Schmidt *et al.* (2023).

Analysis of different drone flying ranges. In the following, we assume a drone speed of $\beta = 3$ and analyze different flying ranges δ for the drone. To generate such flying ranges, we follow the method used in Schmidt *et al.* (2023): We introduce a new parameter $\gamma \in [0, 100]$ and limit the instance-specific flying range so that

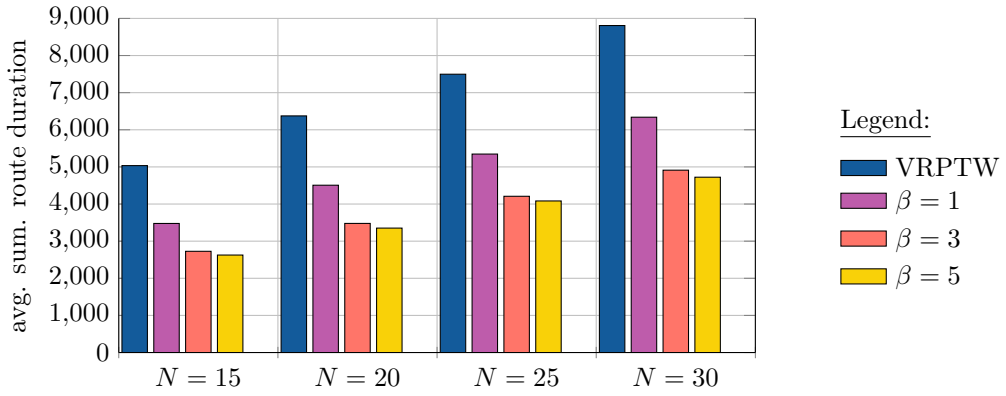

Figure 3: Comparison between VRPTW and VRP-DTW with different drone speeds.

the largest value $t^{\mathrm{dr}}_{i^{\mathrm{dr}}k} + t^{\mathrm{dr}}_{k,j^{\mathrm{dr}}} \leq \delta$ is fulfilled for no more than $\gamma \cdot |\mathcal{A}^{drone}|/100$ drone arcs. Thus, all other $100 - \gamma$ percent of the drone arcs can be removed from the network.

Figure 4 visualizes the effects of using different types of drones with different flying ranges.

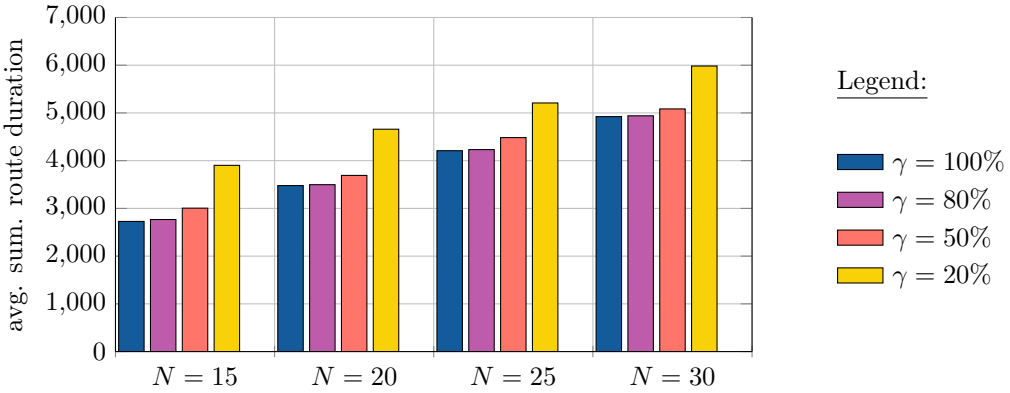

Figure 4: Comparison of different drone flying ranges.

Obviously, the more customers can theoretically reached by a drone, the lower is the summarized duration over all routes. However, this figure shows that it is not necessary to use a drone with an unlimited flying range of $\gamma = 100\%$. The time savings compared to a drone that can only reach 80% of all customers is negligible. Using drones with a flying range that theoretically reaches half of all customers results in an average increase of the summarized delivery time by approximately 6% compared to an unlimited drone. Only if the drone's flying range is further limited to reach a maximum of only 20% of all customers, the summarized delivery time increases by approximately 30% on average.

Minimizing route duration vs. minimizing completion time. With only slight adjustments, our BPC algorithm can also solve a VRP-DTW instance with the objective to minimize the summarized completion time. This allows us to analyze the impact of both objective functions (minimizing the summarized route duration and minimizing the summarized completion time) on the total time needed to serve all customers. Table 4 presents the average summarized completion time (*Compl. time*), the average summarized duration (*Duration*) and the average percentage savings in time (*Savings (%)*) that can be realized with an as-late-as-possible schedule compared to an as-early-as-possible schedule. All values are grouped by the number of customers considered in the instance ($|N|$). It shows that the average time needed to serve all customers can be reduced by approximately 64% on average over all instances if both vehicles can leave the depot at

$\|N\|$	Compl. time	Duration	Savings (%)
15	7,819	2,727	65.12
20	9,194	3,478	62.17
25	11,694	4,209	64.00
30	14,161	4,923	65.24
Avg.	10,717	3,834	64.13

Table 4: Comparison of route duration and completion time.

any time instead of starting immediately at e_0. At first sight, these values seem high, but waiting times are often long, especially on routes that serve customers with late time windows.

6. Conclusions and Outlook

In the paper at hand, we studied the VRP-DTW that generalizes the VRP-D by the existence of time windows. Further we presented the first exact BPC algorithm to solve the VRP-DTW with the objective to minimize the summarized duration over all routes. The presentation mainly focused on the modeling and the effective solution of the column-generation pricing problem, the SPPRC. Further, non-robust capacity cuts and dynamic neighborhood extensions were implemented to strengthen the linear relaxation of the set-partitioning formulation. A computational study showed that only the addition of capacity cuts is not enough to decrease the computation times compared to a branch-and-price algorithm. Only by adding dynamic neighborhood extensions, the solution times were considerably reduced and 9 more instances were solved to proven optimality. Managerial insights showed that combined truck-and-drone routing can reduce the average delivery times by up to 45% (depending on the speed of a drone) compared to classical truck routing. We also showed that it is not necessary to use drones with an "unlimited" flying range. The average delivery time increases only by 6% when using a drone that can reach at most 50% of all customers. Finally, we showed that an as-late-as-possible schedule can considerably reduce the delivery times compared to an as-early-as-possible schedule.

Our version of the VRP-DTW is based on the assumption that a drone always waits on the ground instead of hovering, which implies that there is no energy consumption that affects its flying range while waiting. Of course, this assumption can be changed by assuming that a drone always hovers while waiting. In this case, the proposed BPC algorithm is no longer applicable. When propagating a path through the artificial network, the latest departure time for both vehicles may change, resulting in (additional) waiting times at a customer node. The labeling algorithm in its current form cannot ensure that existing drone flights along a truck-and-drone path remain feasible in such a situation. New techniques, similar to those that ensure *ride time constraints* in a *Dial-A-Ride Problem*, need to be developed to keep track of the feasibility of a truck-and-drone path.

References

Amazon Prime Air (2023). Amazon is launching ultra-fast drone deliveries in Italy, the UK, and a third location in the U.S. https://www.aboutamazon.com/news/operations/amazon-prime-air-drone-delivery-updates.

Bakir, I. and Tiniç, G. Ö. (2020). Optimizing drone-assisted last-mile deliveries: The vehicle routing problem with flexible drones. Optimization online. https://optimization-online.org/?p=16382.

Baldacci, R., Christofides, N., and Mingozzi, A. (2008). An exact algorithm for the vehicle routing problem based on the set partitioning formulation with additional cuts. *Mathematical Programming*, **115**(2), 351–385.

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

Blufstein, M., Lera-Romero, G., and Soulignac, F. J. (2024). Decremental state-space relaxations for the basic traveling salesman problem with a drone. *INFORMS Journal on Computing*.

Bode, C. and Irnich, S. (2015). In-depth analysis of pricing problem relaxations for the capacitated arc-routing problem. *Transportation Science*, **49**(2), 369–383.

Carlsson, J. G. and Song, S. (2018). Coordinated logistics with a truck and a drone. *Management Science*, **64**(9), 4052–4069.
Chen, C., Demir, E., and Huang, Y. (2021a). An adaptive large neighborhood search heuristic for the vehicle routing problem with time windows and delivery robots. *European Journal of Operational Research*, **294**(3), 1164–1180.
Chen, C., Demir, E., Huang, Y., and Qiu, R. (2021b). The adoption of self-driving delivery robots in last mile logistics. *Transportation Research Part E: Logistics and Transportation Review*, **146**, 102214.
Chung, S. H., Sah, B., and Lee, J. (2020). Optimization for drone and drone-truck combined operations: A review of the state of the art and future directions. *Computers & Operations Research*, **123**, 105004.
Coindreau, M.-A., Gallay, O., and Zufferey, N. (2021). Parcel delivery cost minimization with time window constraints using trucks and drones. *Networks*, **78**(4), 400–420.
Das, D. N., Sewani, R., Wang, J., and Tiwari, M. K. (2021). Synchronized truck and drone routing in package delivery logistics. *IEEE Transactions on Intelligent Transportation Systems*, **22**(9), 5772–5782.
Desaulniers, G., Desrosiers, J., and Solomon, M., editors (2005). *Column Generation*. Springer, New York, NY.
Desaulniers, G., Madsen, O., and Ropke, S. (2014). The vehicle routing problem with time windows. In P. Toth and D. Vigo, editors, *Vehicle Routing*, volume 18 of *MOS-SIAM Series on Optimization*, chapter 5, pages 119–159. Society for Industrial and Applied Mathematics, Philadelphia, PA, Philadelphia, PA.
Federal Aviation Administraion (2023). The FAA authorizes UPS flight forward and uAvionix to operate drones beyond visual line of sight. https://www.faa.gov/newsroom/faa-authorizes-ups-uavionix.
Gamache, M., Soumis, F., Marquis, G., and Desrosiers, J. (1999). A column generation approach for large-scale aircrew rostering problems. *Operations Research*, **47**(2), 247–263.
Goodchild, A. and Toy, J. (2018). Delivery by drone: An evaluation of unmanned aerial vehicle technology in reducing CO_2 emissions in the delivery service industry. *Transportation Research Part D: Transport and Environment*, **61**, 58–67.
Irnich, S. (2008). Resource extension functions: Properties, inversion, and generalization to segments. *OR Spectrum*, **30**(1), 113–148.
Irnich, S. and Desaulniers, G. (2005). Shortest path problems with resource constraints. In Desaulniers *et al.* (2005), chapter 2, pages 33–65.
Jepsen, M., Petersen, B., Spoorendonk, S., and Pisinger, D. (2008). Subset-row inequalities applied to the vehicle-routing problem with time windows. *Operations Research*, **56**(2), 497–511.
Joerss, M., Schröder, J., Neuhaus, F., Klink, C., and Mann, F. (2016). Parcel delivery: The future of last mile. McKinsey&Company, Brochure.
Kuo, R., Lu, S.-H., Lai, P.-Y., and Mara, S. T. W. (2022). Vehicle routing problem with drones considering time windows. *Expert Systems with Applications*, **191**, 116264.
Li, H. and Wang, F. (2022). Branch-price-and-cut for the truck–drone routing problem with time windows. *Naval Research Logistics (NRL)*, **70**(2), 184–204.
Li, H., Wang, H., Chen, J., and Bai, M. (2020). Two-echelon vehicle routing problem with time windows and mobile satellites. *Transportation Research Part B: Methodological*, **138**, 179–201.
Macrina, G., Di Puglia Pugliese, L., Guerriero, F., and Laporte, G. (2020). Drone-aided routing: A literature review. *Transportation Research Part C: Emerging Technologies*, **120**, 102762.
Madani, B. and Ndiaye, M. (2022). Hybrid truck-drone delivery systems: A systematic literature review. *IEEE Access*, **10**, 92854–92878.
Martinelli, R., Poggi, M., and Subramanian, A. (2013). Improved bounds for large scale capacitated arc routing problem. *Computers & Operations Research*, **40**(8), 2145–2160.
Moshref-Javadi, M. and Winkenbach, M. (2021). Applications and research avenues for drone-based models in logistics: A classification and review. *Expert Systems with Applications*, **177**, 114854.
Otto, A., Agatz, N., Campbell, J., Golden, B., and Pesch, E. (2018). Optimization approaches for civil applications of unmanned aerial vehicles (UAVs) or aerial drones: A survey. *Networks*, **72**(4), 411–458.
Poikonen, S., Wang, X., and Golden, B. (2017). The vehicle routing problem with drones: Extended models and connections. *Networks*, **70**(1), 34–43.
Pugliese, L. D. P. and Guerriero, F. (2017). Last-mile deliveries by using drones and classical vehicles. In *Springer Proceedings in Mathematics & Statistics*, pages 557–565. Springer International Publishing.
Pugliese, L. D. P., Macrina, G., and Guerriero, F. (2020). Trucks and drones cooperation in the last-mile delivery process. *Networks*, **78**(4), 371–399.
Pugliese, L. D. P., Guerriero, F., and Scutellá, M. G. (2021). The last-mile delivery process with trucks and drones under uncertain energy consumption. *Journal of Optimization Theory and Applications*, **191**(1), 31–67.
Righini, G. and Salani, M. (2006). Symmetry helps: Bounded bi-directional dynamic programming for the elementary shortest path problem with resource constraints. *Discrete Optimization*, **3**(3), 255–273.
Roberti, R. and Mingozzi, A. (2014). Dynamic ng-path relaxation for the delivery man problem. *Transportation Science*, **48**(3), 413–424.
Roberti, R. and Ruthmair, M. (2021). Exact methods for the traveling salesman problem with drone. *Transportation Science*, **55**(2), 315–335.
Savelsbergh, M. W. P. (1992). The vehicle routing problem with time windows: Minimizing route duration. *ORSA Journal on Computing*, **4**(2), 146–154.
Schmidt, J., Tilk, C., and Irnich, S. (2023). Exact solution of the vehicle routing problem with drones. Technical Report LM-2023-03, Chair of Logistics Management, Gutenberg School of Management and Economics, Johannes Gutenberg University Mainz, Mainz, Germany.
Solomon, M. M. (1987). Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations*

Research, **35**(2), 254–265.

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

Tilk, C., Rothenbächer, A.-K., Gschwind, T., and Irnich, S. (2017). Asymmetry matters: Dynamic half-way points in bidirectional labeling for solving shortest path problems with resource constraints faster. *European Journal of Operational Research*, **261**(2), 530–539.

Valério de Carvalho, J. M. (1999). Exact solution of bin-packing problems using column generation and branch-and-bound. *Annals of Operations Research*, **86**, 629–659.

Wang, X., Poikonen, S., and Golden, B. (2017). The vehicle routing problem with drones: several worst-case results. *Optimization Letters*, **11**(4), 679–697.

Yin, Y., Li, D., Wang, D., Ignatius, J., Cheng, T., and Wang, S. (2023). A branch-and-price-and-cut algorithm for the truck-based drone delivery routing problem with time windows. *European Journal of Operational Research*, **309**(3), 1125–1144.

Zhen, L., Gao, J., Tan, Z., Wang, S., and Baldacci, R. (2023). Branch-price-and-cut for trucks and drones cooperative delivery. *IISE Transactions*, **55**(3), 271–287.

Zhou, H., Qin, H., Cheng, C., and Rousseau, L.-M. (2023). An exact algorithm for the two-echelon vehicle routing problem with drones. *Transportation Research Part B: Methodological*, **168**, 124–150.