Transportation Research Part C 106 (2019) 166–204

Contents lists available at ScienceDirect

Transportation Research Part C

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

A matheuristic for the vehicle routing problem with drones and its variants

Daniel Schermer*, Mahdi Moeini, Oliver Wendt

Chair of Business Information Systems and Operations Research (BISOR), Technische Universität Kaiserslautern, 67663 Kaiserslautern, Germany

ARTICLE INFO

Keywords:
Vehicle Routing Problem
Drones
Logistics
Last-mile delivery
Valid inequalities
Matheuristics
Heuristics
Large-scale instances

ABSTRACT

In this work, we are interested in studying the Vehicle Routing Problem with Drones (VRPD). Given a fleet of trucks, where each truck carries a given number of drones, the objective consists in designing feasible routes and drone operations such that all customers are served and minimal makespan is achieved. We formulate the VRPD as a Mixed Integer Linear Program (MILP), which can be solved by any standard MILP solver. Moreover, with the aim of improving the performance of solvers, we introduce several sets of valid inequalities (VIEQ). Due to limited performance of the solvers in addressing large instances, we propose a matheuristic approach that effectively exploits the problem structure of the VRPD. Integral to this approach, we propose the Drone Assignment and Scheduling Problem (DASP) that, given an existing routing of trucks, looks for an optimal assignment and schedule of drones such that the makespan is minimized. In this context, we propose two MILP formulations for the DASP. In order to evaluate the performance of a state-of-the-art solver in tackling the MILP formulation of the VRPD, the benefit of the proposed VIEQs, and the performance of the matheuristic, we carried out extensive computational experiments. According to the numerical results, the use of drones can significantly reduce the makespan and the proposed VIEQ as well as the matheuristic approach have a significant contribution in solving the VRPD effectively.

1. Introduction

In recent years, drones have started to play an increasing role in logistic systems in both, academic research and practical context. Several companies in the logistic industry, such as Amazon Inc., Deutsche Post AG, UPS Inc. and the DPDgroup are actively investigating the potentials of drones for parcel delivery (see Murray and Chu, 2015, and references therein). Furthermore, drones have been successfully applied in many public and private sectors including energy, agriculture and forestry, environmental protection, and emergency response (see Otto et al., 2018, and references therein).

Most recently, several problems have been proposed in the academic literature that integrate drones into last-mile delivery, in an effort to lower costs and reduce delivery times (see, e.g., Murray and Chu, 2015; Wang et al., 2017; Agatz et al., 2018). Drones might generally move faster between two locations than trucks due to not being restricted to the road network or congestion. In addition, drones and their payload are far more lightweight than trucks, which causes drones to consume less energy for the movement between two points. However, a drone's capacity is inherently limited to just one or few parcels. Moreover, as drones rely on comparatively small batteries for powering their flight, their range of operation is quite restricted compared to a commercial delivery

* Corresponding author.
E-mail addresses: daniel.schermer@wiwi.uni-kl.de (D. Schermer), mahdi.moeini@wiwi.uni-kl.de (M. Moeini), wendt@wiwi.uni-kl.de (O. Wendt).

https://doi.org/10.1016/j.trc.2019.06.016
Received 24 January 2019; Received in revised form 17 May 2019; Accepted 22 June 2019
Available online 23 July 2019
0968-090X/

truck that is powered by a fossil fuel or a high-capacity battery. Consequently, due to the complementary nature of trucks and drones, the integration of the latter into existing architectures in last-mile delivery might yield synergetic benefits.

With the objective of taking into account the potential benefits of using drones, the *Vehicle Routing Problem with Drones* (VRPD) was proposed by Wang et al. (2017). They propose the integration of drones into the classical *Vehicle Routing Problem* (VRP). More precisely, given a fleet of trucks that are equipped with sets of drones, the objective consists in designing feasible routes and drone operations such that all customers are served and minimal makespan is achieved.

In this paper, we address the VRPD. The contributions of our work are manyfold:

1. To the best of our knowledge, we are the first to provide a formal specification of the *multi-drone* VRPD through a *Mixed Integer Linear Program* (MILP). In particular, as opposed to existing works that consider a multi-truck case (see, e.g., Sacramento et al., 2019; Schermer et al., 2019), our formulation allows for *cyclic* drone operations. Furthermore, we show that our model is very flexible and easily adaptable when deviating assumptions apply.
2. In order to improve the performance of MILP solvers in solving VRPD instances, we derive several sets of valid inequalities. We show that our proposed valid inequalities have a significant impact on the solver's performance in finding optimal solutions of small instances.
3. Nevertheless, using the MILP formulation to obtain (optimal) solutions in a reasonable amount of time is only possible for small instances. Therefore, in order to address larger instances of the VRPD, we propose a matheuristic that effectively exploits the problem structure of the VRPD. As an integral component of our matheuristic, we propose the *Drone Assignment and Scheduling Problem* (DASP) as a MILP. Indeed, given an existing routing of trucks, the DASP looks for the optimal assignment and schedule of a set of drones, such that the makespan is minimized.
4. In any case, through our numerical study, we show that the application of drones can significantly reduce the makespan, thus making them a valuable prospect in last-mile delivery. Furthermore, we provide an in-depth sensitivity analysis on relevant drone parameters.

The remainder of this paper is organized as follows. We begin in Section 2 by providing a brief overview of the existing academic literature that is related to the application of drones in last-mile delivery. Afterwards, Section 3 introduces the notation, mathematical model, and the proposed valid inequalities for the VRPD. In Section 4, we explain the basic structure of the matheuristic and show how it effectively exploits the problem structure of the VRPD. Within this section, we propose the DASP as a mathematical program through two MILP formulations. Detailed results on the numerical studies are presented in Section 5. Finally, concluding remarks on this work and future research implications are provided in Section 6.

2. Related literature

In this section, we provide a short overview on drone-related optimization problems in last-mile delivery. While there are many works that specifically consider routing a single drone or a swarm of drones (see, e.g., Dorling et al., 2017), for the purpose of this work, we limit ourselves to problems that require intensive *synchronization* (Drexl, 2012) between trucks and drones. A more general literature review on civil applications of drones is given by Otto et al. (2018).

Murray and Chu (2015) introduced two novel problems, that are visualized in Fig. 1, where a truck is working in tandem with a drone. Both problems assume that a depot and a set of customers, each of whom must be served exactly once by either the truck or drone, are given:

- In the *Flying Sidekick Traveling Salesman Problem* (FSTSP), the drone travels alongside the truck. Both start from a common depot and must return to the same depot at the end of the tour. At any customer location (or at the depot), the drone may be launched from the truck, starting an *operation* (or sortie), which is described as follows: the drone begins its flight, performs a delivery to a customer, and will then be retrieved by the truck at a later customer location (or the depot).
- In the *Parallel Drone Scheduling TSP* (PDSTSP), it is assumed that the depot is in close proximity to most customers. In this case, it might be beneficial to let the drone make its deliveries independently from the truck. Whereas the truck serves several customers

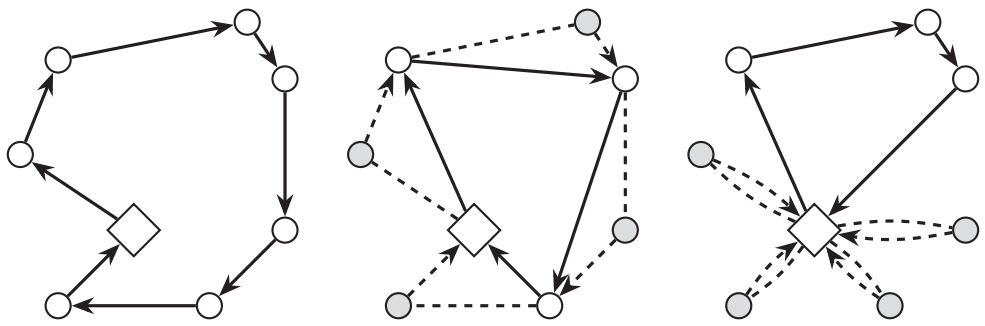


Fig. 1. An illustration of TSP, FSTSP, and PDSTSP solutions where the truck's and drone's paths are indicated by solid and dashed lines, respectively.

that are located further away from the depot, the drone will continuously return to the depot to pick up a new parcel after each delivery.

Both problems, i.e., the FSTSP and PDSTSP, share the objective of minimizing the makespan. However, in the case of the FSTSP extended synchronization between the truck and the drone is required.

Murray and Chu (2015) introduced MILP formulations as well as two heuristic methods for solving the FSTSP and PDSTSP. The authors conclude that, due to the $\mathcal{NP}$-hard nature of the proposed problems, only small-sized instances can be solved to optimality within a reasonable amount of time. Based on their experiments, they note that the FSTSP is significantly harder to solve due to the synchronization requirements. Finally, given a problem instance, Murray and Chu (2015) highlight that it is difficult to assess a priori whether treating a problem as the FSTSP or the PDSTSP will yield better results with regards to the objective value.

The *Traveling Salesman Problem with Drone* (TSP-D) by Agatz et al. (2018) was proposed independently from the FSTSP but shares most of the common assumptions nonetheless. In this problem, a key difference to the FSTSP is that a drone can be retrieved at the same location from which it was launched by the truck and that drone operations are constrained by flight distance rather than time. The authors propose an *Integer Programming* (IP) formulation as well as heuristics for solving the TSP-D. Agatz et al. performed a numerical study to assess the performance of their heuristics on various problem instances. The authors suggest to extend their model in order to take into account multiple trucks (or drones) as well as to add different possibilities for recharging the drones.

Based on dynamic programming, Bouman et al. (2018) provide an exact solution method for the TSP-D. The authors highlight that their method is able to find the optimal solution for problem instances with 10 vertices much faster compared to the IP approach presented by Agatz et al. (2018). In particular, Bouman et al. (2018) also perform an in-depth analysis on the impact of (heuristically) adding additional constraints that restrict the number of drone operations. Based on their experiments, this can significantly reduce the time required to find the optimal solution, albeit, a non-optimal solution might be achieved after the imposition of these restrictions. They note that it might be beneficial to identify a procedure that detects unprofitable operations upfront. Furthermore, they highlight that their method cannot be readily applied to cases where multiple drones are present.

Es Yurek and Ozmutlu (2018) provide an approach for solving the TSP-D that is based on a decomposition of the problem into two components:

- First, all feasible tours to the *Traveling Salesman Problem* (TSP) are generated that visit only a subset of customers. Each tour provides a lower bound on a (feasible) TSP-D solution that features the same TSP solution and serves the remaining customers by drones.
- Starting with the shortest tour, mathematical programming is used to find the optimal drone operations such that the customers that were not contained in the TSP tour are served by drone. Here, the objective is to minimize the waiting time of the truck such that the objective value after solving the mathematical model is as close as possible to the lower bound that is associated with the initial TSP tour.

This procedure continues until the best remaining lower bound (of a tour that has not been examined) is worse than the incumbent TSP-D solution. Es Yurek and Ozmutlu (2018) compare their results to Murray and Chu (2015) and Agatz et al. (2018) and highlight that they are able to solve instances with 10–12 vertices in shorter computation time. However, the authors highlight that their approach might be limited to small instances due to the enumerative nature of the approach.

Carlsson and Song (2018) worked on a variant of the TSP-D called the *Horsefly Routing Problem* w.r.t. a commercially available system with the same name. Compared to the TSP-D, their variant features a single truck that can be equipped with one or more drones. The authors propose a heuristic method based on convex optimization and use large instances (with up to 500 vertices) as well as practical data (with up to 100 vertices). One of their key findings is the hypothesis that the potential benefit of using a drone in tandem with a truck is proportional to the square root of the relative velocity between the truck and the drone.

Wang et al. (2017) introduced the VRPD as a generalization of the TSP-D. In the VRPD a fleet of trucks, each truck equipped with a given number of drones, is tasked with delivering parcels to customers. The drones may be launched from the truck at the depot or at any customer location and might be retrieved by the truck at a different customer location (or at the depot, concluding its tour). When a drone is launched, it can serve exactly one customer before it needs to return to the truck such that it can be resupplied. In contrast to classic VRPs, the objective is to minimize the makespan, i.e., the time required to serve all customers such that the entire fleet has returned to the depot. Wang et al. studied the problem from a theoretical point of view. In particular, they introduced several upper bounds on the amount of time that can be saved by employing drones. The upper bounds are obtained by investigating the structure of optimal solutions and depend on the relative velocity of the drones compared to the trucks and the number of drones per truck. Poikonen et al. (2017) refined the work of Wang et al. (2017) by considering the impact on the previously introduced bounds when integrating limited battery life, different distance metrics, and operational expenditures of deploying drones and trucks in the objective function. The authors suggest the possibility of launching and retrieving drones from arbitrary locations instead of being restricted to customer locations. Furthermore, as a future research direction, the authors suggest to develop models, heuristics, and benchmark VRPD instances.

Di Puglia Pugliese and Guerriero (2017) introduced the *Vehicle Drone Routing Problem with Time Windows* (VDRPTW), where the objective consists in serving customers at minimum cost using a truck and drones, while respecting the time-window restrictions. The authors provide a mathematical model for the VDRPTW and use a commercial solver for solving the problem on randomly generated instances with 5 or 10 customers. The authors conclude that the benefit of drones for last-mile delivery strongly depends on the marginal cost per mile of both drones and trucks.

Boysen et al. (2018) proposed the *Drone Scheduling Problem* (DSP), where a fixed sequence of customers that are visited by a single truck and a disjoint set of customers that must be served by drone(s) are given. In this problem, they look for the optimal schedule of drones, i.e., launch and retrieval locations, such that the makespan is minimized. The authors differentiate between a truck that is equipped with a single or multiple drones and different degrees of freedom regarding feasible launch and retrieval locations. However, for cases that restrict potential launch and retrieval locations, polynomial time algorithms can be derived. They propose two different MILPs for solving the DSP and conclude that the DSP might also be used for solving a more holistic problem such as the TSP-D. In this case, a subsequence of customers visited by the truck might be generated and the drones schedule for the remaining customers derived through the DSP (similar to the work of Es Yurek and Ozmutlu (2018)).

Ha et al. (2018) considered a variant of the FSTSP where the objective is to minimize the operational costs. These costs include the variable cost of driving and flying and might also include costs for the time spent waiting by the truck and drone. In particular, the authors adapt the MILP model and heuristics proposed by Murray and Chu (2015) to their new problem. Furthermore, they propose a *Greedy Randomized Adaptive Search Procedure* (GRASP) as a solution method. The authors can show that their MILP formulation can be used for solving small-sized instances under the min-cost objective within a few minutes. Moreover, the proposed GRASP heuristic is able to achieve optimal results in all cases for 5 small instances with 10 vertices. Notably, Ha et al. (2018) also test their heuristic under the min-time objective. A detailed numerical study reveals the effectiveness of their approach. According to their experiments, under the min-cost objective, the costs are reduced by about 30% w.r.t. truck-only solutions (with an increase in the makespan by typically 50%). Moreover, under the min-time objective, the makespan is reduced by about 10% (with a decrease in cost by typically 20%).

Sacramento et al. (2019) investigated a variant of the FSTSP, in which multiple trucks are present and call it the *Vehicle Routing Problem with Drones* (VRP-D). The authors consider an objective where the operational costs are minimized under the restriction of a maximum duration for all routes. For this purpose, the MILP formulation of Murray and Chu (2015) is adapted and an *Adaptive Large Neighborhood Search* (ALNS) procedure proposed. As integral components of their ALNS algorithm, the authors suggest several problem-specific *destroy* and *repair* methods. Sacramento et al. (2019) show that their MILP formulation can be used to solve some small-sized instances with 6, 10, and 12 vertices to optimality. However, significant increases in runtime are observed with an increase in the instance size. On larger instances, they can show that the truck-drone fleet allows cost-savings of typically 20–30% compared to truck-only delivery.

The possibility of *en route* operations in the context of the VRPD has been studied in Schermer et al. (2019). En route operations were defined by Marinelli et al. (2018) as the possibility of constructing arc-based truck-drone operations. In Schermer et al. (2019), a formal specification of the VRPD with en route operations is given through a MILP. In this MILP, en route operations can be performed at some discrete points on each arc. Moreover, an algorithm that follows the framework of *Variable Neighborhood Search* (VNS) is proposed. Through a computational study, it is revealed that, in some cases, en route operations allow for additional savings, compared to the non en route case. Moreover, a strong dependence of these additional savings on drone parameters such as their velocity or endurance is shown. Albeit, it is revealed that the possibility of en route operations makes the problem significantly more difficult to solve compared to the non en route case.

Table 1 provides an overview of the problems that were introduced in this section. As we have already established, in contrast to classic VRPs (see, e.g., Toth and Vigo, 2014 and references therein), the objective in these problems often consists in minimizing the makespan rather than cost. Furthermore, if multiple trucks are allowed, there is no fixed cost incurred when an additional truck is deployed. Rather, the number of available trucks and drones per truck (fleet) is considered a parameter of the problem itself as opposed to a decision-relevant component (see Wang et al., 2017; Di Puglia Pugliese and Guerriero, 2017; Sacramento et al., 2019). Finally, problem assumptions often differ with regards to *cyclic* drone operations, i.e., the possibility of drone operations that start and end at the same vertex (while the truck remains stationary). In fact, even though all problems allow acyclic operations, cyclic operations are often not permitted.

3. Problem definition

In this paper, we are interested in studying the VRPD as proposed by Wang et al. (2017). We recall that the VRPD asks for routing

Table 1
Overview of the drone-related optimization problems with extensive synchronization requirements presented in Section 2.

Reference	Trucks	Drones	Objective	Cyclic Operations	Contribution	Vertices
Murray and Chu (2015)	1	1	Time	no	MILP, heuristic	10...20
Wang et al. (2017), Poikonen et al. (2017)	n	m	Time	Yes	Theoretical insights	–
Di Puglia Pugliese and Guerriero (2017)	n	m	Cost	no	MILP	5...10
Bouman et al. (2018), Agatz et al. (2018)	1	1	Time	Yes	IP, DP, heuristic	10...100
Carlsson and Song (2018)	1	m	Time	No	Heuristic	25...500
Es Yurek and Ozmutlu (2018)	1	1	Time	No	Heuristic	10...20
Boysen et al. (2018)	1	m	Time	Yes	MILP, SA	5...100
Ha et al. (2018)	1	1	Cost/time	No	MILP, GRASP	10...100
Sacramento et al. (2019)	n	1	Cost	No	MILP, ALNS	6...200
Schermer et al. (2019)	n	m	Time	No	MILP, VNS	10...50
This work	n	m	Time	Yes	MILP, matheuristic	10...100

a fleet of trucks, each truck equipped with a set of drones, and seeks to minimize the makespan, i.e., the time required to serve all customers using the trucks and the drones such that, by the end of the mission, all trucks and drones have returned to the depot.

In Section 3.1, we formulate the VRPD as MILP model. Afterwards, in Section 3.2, we provide additional valid inequalities. As a key difference to existing models (refer to Table 1), our model permits not only multiple trucks, but also multiple drones per truck and cyclic drone operations, i.e., operations that start and end at the same vertex.

3.1. Notation and mathematical model

Suppose that a set of customer locations, each with a uniform demand, and a fleet of homogeneous trucks, each carrying the same number of homogeneous drones, are given. In the VRPD, we look for minimizing the makespan required to serve all customers by using the given fleet such that, by the end of the mission, all trucks and drones must be at the depot (Wang et al., 2017; Poikonen et al., 2017). Furthermore, we make the following assumptions regarding the nature of drones (Murray and Chu, 2015; Wang et al., 2017; Agatz et al., 2018):

- When launched from the truck, a drone can serve exactly one customer before it needs to return to the same truck from which it was launched. Furthermore, we ask that each truck has sufficient capacity.
- We assume that a drone has a limited endurance of $\mathcal{E}$ *distance units* per operation. After returning to the truck, the battery of the drone is recharged (or swapped) instantaneously. For a more detailed discussion on this assumption, we refer to Appendix C.
- Associated with each launch and retrieval of a drone are two overhead times, $\bar{t}_l$ and $\bar{t}_r$. Furthermore, we assume that the truck is capable of launching and retrieving drones in parallel.
- Drones may only be dispatched and picked up from vertices, i.e., at the depot or any other customer location.

Inspired by the work of Di Puglia Pugliese and Guerriero (2017), we use the following notation in order to formulate the VRPD. Assume that a complete graph $\mathcal{G} = (V, E)$ is given, where V is the set of vertices (or nodes) and E is the set of edges. The set V contains n vertices associated with the customers, named $V_N = \{1, 2, ..., n-1, n\}$ and two extra vertices 0 and $n+1$ that (in order to simplify the notation and the mathematical formulation) both represent the same depot location at the start and end of each tour, respectively. Thus, $V = \{0, 1, ..., n, n+1\}$, where $0 \equiv n+1$. In addition, we call $V_D \subseteq V_N$ the *drone-serviceable* subset of customers.

We refer to $V_L = V \backslash \{n+1\}$ as the subset of vertices in V from which drones may be launched and retrieved after performing cyclic operations (that start and end at the same location). Furthermore, $V_R = V \backslash \{0\}$ denotes the subset of vertices where drones may be retrieved after performing an acyclic operation.

By the parameters d_{ij} and $\bar{d}_{ij}$ we define the distance required to travel from vertex i to vertex j by truck and drone, respectively. Additionally, as the drone may not be limited to the road network, we can assume that $\bar{d}_{ij} \leqslant d_{ij}$: $\forall\, i, j \in V$.

We consider a limited set $\mathcal{K} = \{1, ..., K_n\}$, $K_n \in \mathbb{Z}_{>0}$ of $|\mathcal{K}| = K_n$ trucks. Furthermore, each truck $k \in \mathcal{K}$ holds an equal set of $\mathcal{D} = \{1, ..., D_n\}$, $D_n \in \mathbb{Z}_{>0}$, $|\mathcal{D}| = D_n$ drones. Each drone may travel a maximum range of $\mathcal{E}$ distance units per operation, where a *drone-operation* is characterized by a triple (i, w, j) as follows: the drone is launched from a truck at a vertex $i \in V_L$, delivers a parcel to $w \in V_N$, and is retrieved by the same truck from which it was launched at vertex $j \in V$ ($i \neq w, w \neq j$). If $i = j$ (i.e., the operation is (i, w, i)), we call it a *cyclic operation*; otherwise, we call it an *acyclic operation*.

We use the following decision variables for modeling the VRPD:

x_{ij}^k, $\forall\, i,j \in V, k \in \mathcal{K}, i \neq j$: Binary variables that state whether arc (i, j) is used by truck k.

y_{iwj}^{kd}, $\forall\, i \in V_L, w \in V_D, j \in V, k \in \mathcal{K}, d \in \mathcal{D}, i \neq w, w \neq j$: Binary variables that specify whether a drone d associated with truck k performs the operation (i, w, j).

p_{ij}^k, $\forall\, i,j \in V, k \in \mathcal{K}, i \neq j$: Binary variables that indicate whether vertex i is served before but not necessarily adjacent to vertex j in the route of truck k.

u_i^k, $\forall\, i \in V, k \in \mathcal{K}$: Integer variables with a lower bound of 0 that state the position of vertex i in the route of truck k.

z_{aiwje}^{kd}, $\forall\, a,i,w,j,e \in V, k \in \mathcal{K}, d \in \mathcal{D}, |\{a,i,w,j,e\}| = 5$: Binary variables that specify whether the drone delivery y_{awe}^{kd} is performed circumjacent to x_{ij}^k, i.e., drone d is launched at vertex a before the truck k has reached i and retrieved at vertex e after the truck has reached j ($|\{a, i, w, j, e\}| = 5$ implies that all vertices are pairwise different).

a_i^k, $\forall\, i \in V, k \in \mathcal{K}$: Continuous variables with a lower bound of 0 that indicate the arrival time of truck k at i.

s_i^k, $\forall\, i \in V, k \in \mathcal{K}$: Continuous variables with a lower bound of 0 that indicate the earliest possible departure time of truck k from i.

r_i^{kd}, $\forall\, i \in V, k \in \mathcal{K}, d \in \mathcal{D}$: Continuous variables with a lower bound of 0 that indicate the earliest possible release time of drone d that belongs to truck k at i.

s_i^{kd}, $\forall\, i \in V, k \in \mathcal{K}, d \in \mathcal{D}$: Continuous variables with a lower bound of 0 that indicate the earliest time at which drone d that belongs to truck k is ready to depart from i.

y_i^{kd}, $\forall\, i \in V, k \in \mathcal{K}, d \in \mathcal{D}$: Binary variables that take value 1, if drone d that belongs to truck k is available for an operation at i.

τ : Continuous variable that indicates the makespan.

The MILP formulation of the VRPD is given in (1)–(27) where (1) is the objective function and the constraints are given by

(2)–(27). Please note, due to the large number of variables involved in the model, for the sake of compactness, all constraints that restrict the domains of binary variables to the set {0, 1}, integer variables to $\mathbb{Z}^+$, and continuous variables to take a value within their respective intervals $[0, \infty)$ are not listed explicitly.

$$\min \quad \tau \tag{1}$$

$$\text{s. t.} \qquad \tau \geqslant s_{n+1}^k: \quad \forall\, k \in \mathcal{K}, \tag{2}$$

$$\sum_{j \in V_R} x_{0j}^k - \sum_{i \in V_L} x_{i,n+1}^k = 0: \quad \forall\, k \in \mathcal{K}, \tag{3}$$

$$\sum_{\substack{i \in V_L, \\ i \neq h}} x_{ih}^k - \sum_{\substack{j \in V_R, \\ h \neq j}} x_{hj}^k = 0: \quad \forall\, k \in \mathcal{K}, h \in V_N, \tag{4}$$

$$\sum_{j \in V_R} x_{0j}^k = 1: \quad \forall\, k \in \mathcal{K}, \tag{5}$$

$$\sum_{k \in \mathcal{K}} \sum_{\substack{i \in V_L, \\ i \neq w}} x_{iw}^k + \sum_{k \in \mathcal{K}} \sum_{d \in \mathcal{D}} \sum_{\substack{i \in V_L, \\ i \neq w}} \sum_{\substack{j \in V, \\ w \neq j}} y_{iwj}^{kd} = 1: \quad \forall\, w \in V_D, \tag{6}$$

$$\sum_{k \in \mathcal{K}} \sum_{\substack{i \in V_L, \\ i \neq w}} x_{iw}^k = 1: \quad \forall\, w \in V_N \backslash V_D, \tag{7}$$

$$y_{0wj}^{kd} \leqslant \sum_{\substack{h \in V_L, \\ h \neq j}} x_{hj}^k: \quad \forall\, k \in \mathcal{K}, d \in \mathcal{D}, w \in V_D, j \in V_R, w \neq j, \tag{8}$$

$$2y_{iwj}^{kd} \leqslant \sum_{\substack{h \in V_L, \\ h \neq i}} x_{hi}^k + \sum_{\substack{l \in V_N, \\ l \neq j}} x_{lj}^k: \quad \forall\, k \in \mathcal{K}, d \in \mathcal{D}, i \in V_N, w \in V_D, j \in V_R, \left| \left\{ \begin{matrix} \\ i, w, j \\ \\ \end{matrix} \right\} \right| = 3, \tag{9}$$

$$y_{iwi}^{kd} \leqslant \sum_{\substack{h \in V_L, \\ h \neq i}} x_{hi}^k: \quad \forall\, k \in \mathcal{K}, d \in \mathcal{D}, i \in V_N, w \in V_D, i \neq w, \tag{10}$$

$$u_i^k - u_j^k + 1 \leqslant (n+1)(1 - x_{ij}^k): \quad \forall\, k \in \mathcal{K}, i \in V_N, j \in V_R, i \neq j, \tag{11}$$

$$u_i^k - u_j^k + 1 \leqslant \left(n+1\right)\left(1 - y_{iwj}^{kd}\right): \quad \forall\, k \in \mathcal{K}, d \in \mathcal{D}, i \in V_N, w \in V_D, j \in V_R, \left| \left\{ i, w, j \right\} \right| = 3, \tag{12}$$

$$x_{ij}^k \leqslant p_{ij}^k: \quad \forall\, k \in \mathcal{K}, i \in V_L, j \in V_R, i \neq j, \tag{13}$$

$$u_i^k - u_j^k \geqslant 1 - \left(n+1\right)p_{ij}^k: \quad \forall\, k \in \mathcal{K}, i \in V_N, j \in V_R, i \neq j, \tag{14}$$

$$u_i^k - u_j^k \leqslant -1 + \left(n+1\right)\left(1 - p_{ij}^k\right): \quad \forall\, k \in \mathcal{K}, i \in V_N, j \in V_R, i \neq j, \tag{15}$$

$$3z_{aiwje}^{kd} \leqslant y_{awe}^{kd} + p_{ai}^k + p_{je}^k: \forall\, k \in \mathcal{K}, d \in \mathcal{D}, a \in V_L, i, j \in V_N, w \in V_D, e \in V_R, |\{a, i, w, j, e\}| = 5, \tag{16}$$

$$2 + z_{aiwje}^{kd} \geqslant y_{awe}^{kd} + p_{ai}^k + p_{je}^k: \forall\, k \in \mathcal{K}, d \in \mathcal{D}, a \in V_L, i, j \in V_N, w \in V_D, e \in V_R, |\{a, i, w, j, e\}| = 5, \tag{17}$$

$$M(x_{ij}^k - 1) + s_i^k + t_{ij} \leqslant a_j^k \;: \forall\, k \in \mathcal{K}, i \in V_L, j \in V_R, i \neq j, \tag{18}$$

$$a_i^k \leqslant s_i^k: \quad \forall\, k \in \mathcal{K}, i \in V, \tag{19}$$

$$M\left(y_{iwj}^{kd} - 1\right) + s_i^{kd} + \left(\bar{t}_l + \bar{t}_{iw} + \bar{t}_{wj} + \bar{t}_r\right) \leqslant r_j^{kd} \;: \forall\, k \in \mathcal{K}, d \in \mathcal{D}, i \in V_L, w \in V_D, j \in V_R, |\{i, w, j\}| = 3, \tag{20}$$

$$r_i^{kd} + \sum_{\substack{w \in V_D, \\ i \neq w}} \left(\bar{t}_l + \bar{t}_{iw} + \bar{t}_{wi} + \bar{t}_r \right) y_{iwi}^{kd} \leqslant s_i^{kd}: \quad \forall\, k \in \mathcal{K},\, d \in \mathcal{D},\, i \in V_L, \tag{21}$$

$$s_i^{kd} + \bar{t}_l \sum_{\substack{w \in V_D, \\ i \neq w}} \sum_{\substack{j \in V_R, \\ |\{i,w,j\}| = 3}} y_{iwj}^{kd} \leqslant s_i^k: \quad \forall\, k \in \mathcal{K},\, d \in \mathcal{D},\, i \in V, \tag{22}$$

$$a_j^k + \bar{t}_r \sum_{\substack{i \in V_L \\ i \neq j}} \sum_{\substack{w \in V_D, \\ |\{i,w,j\}| = 3}} y_{iwj}^{kd} \leqslant r_j^{kd}: \quad \forall\, k \in \mathcal{K},\, d \in \mathcal{D},\, j \in V, \tag{23}$$

$$\left(\bar{d}_{iw} + \bar{d}_{wj} \right) y_{iwj}^{kd} \leqslant \mathcal{E}: \quad \forall\, k \in \mathcal{K},\, d \in \mathcal{D},\, i \in V_L,\, w \in V_D,\, j \in V,\, i \neq w,\, w \neq j, \tag{24}$$

$$M \left(x_{ij}^k - 1 \right) + y_i^{kd} + \sum_{\substack{a \in V_L, \\ a \neq i, \\ a \neq j}} \sum_{\substack{w \in V_D, \\ i \neq w, \\ |\{a,w,j\}| = 3}} y_{awj}^{kd} + \sum_{\substack{a \in V_L, \\ a \neq i, \\ a \neq j}} \sum_{\substack{w \in V_D, \\ |\{a,i,w,e\}| = 4}} \sum_{\substack{e \in V_R, \\ |\{a,i,w,j,e\}| = 5}} z_{aiwje}^{kd} \leqslant 1 \;:\; \forall\, k \in \mathcal{K},\, d \in \mathcal{D},\, i \in V_L,\, j \in V_R,\, i \neq j, \tag{25}$$

$$M \left(x_{ij}^k - 1 \right) + \sum_{\substack{w \in V_D, \\ i \neq w}} \sum_{\substack{e \in V_R, \\ |\{i,w,e\}| = 3}} y_{iwe}^{kd} \leqslant y_i^{kd} \;:\; \forall\, k \in \mathcal{K},\, d \in \mathcal{D},\, i \in V_L,\, j \in V_R,\, i \neq j, \tag{26}$$

$$M \left(x_{ij}^k - 1 \right) + \sum_{\substack{w \in V_D, \\ i \neq w}} y_{iwi}^{kd} \leqslant M y_i^{kd} \;:\; \forall\, k \in \mathcal{K},\, d \in \mathcal{D},\, i \in V_L,\, j \in V_R,\, i \neq j. \tag{27}$$

The objective function, represented by (1), minimizes the makespan through constraints (2). Constraints (3)–(5) form the preservation of flow for the trucks. More precisely, constraints (3) guarantee that each truck that departs from the depot must also return to the depot. Constraints (4) ensure that each truck that visits a customer $h \in V_N$ must also depart from the same customer. Due to constraints (5), each truck $k \in \mathcal{K}$ must start from the depot exactly once (note that a truck might remain stationary by following the arc $x_{0,n+1}^k$).

Constraints (6) ensure task synchronization, i.e., each customer $j \in V_D$ must be served exactly once by either a truck or a drone. Furthermore, constraints (7) guarantee that the non drone-eligible customers are served by a truck only. Movement synchronization between routes of the trucks and its drones is handled through constraints (8)–(10). Constraints (8) handle the synchronization between the truck and drone in the case that a drone is launched from the depot. In the case of constraint (9), if a drone performs an acyclic operation (i, w, j), then truck $k \in \mathcal{K}$ must visit vertices $i \in V_n$ and $j \in V_R$ at some point to allow a launch and retrieval of the drone at vertices i and j, respectively. Constraints (10) ensure that cyclic operations are only possible, if the truck $k \in \mathcal{K}$ visits i.

Constraints (11) and (12) link the variables u_i^k, for truck $k \in \mathcal{K}$, to the decision variables x_{ij}^k and y_{iwj}^k. When i is adjacent to j, constraints (13) provide a strong coupling between x_{ij}^k and p_{ij}^k. For the remaining cases, constraints (14) and (15) associate the variables p_{ij}^k with u_i^k and u_j^k. We ask that $u_0 = 0$ and $p_{0j} = 1 \;\; \forall\, j \in V_R$. Constraints (16) and (17) determine a value on the variables z_{aiwje}^k depending on the decision variables p_{ij}^k and y_{iwj}^k.

The set of constraints (18)–(23) guarantee temporal operational synchronization by providing lower bounds on the arrival and departure times for the trucks and drones. For each possible transition of a truck and acyclic operation of a drone, constraints (18) and (20) set lower bounds on the arrival times and release times for the trucks and drones, respectively. Furthermore, constraints (19) enforce the arrival time as a lower bound on the departure time of the truck. In case of cyclic operations, constraints (21) set lower bounds on the earliest departure time of the drone. More precisely, a drone can depart from a vertex as soon its cyclic operations have been completed. In contrast, the truck must wait for the last drone to complete its cyclic operation which is handled through constraints (22). Finally, constraints (23) guarantee that a drone can only be released after the truck has arrived. Constraints (22) and

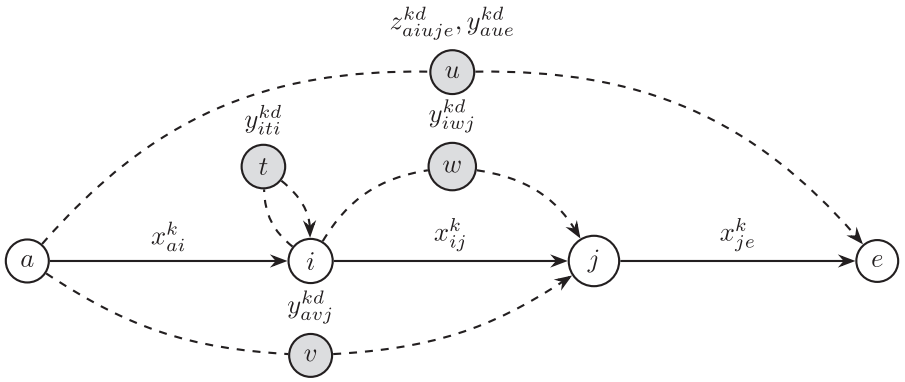


Fig. 2. A visualization of some possible operations that might occur as a truck proceeds from i to j. In particular, if drone d performs a circumjacent operation such as, e.g., either y^{kd}_{avj} or y^{kd}_{aue}, no operation can be performed by drone d at i. Otherwise, the drone might perform a number of cyclic operations such as y^{kd}_{iti} followed by a single acyclic operation such as y^{kd}_{iwj}.

(23) also incorporate the overhead times $\bar{t}_l$ (and $\bar{t}_r$) in the case that a drone is launched for (retrieved after) an acyclic operation.

Constraints (24) impose a maximum distance (endurance) constraint on each drone operation (i, w, j). Finally, constraints (25)–(27) guarantee that during each transition of the truck from vertex i to vertex j a feasible number of drone operations are performed (see Fig. 2). More precisely, constraints (25) force the variables y^d_i to take value 0, if a circumjacent operation is performed. Furthermore, the set of constraints (26) and (27) ensure that acyclic and cyclic operations can only be performed if the drone is available at i.

Any MILP solver, e.g., IBM CPLEX or Gurobi Optimizer, might be used to solve MILP (1)–(27). However, a glance at MILP (1)–(27) reveals that size of the model grows significantly with an increase in the size of $\mathcal{G}$ as well as the sets $\mathcal{K}$ and $\mathcal{D}$. In particular, tracking circumjacent drone operations requires a significant modelling effort. Furthermore, compared to our previous work in (Schermer et al., 2018a,b, 2019), cyclic drone operations lead to several additional variables and constraints. A detailed discussion on the flexibility of the model to deviating assumptions can be found in A.

3.2. Valid inequalities

In order to enhance the performance of the MILP solvers in solving the VRPD, we derive several *Valid Inequalities* (VIEQ) as cuts in the solution space of the problem that exclude none of the optimal solutions. Hence, the addition of VIEQ might have a positive impact on the computational time that is required for solving the VRPD.

3.2.1. Makespan bounds

In this section, we derive two sets of valid inequalities that are used to place *lower bounds* (LB) on the objective value τ. While the derivation of these VIEQ is relatively straightforward, they have a significant impact on the performance of the solver by providing an improved linear relaxation to the solver (refer to Section 5).

Proposition 1. *In the MILP (1)–(27),τ is bounded by the time spent traveling by trucks.*

Proof. For each truck $k \in \mathcal{K}$, the time spent traveling can be calculated as follows:

$$\sum_{i \in V_L} \sum_{\substack{j \in V_R, \\ i \neq j}} t_{ij} x^k_{ij} : \forall\, k \in \mathcal{K}. \tag{28}$$

Furthermore, the time spent waiting at vertices can be accounted as follows:

$$\sum_{i \in V} \left(s^k_i - a^k_i \right) : \forall\, k \in \mathcal{K}. \tag{29}$$

The variable τ marks the time at which all trucks and all drones have arrived at vertex $n + 1$. In a feasible VRPD solution, a truck spends time traveling on arcs and might spend further time waiting on vertices for a drone to be retrieved. Therefore, the time spent traveling and waiting by each truck qualifies as a valid lower bound on the objective value as follows:

$$\sum_{i \in V} \left(s^k_i - a^k_i \right) + \sum_{i \in V_L} \sum_{\substack{j \in V_R, \\ i \neq j}} t_{ij} x^k_{ij} \leqslant \tau : \forall\, k \in \mathcal{K}. \tag{30}$$

This completes the proof of Proposition 1. □

Proposition 2. *In the MILP (1)–(27),τ is bounded by the time spent traveling by each drone.*

Proof. For each drone $d \in \mathcal{D}$ that belongs to a truck $k \in \mathcal{K}$, the total time spent travelling can be accounted as follows:

$$\sum_{i \in V_L} \sum_{\substack{w \in V_D, \\ i \neq w}} \left(\bar{t}_l + \bar{t}_{iw} + \bar{t}_{wi} + \bar{t}_r \right) y_{iwi}^{kd} + \sum_{i \in V_L} \sum_{\substack{w \in V_D, \\ i \neq w}} \sum_{\substack{j \in V_R, \\ i \neq j, \\ w \neq j}} \left(\bar{t}_l + \bar{t}_{iw} + \bar{t}_{wi} + \bar{t}_r \right) y_{iwj}^{kd} : \forall\, k \in \mathcal{K},\, d \in \mathcal{D}. \tag{31}$$

The variable τ marks the time at which all trucks and all drones have arrived at vertex $n + 1$. In a feasible VRPD solution, a drone spends time traveling on arcs and might spend further time waiting at vertices after performing an acyclic operation for the truck to arrive, such that it can be retrieved. Thus, the time spent traveling by each drone qualifies as a valid lower bound on the objective value as follows:

$$\sum_{i \in V_L} \sum_{\substack{w \in V_D, \\ i \neq w}} \left(\bar{t}_l + \bar{t}_{iw} + \bar{t}_{wi} + \bar{t}_r \right) y_{iwi}^{kd} + \sum_{i \in V_L} \sum_{\substack{w \in V_D, \\ i \neq w}} \sum_{\substack{j \in V_R, \\ i \neq j, \\ w \neq j}} \left(\bar{t}_l + \bar{t}_{iw} + \bar{t}_{wi} + \bar{t}_r \right) y_{iwj}^{kd} \leqslant \tau : \forall\, k \in \mathcal{K},\, d \in \mathcal{D}. \tag{32}$$

Therefore, we have proven Proposition 2. □

3.2.2. Symmetry in the VRPD

In this section, we show that in the case of a symmetric graph $\mathcal{G}$, the VRPD is a symmetric problem, hence, has symmetric solutions. We use this property in order to derive a set of *symmetry-breaking cuts*, and for this purpose, we introduce the following proposition.

Proposition 3. *Assume that we have a solution for the MILP (1)–(27) with components x_{ij}^k and y_{iwj}^k. Furthermore, in the given solution, let π^k define the sequence of customers as visited by the truck k, for each truck $k \in \mathcal{K}$. Then, the following statements are true:*

- (i) *The VRPD is a symmetric problem, i.e., corresponding to π^k, there is another solution, having the same objective value, consisting of reverse order of visiting customers as it is in π^k.*
- (ii) *The following inequality breaks the symmetry in the VRPD*

$$\sum_{i \in V_N} i \cdot x_{0,i}^k \leqslant \sum_{j \in V_N} j \cdot x_{j,n+1}^k : \forall\, k \in \mathcal{K}. \tag{33}$$

Proof.

(i) In the arc-based formulation MILP (1)–(27), for each truck $k \in \mathcal{K}$, a feasible selection of decision variables x_{ij}^k and y_{iwj}^k imply makespan s_{n+1}^k and a directed graph $\mathcal{G}(V, A)$. We call $\mathcal{G}^T(V, A^T)$ the transpose graph of $\mathcal{G}$. We prove that the same makespan s_{n+1}^k is associated with $\mathcal{G}$ and the transpose graph $\mathcal{G}^T$. We consider the following three possible cases:
1. If no drones are present, $\mathcal{G}$ is a simple path. In this case, the longest path through $\mathcal{G}$ (which corresponds to s_{n+1}^k) is equal to the inverse longest path through the transpose graph $\mathcal{G}^T$.
2. Next, assume that acyclic drone operations occur. In this case, $\mathcal{G}$ is no longer a simple path but remains a directed acyclic graph nonetheless. Thus, the longest path through the $\mathcal{G}$ (which corresponds to s_{n+1}^k) remains equals the inverse longest path through $\mathcal{G}^T$.
3. Finally, consider the general cases where acyclic and cyclic drone operations occur. In this case, based on the structure of the problem, any directed graph G with cyclic operations can be transformed into directed acyclic graph through the introduction of a dummy vertex for each cyclic operation (see Fig. 3). Therefore, this case can be reduced to previous case.

(ii) For each truck $k \in \mathcal{K}$, the decision variables x_{ij}^k directly correspond to a sequence of customers π^k, as visited by the truck k. Based on the symmetry of the problem, we might introduce a constraint that guarantees that a tour $\pi^k = \{0, i..., j, n + 1\}$ (here, $x_{0,i}^k = x_{j,n+1}^k = 1$) and its partially inverted sequence $\pi^{Tk} = \{0, j..., i, n + 1\}$ are not both evaluated. To this end, we introduce the following set of *static symmetry breaking inequalities* (Jünger et al., 2010) that guarantee that for any sequence $\pi^k = \{0, i, ..., j, n + 1\}$, $i \leqslant j$, the partially inverted sequence $\pi^{Tk} = \{0, j, ..., i, n + 1\}$ must not be evaluated.

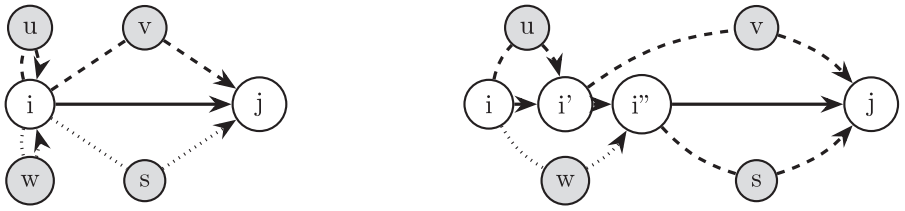


Fig. 3. A segment of a VRPD route as a directed graph with two drones indicated by dashed and dotted lines, respectively, and the corresponding directed acyclic graph. In the corresponding graph, we assume that the weight of solid edges that connect dummy vertices is equal to 0. In this case, the order of the dummy vertices implies that the drone that serves u returns before the drone that serves w.

$$\sum_{i \in V_N} i \cdot x^k_{0,i} \leqslant \sum_{j \in V_N} j \cdot x^k_{j,n+1} \colon \forall\, k \in \mathcal{K}. \tag{34}$$

This completes the proof of Proposition 3. □

3.2.3. Knapsack inequalities

In this section, we provide two sets of knapsack inequalities.

Proposition 4. *Assume that $i \in V_L, j \in V_R$, and $w \in V_D$, where $|\{i, w, j\}| = 3$; then, the following inequalities are valid for the MILP (1)–(27):*

$$x^k_{iw} + \sum_{d \in \mathcal{D}} \left(y^{kd}_{iwi} + \sum_{\substack{j \in V_R, \\ i \neq j, \\ w \neq j}} y^{kd}_{iwj} \right) \leqslant 1 \colon \forall\, k \in \mathcal{K},\, i \in V_L,\, w \in V_D,\, i \neq w, \tag{35}$$

$$x^k_{iw} + x^k_{wj} + x^k_{ij} + \sum_{d \in \mathcal{D}} \left(y^{kd}_{iwj} + y^{kd}_{iwi} + y^{kd}_{jwj} \right) \leqslant 2 \colon\ \forall\, k \in \mathcal{K},\, i \in V_L,\, w \in V_D,\, j \in V_R,\, \left| \left\{ i, w, j \right\} \right| = 3. \tag{36}$$

Proof. Consider a pair of two vertices $i \in V_L$ and $w \in V_D$ where $i \neq w$. Constraints (6) guarantee that the indegree of performed deliveries is at most once. Therefore, w can be served from i at most once by either a truck, a drone that performs a cyclic delivery, or a drone that performs an acyclic delivery. Mathematically, this can be expressed through the following inequalities:

$$x^k_{iw} + \sum_{d \in \mathcal{D}} \left(y^{kd}_{iwi} + \sum_{\substack{j \in V_R, \\ i \neq j, \\ w \neq j}} y^{kd}_{iwj} \right) \leqslant 1 \colon \forall\, k \in \mathcal{K},\, i \in V_L,\, w \in V_D,\, i \neq w. \tag{37}$$

Next, consider a triple of vertices $i \in V_L$, $w \in V_N$, and $j \in V_R$, such that all vertices are different ($|\{i, w, j\}| = 3$). In this triple, with the truck (and possibly drones) located at i, there can be at most two deliveries such that all demands are fulfilled. Starting at i, either the truck first serves w and then j, or the truck directly moves from i to j. In the latter case, a drone is permitted to either perform a cyclic or acyclic operation. Therefore, we can introduce the following inequalities:

$$x^k_{iw} + x^k_{wj} + x^k_{ij} + \sum_{d \in \mathcal{D}} \left(y^{kd}_{iwj} + y^{kd}_{iwi} + y^{kd}_{jwj} \right) \leqslant 2 \colon \forall\, k \in \mathcal{K},\, i \in V_L,\, w \in V_D,\, j \in V_R,\, \left| \left\{ i, w, j \right\} \right| = 3. \tag{38}$$

The introduction of inequalities (37) and (38) completes the proof of Proposition (4). □

4. Matheuristic

As we will see in Section 5.1, a state-of-the-art MILP solver can solve only small VRPD instances within reasonable runtime. As an alternative, we might use a heuristic solution method which must be able to provide high-quality solutions within short computation time. In this context, a heuristic might approach the VRPD by solving the following hierarchical subproblems in succession:

1. *Allocation Problem*: Which customers should be served by which tandem (where each tandem consists of a truck and its drone

fleet)?
2. *Sequencing Problem*: In which order should each allocation be processed?
3. *Assignment Problem*: In each sequence, which customers should be assigned to the truck and drones?
4. *Scheduling Problem*: Given the assignment, which feasible operations should the drones perform, i.e., what are the concrete locations were drones are launched and retrieved?

Methods that embed mathematical programming inside a heuristic framework are called *matheuristics* and have been applied to several different routing problems (see, e.g., Archetti and Speranza, 2014 and references therein). In this work, we propose a matheuristic in which we decompose the problem into two natural components as follows:

1. *Allocation and sequencing*, that can be solved simultaneously through established heuristics for VRPs (see, e.g., (Toth and Vigo, 2014) and references therein). In the following, we refer to this subproblem as the *routing* subproblem of the VRPD.
2. *Drone assignment and scheduling*, that are solved simultaneously through the use of mathematical programming techniques. In the following, we refer to this subproblem as the *Drone Assignment and Scheduling Problem* (DASP). In contrast to existing works (e.g., Boysen et al., 2018; Es Yurek and Ozmutlu, 2018), the assignment of drones to customers is not fixed in advance or determined during routing, i.e., we solve the assignment and scheduling problem simultaneously. Furthermore, our method is suitable for cases where multiple drones are present that might have different relative velocities or endurances.

Algorithm 1 shows the basic structure of our proposed matheuristic which will be explained in more detail in the following sections. Initially, the incumbent objective value $f(x^*)$ is set to a sufficiently large value M. Then, our algorithm iterates until a termination criteria (e.g., a runtime limit) is met. In this algorithm, lines 3–8 refer to the allocation and sequencing problem which will be addressed in Section 4.1. Afterwards, in Section 4.2, we dedicate ourselves to lines 10–17 that are concerned with the drone assignment and scheduling problem.

Algorithm 1. Matheuristic

```
1:  f(x*) ← M
2:  while termination critera is not met do
3:      Routes ← RandomizedSavingsHeuristic
4:      Routes ← LocalSearch(Routes)                                  } Allocation and Sequencing (Section 4.1)
5:      Routes ← sort(Routes)
6:      if tabuList.contains(π : Routes) then
7:          continue while
8:      end if
9:      RUB ← 0
10:     for (π : Routes) do
11:         x_π ← DASP(π, BestObjStop(RUB), Cutoff(f(x*)))
12:         RUB ← (f(x_π) > RUB) ? f(x_π) : RUB                       } Drone Assignment and Scheduling (Section 4.2)
13:         if RUB ≥ f(x*) then
14:             tabuList.add(π)
15:             continue while
16:         end if
17:     end for
18:     f(x*) ← RUB
19:     tabuList.add(π that determined RUB)
20: end while
```

4.1. Allocation and sequencing

In each iteration, a set of routes is generated and sorted according to their length. As we use an exact procedure to determine the optimal drone assignment and schedule for a given route (refer to Section 4.2), a tabu list is employed. The tabu list stores examined routes and inhibits the matheuristic from repeatedly solving the same DASP for routes that are guaranteed not to improve the incumbent solution. The criteria, under that a route is added to the list, will be described in more detail in Section 4.2.

In principle, any algorithm that generates a feasible routing (a set of routes) might be used as an initialization procedure within our heuristic. Moreover, it is conceivable to embed the DASP as a local search method within a more sophisticated metaheuristic approach. However, to show the strength of the DASP as a component, we choose a more simplistic procedure for solving the routing problem. More precisely, we adapt the parallel Clarke-Wright-Savings heuristic as initialization procedure (Clarke and Wright, 1964). For customizing this heuristic for the VRPD, we make two minor adjustments:

- Since the trucks do not possess a limited capacity in the VRPD, we limit each route to contain at most $\lceil |\mathcal{G}|/|\mathcal{K}| \rceil$ customers.
- In order to generate different starting solutions, within the heuristic the savings are weighted with a *random* value drawn at uniform from an interval $[r_{\text{lb}}, 1]$, where $0 < r_{\text{lb}} \leqslant 1$.

Afterwards, the initial routing is improved through some local search operations. Here, for a limited number of iterations, we attempt to improve the routing by minimizing the length of the longest remaining route. As local search method, we use the *2-opt* as single-route improvement heuristic (Lin and Kernighan, 1973). Furthermore, we rely on *String Relocation* (SR) and *String Exchange* (SE) with *best insertion*, respectively, as multi-route improvement heuristics (Toth and Vigo, 2014). All operators work by choosing *random* vertices within the routes. In the case of 2-opt, two vertices from the same route are selected and the sequence is inverted. If this inversion leads to an improved route length, the move is accepted, otherwise it is reverted. This move can be efficiently evaluated in $O(1)$ per iteration. In the case of SR and SE, first, two random routes are selected from the set of routes. Clearly, these moves are only applied when the number of routes (trucks) is larger than one. In the case of SR, a random vertex is selected from the longer route and inserted into the shorter route at the best location in the sequence, i.e., the one that will add the minimal length to the route after insertion. In the case of SE, a random vertex is selected from each route and inserted into the respective other route at the best location in the sequence. Both moves, SR and SE, can be efficiently evaluated in $O(n)$.

4.2. Drone assignment and scheduling

Once a feasible routing (a set of routes) has been generated, the next steps involve solving the DASP for each route. More precisely, we use a MILP solver to determine the optimal assignment and schedule of drones for each truck. The proposed MILP formulations will be discussed in Sections 4.2.1 and 4.2.2. Let the *route upper bound* ($\mathcal{RUB}$) be the supremum on the makespan based on all routes that have been evaluated thus far in the current iteration. The $\mathcal{RUB}$ is reset to zero in every iteration. Then, for each route π in the list of routes, we solve the DASP as a MILP with Gurobi Optimizer 8.1.0 (Gurobi Optimization, 2019), yielding a solution x_π, by supplying the solver with the following parameters (line 11 in Algorithm 1):

- *BestObjStop*($\mathcal{RUB}$): When the solver detects a feasible solution x_π with an objective value $f(x_\pi) \leqslant \mathcal{RUB}$, we can terminate the solver's internal branch-and-cut process. In this case, the objective value associated with the DASP for the current route π is at least as good as the $\mathcal{RUB}$. Therefore, there is no need to solve the current DASP to optimality as the $\mathcal{RUB}$ remains unchanged. In particular, in presence of multiple trucks, once an initial value on the $\mathcal{RUB} \neq 0$ has been established, by solving the DASP for the first truck's route, subsequent DASPs can often be evaluated more quickly through this process.
- *Cutoff*($f(x^*)$): When the solver detects that the best possible solution value is greater than or equal to the incumbent objective value $f(x^*)$, we can cutoff (terminate) the solver's branch-and-cut process. In this case, the objective value $f(x_\pi)$ would generate a new $\mathcal{RUB}$ that is guaranteed to be greater than or equal to the current incumbent objective value $f(x^*)$. Therefore, the makespan associated with the current set of routes is guaranteed not to improve on the incumbent solution.

Fig. 4 visualizes how this procedure can significantly speed up the process, requiring not all MIP gaps to be tight when solving the DASP. Whenever the MILP returns a solution x_π with an objective $f(x_\pi) > \mathcal{RUB}$, we set the $\mathcal{RUB}$ to the value of $f(x_\pi)$ (line 12 in Algorithm 1). Furthermore, if the current $\mathcal{RUB}$ is greater than or equal to $f(x^*)$, the current iteration can terminate, i.e., no further routes have to be evaluated as no improvement on the incumbent solution can be found with the current set of routes. In this case, the route π that determined the $\mathcal{RUB}$ (and, in case of a symmetric graph, the reverse route as a result from Section 3.2.2) are stored on the tabu list and we continue with the next iteration by generating a completely new set of routes (lines 13–16 in Algorithm 1). Finally, if the algorithm evaluated the last route and if $\mathcal{RUB} < f(x^*)$ still holds, then we have a new and improved incumbent solution x (that consists of all x_π for each route π) with an objective value $f(x) = \mathcal{RUB}$. This solution is then stored as the incumbent solution and the route that determined the $\mathcal{RUB}$ is also added to the tabu list (refer to Section 4.1).

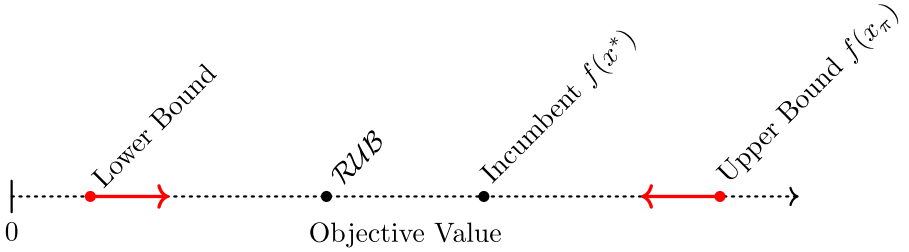


Fig. 4. For each route π, internally, the MILP solver attempts to find an optimal solution x_π to the DASP by closing the gap between the lower and upper bound (indicated by red arrows). If the lower bound is greater than or equal to the incumbent solution $f(x^*)$, a cutoff can be applied. In this case, the next $\mathcal{RUB}$ is guaranteed to be greater than or equal to $f(x^*)$ and, thus, the incumbent solution will not change. Similarly, if the upper bound $f(x_\pi) \leqslant \mathcal{RUB}$, we can stop and do not need to solve the DASP to optimality. In this case, as the $\mathcal{RUB}$ determines the makespan of the VRPD solution, it is sufficient to prove that a solution x_π exists for the current route π such that $f(x_\pi) \leqslant \mathcal{RUB}$.

At this point, we require a MILP that, given a feasible routing, returns the optimal assignment of customers to either the truck or a drone including the schedule of drones for each route π such that the makespan is minimized. In principle, MILP (1)–(27) can be readily used for this, by fixing the variables x_{ij}^k based on each route π. However, in terms of columns and rows, the resulting MILP is still very large and, given a fixed allocation and sequencing decision, contains many redundant constraints.

In what follows, we will show that a much more compact MILP formulation for the DASP can be derived. To this end, assume that a graph $\mathcal{G}(V, E)$ and a route (permutation) $\pi = (0, ..., n+1)$, $\pi \subseteq V$ is given as an ordered list of vertices. Let $V_L = \pi \backslash \{n+1\}$, $V_R = \pi \backslash \{0\}$, and $V_N = V_L \cap V_R$. Furthermore, we call $V_D \subseteq V_N$ the *drone-serviceable* subset of customers. As in Section 3, we assume that t_{ij} (d_{ij}) and $\bar{t}_{ij}$ ($\bar{d}_{ij}$) are the times (distances) required to travel from $i \in V_L$ to $j \in \pi$ by truck and drone, respectively. We use the notation $i \prec j$ ($i \succ j$) to indicate that the index associated with i in the ordered list π is smaller (larger) than the index associated with j. Furthermore, $i \preceq j$ ($i \succeq j$) implies that the index of i is smaller (larger) than or equal to the index of j.

For the DASP, we will now provide two MILP formulations. Section 4.2.1 provides a formulation, where a drone operation (i, w, j) is characterized through three indices, similar to the decision variables that were already used in Section 3. Alternatively, Section 4.2.2 models each drone operation (i, w, j) through two disjoint arcs (i, w) and (w, j). The latter formulation reduces the number of variables required for modelling the problem at the cost of additional constraints.

4.2.1. Three-index-operation formulation

In this section, we model a drone operation (i, w, j) directly through three indices. Compared to the two-index formulation presented in Section 4.2.2, this makes the number of variables larger but the model itself compact. For modelling the problem, we introduce the following decision variables:

y_w $\forall w \in V_N$	:	Binary variables that determine if $w \in V_N$ is served by any drone.
o_i^d $\forall d \in \mathcal{D}, i \in V_L$	:	Binary variables that indicate if drone $d \in \mathcal{D}$ is available at $i \in V_L$.
y_{iwj}^d $\forall d \in \mathcal{D}, i \in V_L, w \in V_D, j \in V_R, i \prec w \prec j$	:	Binary variables that specify if a drone $d \in \mathcal{D}$ performs an acyclic operation (i, w, j).
y_{iwi}^d $\forall d \in \mathcal{D}, i \in V_L, w \in V_D, i \prec w$	:	Binary variables that determine if a drone $d \in \mathcal{D}$ performs a cyclic operation (i, w, i).
a_i $\forall i \in \pi$	:	Continuous variables with a lower bound of 0 that specify the earliest possible arrival time of the truck at i.
s_i $\forall i \in \pi$	:	Continuous variables with a lower bound of 0 that specify the earliest possible departure time of the truck from i.
r_i^d $\forall d \in \mathcal{D}, i \in \pi$	:	Continuous variables with a lower bound of 0 that specify the earliest possible release time of drone d at i.
s_i^d $\forall d \in \mathcal{D}, i \in \pi$	:	Continuous variables with a lower bound of 0 that specify the earliest possible time at which drone d is available for a departure from i.

Given $\mathcal{G}(V, E)$ and a fixed sequence of vertices π, MILP (39)–(54) is used to find the best possible assignment and schedule of drones such that the arrival time of the tandem at vertex $n+1$ is minimized:

$$\min s_{n+1} \tag{39}$$

$$\text{s. t.} \quad y_w = 0: \quad \forall w \in V_N \backslash V_D, \tag{40}$$

$$\sum_{d \in \mathcal{D}} \sum_{\substack{i \in V_L, \\ i \prec w}} \left(y_{iwi}^d + \sum_{\substack{j \in V_R, \\ w \prec j}} y_{iwj}^d \right) = y_w: \quad \forall w \in V_D, \tag{41}$$

$$\sum_{d \in \mathcal{D}} \sum_{\substack{w \in V_D, \\ v \prec w}} \left(y_{vwv}^d + \sum_{\substack{j \in V_R, \\ w \prec j}} y_{vwj}^d \right) \leqslant M \left(1 - y_v \right): \quad \forall v \in V_N, \tag{42}$$

$$\sum_{d \in \mathcal{D}} \sum_{\substack{w \in V_D, \\ w \prec v}} \sum_{\substack{i \in V_L, \\ i \prec w}} y_{iwv}^d \leqslant M \left(1 - y_v \right): \quad \forall v \in V_N, \tag{43}$$

$$-M(y_i + y_j) + s_i + t_{ij} \leqslant a_j: \quad i \in V_L, j \in V_R, i \prec j, \tag{44}$$

$$a_i \leqslant s_i: \quad \forall i \in V, \tag{45}$$

$$-M\left(1-y^d_{iwj}\right)+s^d_i+\left(\bar{t}_l+\bar{t}_{iw}+\bar{t}_{wj}+\bar{t}_r\right)\leqslant r^d_j:\quad \forall\, d\in\mathcal{D},\, i\in V_L,\, w\in V_D,\, j\in V_R,\, i\prec w\prec j, \tag{46}$$

$$r^d_i+\sum_{\substack{w\in V_D,\\ i\prec w}}\left(\bar{t}_l+\bar{t}_{iw}+\bar{t}_{wj}+\bar{t}_r\right)y^d_{iwi}\leqslant s^d_i:\quad \forall\, d\in\mathcal{D},\, i\in V_L, \tag{47}$$

$$s^d_i+\bar{t}_l\sum_{\substack{w\in V_D,\\ i\prec w}}\ \sum_{\substack{j\in V_R,\\ w\prec j}}y^d_{iwj}\leqslant s_i:\quad \forall\, d\in\mathcal{D},\, i\in V, \tag{48}$$

$$a_j+\bar{t}_r\sum_{\substack{i\in V_L,\\ i\prec w}}\ \sum_{\substack{w\in V_D,\\ w\prec j}}y^d_{iwj}\leqslant r^d_j:\quad \forall\, d\in\mathcal{D},\, j\in V, \tag{49}$$

$$\left(\bar{d}_{iw}+\bar{d}_{wj}\right)y^d_{iwj}\leqslant\mathcal{E}:\quad \forall\, d\in\mathcal{D},\, i\in V_L,\, w\in V_D,\, j\in V,\, i\prec w\prec j, \tag{50}$$

$$(\bar{d}_{iw}+\bar{d}_{wi})y^d_{iwi}\leqslant\mathcal{E}:\quad \forall\, d\in\mathcal{D},\, i\in V_L,\, w\in V_D,\, i\prec w, \tag{51}$$

$$o^d_h+\sum_{\substack{i\in V_L,\\ i\prec h}}\ \sum_{\substack{w\in V_D,\\ i\prec w}}\ \sum_{\substack{j\in V_R,\\ j\succ h}}y^d_{iwj}=1:\quad \forall\, d\in\mathcal{D},\, h\in V_N, \tag{52}$$

$$\sum_{\substack{w\in V_D,\\ i\prec w}}\ \sum_{\substack{j\in V_R,\\ w\prec j}}y^d_{iwj}\leqslant o^d_i:\quad \forall\, d\in\mathcal{D},\, i\in V_L, \tag{53}$$

$$\sum_{\substack{w\in V_D,\\ i\prec w}}y^d_{iwi}\leqslant Mo^d_i:\quad \forall\, d\in\mathcal{D},\, i\in V_L. \tag{54}$$

The objective function (39) minimizes the time required such that all customers are served and the truck and drones have returned to the depot. The set of constraints (40) and (41) force the variables y_w to 1 if a customer w is served by any drone and to value 0 for any customer that can not be served by drone. Furthermore, they guarantee that each customer is served at most once by a drone. Constraints (42) and (43) impose that no drone sortie can start or end at a customer v that is served by a drone. Lower bounds on the arrival time of the truck are specified through the set of constraints (44). Moreover, due to constraints (45) the departure time of the truck is bound by its arrival time at each vertex. For each drone operation (i, w, j), constraints (46) provide valid lower bounds on the release times. Constraints (47) ensure that a drone can only depart from a vertex as soon as all cyclic operations have been completed. The earliest possible departure time of each drone qualifies as a lower bound on the earliest possible departure time of the truck (constraints (48)). Furthermore, constraints (49) enforce that a drone can only be released after the truck has arrived. Constraints (50) and (51) ensure that no drone operation violates the maximum endurance. In practice, this can be efficiently handled during a preprocessing step such that the associated variables y^d_{iwj} are bound to 0 if the operation violates the endurance. Constraints (52) determine the values on the variables o^d_h, that indicate if a drone is available for an operation at each vertex h. If there is an operation (i, w, j) that occurs circumjacent to h, then the drone is unavailable at h. Finally, constraints (53) and (54) ensure that at each vertex, a drone can only perform an acyclic and cyclic operation, respectively, if the drone is available.

For completeness, in principle, it is also possible to introduce some additional auxiliary binary variables x_{ij} where $i\in V_L,\ j\in V_R,\ i\prec j$, that explicitly determine the path of the truck. This might be done by including the following inequalities (55)–(57) as constraints in the model and by replacing (44) with (58):

$$x_{ij}\leqslant(1-y_i):\quad \forall\, i\in V_L, j\in V_R, i\prec j, \tag{55}$$

$$x_{ij}\leqslant(1-y_j):\quad \forall\, i\in V_L, j\in V_R, i\prec j, \tag{56}$$

$$\left(1-y_i\right)+\left(1-y_j\right)-1-\sum_{\substack{w\in V_N,\\ i\prec w,\\ w\prec j}}\left(1-y_w\right)\leqslant x_{ij}:\quad \forall\, i\in V_L, j\in V_R, i\prec j, \tag{57}$$

$$s_i+t_{ij}x_{ij}\leqslant a_j:\quad \forall\, d\in D,\, i\in V_L, j\in V,\, i\prec j. \tag{58}$$

Here constraints (55) and (56) ensure that the truck departs or arrives at a customer that is served by drone. Furthermore, constraints (57) guarantees that x_{ij} is equal to 1 if neither i nor j are served by drone and all remaining customers w in between i and j are not served by the truck. Adding these constraints would remove the M in the set of constraints (44) and allow for the inclusion of VIEQ (59) and (60). These VIEQ are analogous to the previously introduced VIEQ (30) and (32) and have proven themselves to be very beneficial in solving MILP (1)–(27) more efficiently.

$$\sum_{i \in V} \left(s_i - a_i \right) + \sum_{i \in V_L} \sum_{\substack{j \in V_R, \\ i \prec j}} t_{ij} x_{ij} \leqslant s_{n+1}, \tag{59}$$

$$\sum_{i \in V_L} \sum_{\substack{w \in V_D, \\ i \prec w}} \left(\left(\bar{t}_{iw} + \bar{t}_{wi} \right) y_{iwi}^d + \sum_{\substack{j \in V_R, \\ w \prec j}} \left(\bar{t}_{iw} + \bar{t}_{wj} \right) y_{iwj}^d \right) \leqslant s_{n+1} \colon \forall\, d \in \mathcal{D}. \tag{60}$$

However, we do not use these additional variables and constraints when solving the DASP.

4.2.2. Two-index-operation formulation

Another possibility is to represent an operation (i, w, j) through two disjoint arcs (i, w) and (w, j), instead. This allows for a significantly reduced number of variables at the cost of several additional constraints that guarantee valid drone operations. For this formulation, consistent with the three-index formulation, we use the variables y_w, o_i^d, a_i, s_i, r_i^d, and s_i^d (refer to Section 4.2.1). Furthermore, we introduce the following decision variables:

y_{iw}^d, $\forall\, d \in \mathcal{D}, i \in V_L, w \in V_D, i \prec w$: Binary variables that specify if a drone $d \in \mathcal{D}$ performs an acyclic *delivery* to $w \in V_D$, which starts at $i \in V_L$.

$\tilde{y}_{wj}^d$, $\forall\, d \in \mathcal{D}, w \in V_D, j \in V_R, w \prec j$: Binary variables that specify if a drone $d \in \mathcal{D}$ is *retrieved* at $j \in V_R$ after an acyclic delivery to $w \in V_D$.

z_{iw}^d, $\forall\, d \in \mathcal{D}, i \in V_L, w \in V_D, i \prec w$: Binary variables that determine if a drone $d \in \mathcal{D}$ performs a cyclic delivery (i, w, i).

The model is then given as follows, where (61) marks the objective and the constraints are given by (62)–(78).

$$\min \quad s_{n+1} \tag{61}$$

$$\text{s. t.} \quad y_w = 0 \colon \quad w \in V_N \backslash V_D, \tag{62}$$

$$\sum_{k \in D} \sum_{\substack{i \in V_L \\ i \prec w}} \left(y_{iw}^d + z_{iw}^d \right) = y_w \colon \quad w \in V_D, \tag{63}$$

$$\sum_{d \in \mathcal{D}} \sum_{\substack{v \in V_D, \\ w \prec v}} \left(y_{wv}^d + z_{wv}^d \right) \leqslant M \left(1 - y_w \right) \colon \quad \forall\, w \in V_N, \tag{64}$$

$$\sum_{d \in \mathcal{D}} \sum_{\substack{v \in V_D, \\ v \prec w}} \left(\tilde{y}_{vw}^d + z_{vw}^d \right) \leqslant M \left(1 - y_w \right) \colon \quad \forall\, w \in V_N, \tag{65}$$

$$\sum_{\substack{i \in V_L, \\ i \prec w}} y_{iw}^d = \sum_{\substack{j \in V_R, \\ w \prec j}} \tilde{y}_{wj}^d \colon \quad \forall\, d \in \mathcal{D}, w \in V_D, \tag{66}$$

$$-M \left(y_i + y_j \right) + s_i + t_{ij} \leqslant a_j \colon \quad \forall\, i \in V_L, j \in V, i \prec j, \tag{67}$$

$$a_i \leqslant s_i \colon \quad \forall\, i \in V, \tag{68}$$

$$-M \left(1 - y_{iw}^d \right) + s_i^d + \bar{t}_i + \bar{t}_{iw} \leqslant r_w^d \colon \quad \forall\, d \in \mathcal{D}, i \in V_L, w \in V_D, i \prec w, \tag{69}$$

$$-M\left(1-\tilde{y}_{wj}^{d}\right)+s_{w}^{d}+\bar{t}_{wj}+\bar{t}_{r}\leqslant r_{j}^{d}:\quad \forall\, d\in\mathcal{D},\, w\in V_{D},\, j\in V_{R},\, i\prec w, \tag{70}$$

$$r_{i}^{d}+\sum_{\substack{w\in V_{D},\\ i\prec w}}\left(\bar{t}_{l}+\bar{t}_{iw}+\bar{t}_{wi}+\bar{t}_{r}\right)z_{iw}^{d}\leqslant s_{i}^{d}:\quad \forall\, d\in\mathcal{D},\, i\in V_{L}, \tag{71}$$

$$s_{i}^{d}+\bar{t}_{l}\sum_{\substack{w\in V_{D},\\ i\prec w}}y_{iw}^{d}\leqslant s_{i}:\quad \forall\, d\in\mathcal{D},\, i\in V, \tag{72}$$

$$a_{j}+\bar{t}_{r}\sum_{\substack{w\in V_{D},\\ w\prec j}}\tilde{y}_{wj}^{d}\leqslant r_{j}^{d}:\quad \forall\, d\in\mathcal{D},\, j\in V, \tag{73}$$

$$\sum_{\substack{i\in V_{L},\\ i\prec w}}\bar{d}_{iw}y_{iw}^{d}+\sum_{\substack{j\in V_{R},\\ w\prec j}}\bar{d}_{wj}\tilde{y}_{wj}^{d}\leqslant\mathcal{E}:\quad \forall\, d\in\mathcal{D},\, w\in V_{D}, \tag{74}$$

$$\sum_{\substack{i\in V_{L},\\ i\prec w}}\left(\bar{d}_{iw}+\bar{d}_{wi}\right)z_{iw}^{k}\leqslant\mathcal{E}:\quad \forall\, d\in\mathcal{D},\, w\in V_{D}, \tag{75}$$

$$o_{h}^{d}+\sum_{\substack{i\in V_{L},\\ i\prec h}}\sum_{\substack{w\in V_{D},\\ w\succ i}}y_{iw}^{d}-\sum_{\substack{j\in V_{R},\\ j\preceq h}}\sum_{\substack{w\in V_{D},\\ w\prec j}}\tilde{y}_{wj}^{d}=1:\quad \forall\, d\in\mathcal{D},\, h\in V_{N}, \tag{76}$$

$$\sum_{w\in V_{D}}y_{iw}^{d}\leqslant o_{w}^{d}:\quad \forall\, d\in\mathcal{D},\, i\in V_{L}, \tag{77}$$

$$\sum_{w\in V_{D}}z_{iw}^{d}\leqslant Mo_{w}^{d}:\quad \forall\, d\in\mathcal{D},\, i\in V_{L}. \tag{78}$$

Overall, this model closely follows the model presented in Section 4.2.1. Here, the objective function is given by (61) and minimizes the arrival time at the depot. Constraints (62) force the variables y_w to value 0 for any customer that can not be served by drone. Furthermore, the variables y_w are forced to value one by constraints (63) if a drone-eligible customer w is served by any drone. This constraint also guarantees that there can be at most one delivery by drone to each customer. Constraints (64) and (65) ensure that no drone starts or ends a delivery at a customer w that is served by any drone. For each delivery that does not start and end at the same location, there must be exactly one incoming and one outgoing arc, that are synchronized through constraints (66).

The set of constraints (67) provides lower bounds on the arrival time of the truck. For acyclic drone operations, constraints (69) and (70) account for the release times of drones during departure and return flight, respectively. Furthermore, the set of constraints (71) provide a lower bound on the departure time of the drone at each vertex in case of cyclic operations. The departure time of the truck is bound by its arrival time through constraints (68) and the departure time of drones (constraints (72)). Furthermore, constraints (73) guarantee that at each vertex where a drone is retrieved, the drone can only be released after its respective truck has arrived. The endurance limit is implemented through the set of constraints (74) and (75). Constraints (76) determine the value on the variables o_h^d, that indicate if a drone k is available for an operation at vertex h. In particular, if a drone k performs a circumjacent operation at h, then o_h^d is forced to 0. Finally, constraints (77) and (78) guarantee that a drone can only perform acyclic and cyclic operations, respectively, if it is available.

4.3. Discussion and scalability

As outlined in Section 4, through the min-max objective of the VRPD, a significant speed up can be achieved in our matheuristic framework through a cutoff and the subsequent prevention of unnecessary evaluations of either MILP. This cutoff can be applied, whenever the MILP yields a lower bound that is greater than or equal to the incumbent solution $f(x^*)$ and by an early termination of the solver whenever the upper bound moves beneath the current $\mathcal{RUB}$ (see Fig. 4).

Both, MILP (39)–(54) and MILP (61)–(78), can be solved relatively quickly for reasonably sized routes. Fig. 5 shows a sample solution found through the matheuristic (Algorithm 1 with the Three-Index DASP formulation) within just a few seconds. As we have outlined, the number of variables required for modelling the DASP is $O(|\pi|^3)$ and $O(|\pi|^2)$ for the Three- and Two-Index-Operation formulation, respectively. For routes that contain a small amount of customers, the performance difference is negligible. Nevertheless, the reduced number of constraints involved in the formulation presented in Section 4.2.1 and the large number of variables that can be removed during preprocessing often make it the preferred formulation for the instances considered in this work. Overall,

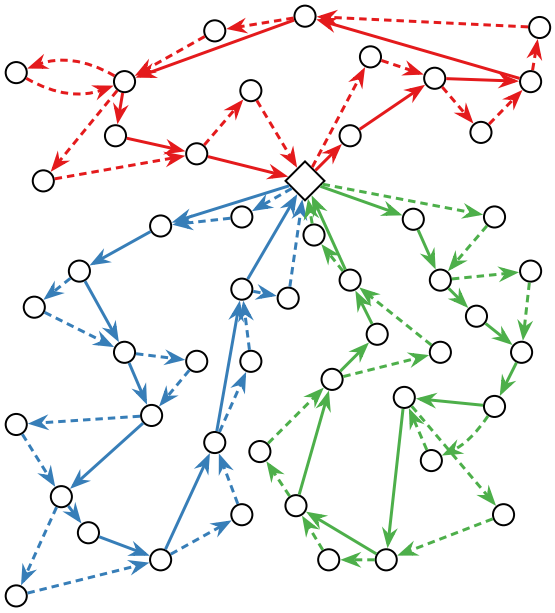


Fig. 5. A sample solution for an instance with 50 customers found through the matheuristic with $|\mathcal{K}| = 3$, $|\mathcal{D}| = 1$.

the runtime is heavily influenced by the set of binary variables that determine the drone operations. In particular, the value of $\mathcal{E}$ can severely limit the number of feasible drone operations and therefore significantly reduce the model size. This can be effectively exploited by the three-index formulation during preprocessing.

When solving the DASP for larger routes with up to 100 customers, it might take the solver several seconds to resolve the MILP formulation. At the cost of an optimal drone schedule, an improved runtime behavior can be achieved in several ways:

- It is easily possible to restrict the set of feasible drone operations further, e.g., by limiting the maximum number of vertices that can be visited by the truck between the launch and recovery of a drone (similar approaches were used by Bouman et al. (2018), Boysen et al. (2018)).
- Furthermore, at the expense of potentially sub-optimal drone placements, a time limit or relative MIP gap might be imposed on the MILP solver when solving the DASP.

As an alternative approach, within the matheuristic, we suggest splitting the route (permutation) π into m overlapping partitions $\pi_1|...|\pi_m$. As an example, consider the route $\pi = (0, ..., n/2, ..., n+1)$ where n is an even number in $\mathbb{Z}^+$. If n is very large, instead of solving the DASP on π, we might consider two (or m) overlapping partitions $\pi_1 = (0, ..., n/2)$ and $\pi_2 = (n/2, ..., n+1)$ such that the makespan $s_{n/2}$ determined by solving the DASP on π_1 marks the starting time of the DASP in π_2. This might be achieved by replacing line 11 in Algorithm 1 with Algorithm 2. In this algorithm, the function Partition (π, m) generates m overlapping partitions of π, $f(x_\pi)$ is the accumulated cost associated with the route after evaluating each partition, and $f(x_{\pi_i})$ is the cost of solving the DASP on each partition π_i. Preliminary experiments have shown that this approach is well-suited for large routes where the runtime behavior is improved significantly with little or no loss on the solution quality. Fig. 6 shows a sample application of the DASP involving many partitions.

Algorithm 2. DASP Partition

1: $f(x_\pi) = 0$
2: **for** $\pi_i \in$ Partition(π, m) **do**
3: **if** $i \neq m$ **then**
4: $x_{\pi_i} \leftarrow$ DASP$(\pi_i$, BestObjStop(0), Cutoff$(f(x^*) - f(\pi)))$
5: **else**
6: $x_{\pi_i} \leftarrow$ DASP$(\pi_i$, BestObjStop$(\mathcal{RUB} - f(\pi))$, Cutoff$(f(x^*) - f(\pi)))$
7: **end if**
8: $f(x_\pi) \leftarrow f(x_\pi) + f(x_{\pi_i})$
9: **end for**

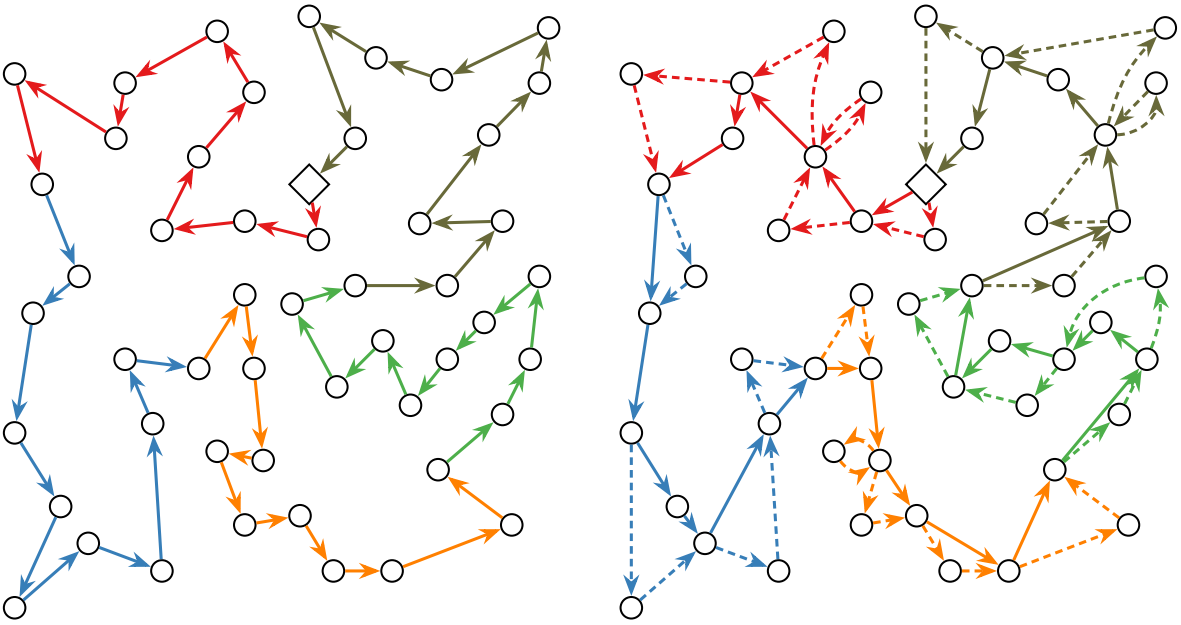


Fig. 6. A sample solution that was found through the Clarke-Wright savings heuristic and improved through local search for a single vehicle (left-hand side) and the respective solution after solving the DASP ($|\mathcal{D}| = 1$) on each partition, as indicated by the different colors (right-hand side).

5. Computational experiments and their numerical results

We carried out two main classes of computational experiments, on small- and large-scale instances and we present the numerical results in this section. More precisely, in Section 5.1 we discuss the performance of Gurobi Optimizer in solving MILP (1)–(27). Moreover, we highlight the benefits of adding the VIEQ with bounds on the makespan (30) and (32), the symmetry breaking constraints (34), and (knapsack) inequalities (37) and (38) to the model. Finally, we compare the performance of the matheuristic in solving the same instances through the three-index formulation of the DASP MILP (39)–(54) and two-index formulation of the DASP MILP (61)–(78). Afterwards, Section 5.2 shows that the proposed matheuristic might also be used to solve large-scale instances.

For the matheuristic, the set of constraints (46) for the three-index formulation and (69) and (70) for the two-index formulation were treated as lazy constraints. All experiments were performed on Intel® Xeon® Gold 6126 CPUs and 16 GB of RAM. We implemented our algorithms in Java SE 8 and for solving MILPs we used Gurobi Optimizer 8.1.0 (Gurobi Optimization, 2019).

5.1. Small instances

In this section, we describe the computational experiments on small instances and provide their numerical results. Since there is no commonly accepted benchmark for the VRPD, we perform our experiments on the benchmark instances that were used by Agatz et al. (2018) for the TSP-D and Murray and Chu (2015) for the FSTSP. More precisely, in Section 5.1.1 we test the MILP and matheuristic on the instances used by Agatz et al. (2018). Furthermore, in Section 5.1.2, we evaluate our model and heuristic on the instances used by Murray and Chu (2015). On the latter instances, to respect the problem description of Murray and Chu (2015) and to produce comparable results, several minor adjustments to the VRPD and DASP MILPs are necessary that we explain in detail in Appendix B.

5.1.1. Small instances used by Agatz et al. (2018)

In this section, we provide the results that we obtained on the instances used by Agatz et al. (2018). These instances are specified by a symmetric graph $\mathcal{G}(V, E)$ and assume that the truck and drone follow the Euclidean distance metric. Furthermore, the instances specify two parameters: the relative velocity α and the endurance $\mathcal{E}$. In particular, through a relative endurance parameter $\mathcal{E}_r \in \mathbb{R}^+$, the endurance can be calculated as follows, where $\max(E)$ refers to the edge with maximum weight in the graph $\mathcal{G}$:

$$\mathcal{E} := \mathcal{E}_r \cdot \max(E) \colon \mathcal{E}_r \in [0, 2]. \tag{79}$$

In this case, $\mathcal{E}_r = 0$ guarantees that no drone operation is possible while $\mathcal{E}_r = 2$ marks any operation as feasible. The velocity of the drones $\overline{v}$ is assumed to be $\alpha \in \mathbb{R}^+$ times the velocity of the truck. Furthermore, the times required to launch and retrieve drones, i.e., $\bar{t}_l$ and $\bar{t}_r$, are assumed to be negligible. Agatz et al. (2018) provide 10 instances (with 10 vertices each), where each instance is associated with a value $\alpha \in \{1, 2, 3\}$ and a value $\mathcal{E}_r \in \{0.2, 0.4, 0.6, 1.0, 1.5, 2.0\}$ yielding a total of 180 different instances. By definition of Wang et al. (2017,), the number of trucks and drones per truck are a parameter in the VRPD. Thus, as a meaningful selection on for small instances, we limit ourselves to $|\mathcal{K}| \in \{1, 2\}$ and $|\mathcal{D}| \in \{1, 2\}$, i.e., we either have one or two trucks that are equipped with

either one or two drones each. Therefore, we consider a total of $2 \cdot 2 \cdot 180 = 720$ instances for our experiments. We solve these 720 instances in five different ways as follows:

1. We solve the MILP (1)–(27) with no VIEQ.
2. We further include the VIEQ (30)–(32) that provide bounds on the makespan to the model.
3. Additionally, we include the symmetry breaking constraints (34) and (knapsack) inequalities (37) and (38) to previous model and VIEQ.
4. Finally, we consider the matheuristic with either the three-index formulation MILP (39)–(54) or the two-index formulation MILP (61)–(78) for solving the DASP.

When solving the VRPD as MILPs, we limit the runtime of the solver to 15 min. For solving the VRPD through the matheuristic, we limit the runtime of the heuristic to 10 min. Furthermore, within this time limit, if the incumbent value remains unchanged for more than 1 min, the heuristic terminates. The parameter r_{lb} that randomizes the savings during initialization was set to 0.7. Furthermore, the number of iterations spent on local search were set to $\lceil |V|^{1.5} \rceil$, where $|V|$ marks the number of vertices in the graph. Finally, we perform five runs of the matheuristic per instance.

In total, this yields $3 \cdot 720 = 2160$ unique experiments using the MILP and the VIEQ and $5 \cdot 720 = 3600$ unique experiments using the matheuristic with each MILP for solving the DASP. Therefore, we have done $2160 + 2 \cdot 3600 = 9360$ experiments in total.

For the analysis of our results, we introduce the following metric. More precisely, given a fixed number of trucks, Δ indicates the relative change in makespan through the introduction of drones and is calculated as follows:

$$\Delta := 100\% - \frac{\text{objective value}}{\text{optimal objective value with } \mathcal{D} = \{\}} \tag{80}$$

Tables D.1–D.3 (see Appendix D) show the average solutions achieved by solving the instances through steps 1.–3. (see above). Each table shows the average value of Δ, the average MIP gap returned by the solver, the average runtime t, the average number of acyclic and cyclic operations in a solution vector $\#a$ and $\#c$, respectively, and the number of runs where an optimal solution was found within the runtime limit.

Furthermore, Tables D.4 and D.5 (see Appendix D) show the average solution achieved through the matheuristic using the three- and two-index DASP formulation, respectively. In this case, we report the average value of Δ, the average runtime t_{best} after which the incumbent solution was found, the average number of changes in the incumbent solution until convergence n, the average number of acyclic and cyclic operations, respectively, and the share of solutions where the heuristic found a solution of equal or better quality than the one returned by the solver.

Overall, we can make the following observations. Regarding the MILP in Tables D.1–D.3, on average, the introduction of the VIEQ significantly improved the performance of the MILP solver. In particular, we can observe a larger share of instances that are solved to optimality and significantly reduced runtimes (and MIP gaps) while doing so. The average time to optimality is significantly influenced by the number of drones and the relative endurance $\mathcal{E}_r$ which by implication have a large influence on the number of binary variables y_{iwj}^{kd} that might be removed during preprocessing. Furthermore, $\mathcal{E}_r$ has a strong influence on the quality of the root relaxation. This is highlighted in Table 2, which shows the average objective, iterations, and time (in seconds) after solving the root relaxation. In this table, the column labeled $\%_R$ is calculated as follows:

$$\%_R := \frac{\text{root relaxation objective}}{\text{objective value of the feasible solution after runtime}} \tag{81}$$

Regarding the matheuristic in Tables D.4 and D.5, on average, optimal or near-optimal solutions are found in almost all cases. On small instances, there is no significant difference in solving the DASP through either the three- or two-index formulation with regards to the time required to solve the MILP. In particular, the matheuristic converges very fast and does not require many changes to the incumbent solution to find optimal solutions. Solving the DASP through mathematical programming provides an intensive local search mechanism that is also very robust. In some cases, the optimal VRPD solution can be constructed by solving the DASP on structurally similar but not necessarily equal routes (i.e., distinct routes that share a large number of the same arcs).

Table 2
Results obtained after solving the root relaxation depending on the relative endurance $\mathcal{E}_r$ (averaged over all instances).

$\mathcal{E}_r$	MILP (1)–(27) without VIEQ				MILP + makespan bounds				MILP + all VIEQ			
	Objective	Iterations	Time	$\%_R$	Objective	Iterations	Time	$\%_R$	Objective	Iterations	Time	$\%_R$
0.2	0	281.1	0.0	0.0%	163.0	399.7	0.0	58.0%	163.0	388.1	0.0	58.0%
0.4	0	334.5	0.0	0.0%	131.8	579.2	0.1	47.5%	131.8	536.6	0.1	47.5%
0.6	0	438.4	0.2	0.0%	82.5	892.5	0.3	31.0%	82.5	825.0	0.3	31.0%
1.0	0	502.2	0.8	0.0%	65.4	1,017.0	1.5	28.1%	65.4	1,016.1	1.3	28.3%
1.5	0	588.4	1.4	0.0%	62.3	1,081.1	2.8	30.6%	62.3	1,167.1	2.3	30.7%
2.0	0	227.5	1.2	0.0%	62.3	1,348.1	3.6	31.7%	62.3	1,454.8	3.4	32.0%
Average	0	395.3	0.6	0.0%	94.6	886.3	1.4	37.8%	94.6	898.0	1.2	37.9%

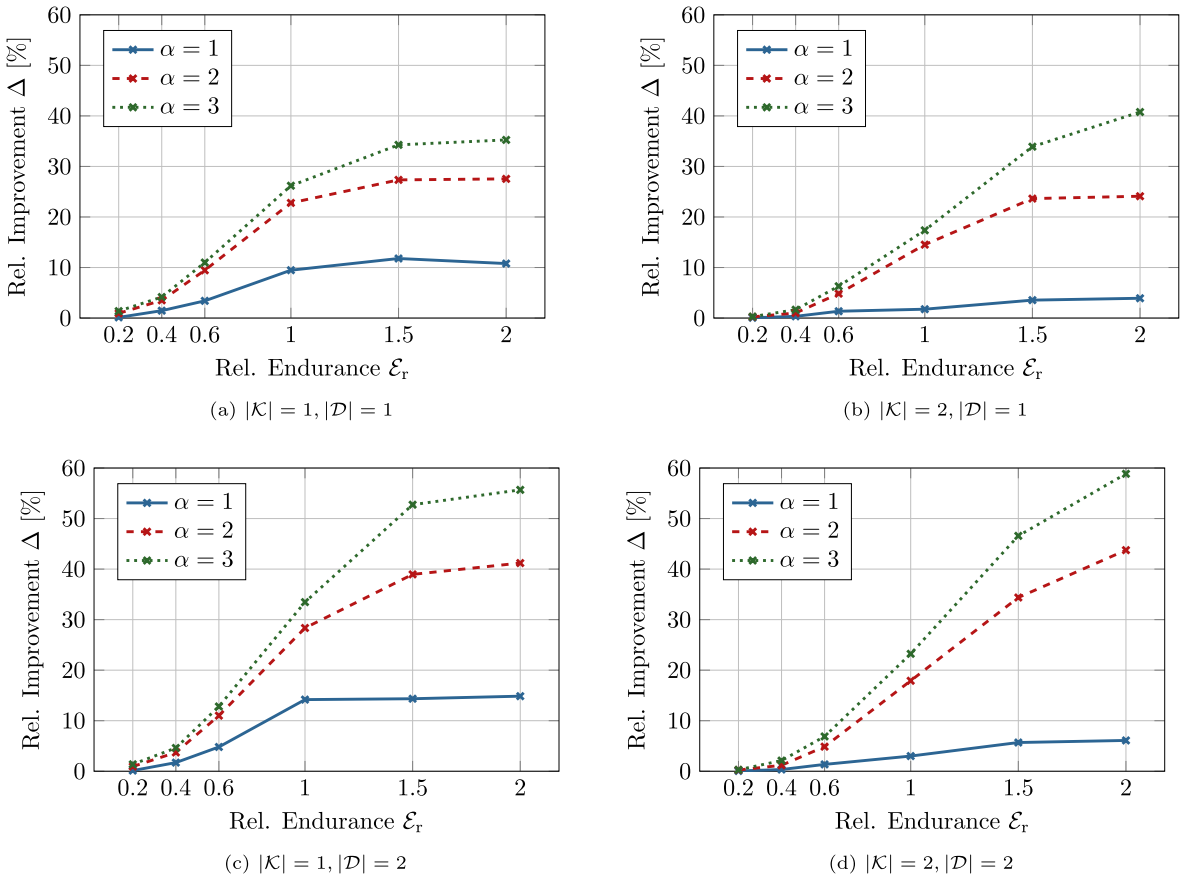


Fig. 7. The relative improvement Δ depending on the number of vehicles $|\mathcal{K}|$, the number of drones per vehicle $|\mathcal{D}|$, the relative endurance $\mathcal{E}_r$, and the relative velocity α (averaged over 10 instances).

In almost all cases, the matheuristic is able to provide a solution that is at least as good (or better) than the one provided by the MILP solver. In order to highlight this behavior, Tables D.4 and D.5 contain a column that shows the number of times where a solution of equal or better quality was found by the matheuristic.

In what follows, we will provide a more detailed analysis of solutions based on Table D.3 (MILP with all VIEQ active) which performed the best on average. To this end, Fig. 7 shows the average relative improvement Δ as a function of the relative endurance $\mathcal{E}_r$, the relative velocity α, the number of trucks $|\mathcal{K}|$, and the number of drones per truck $|\mathcal{D}|$. Overall, we can observe S- or logarithmic-shaped curves in all cases, indicating a diminishing gradient with regards to an increase in the relative endurance. Recall that a value of $\mathcal{E}_r = 0$ indicates that no drone operation is possible while a value of $\mathcal{E}_r = 2$ indicates that, in principle, all operations are possible. In all cases, even small increases in the drone's endurance can lead to significant savings (e.g., moving $\mathcal{E}_r$ from 0 to 1) whereas further increases often lead to smaller returns (e.g., moving $\mathcal{E}_r$ from 1 to 2). In general, α and $\mathcal{E}_r$ seem strongly correlated. If the drone is relatively slow, performing operations that require a large endurance does not seem to be beneficial. In contrast, if the drone is relatively fast, it can be beneficial to perform operations that require a high endurance and visit distant locations.

For a more detailed analysis on the performed operations, see Fig. 8. This figure shows the average number of acyclic and cyclic operations performed by drones as a function of the relative endurance $\mathcal{E}_r$, the relative velocity α, the number of trucks $|\mathcal{K}|$, and the number of drones per truck $|\mathcal{D}|$. Overall, the share of acyclic operations performed is much larger than the share of cyclic operations. Moreover, the number of acyclic operations performed as a function of the relative endurance also appears S-shaped. This might be an indication that after reaching a certain threshold of relative endurance $\mathcal{E}_r$, depending on the relative velocity α, the operations performed by drones will not change structurally. These results are in line with Fig. 7. However, we also have an indication that an improvement in Δ is not necessarily linked to additional drone operations. To this end, consider, e.g., the cases for $\alpha \in \{2, 3\}$, $|\mathcal{K}| = 1$, $|\mathcal{D}| = 1$ in Figs. 7 and 8. Even though the number of acyclic and cyclic operations is almost identical, the relative improvements Δ are very much different. Therefore, in some cases, it can be beneficial to perform the same operations at an increased relative velocity. Clearly, once the drone is fast enough, the velocity of the truck becomes the bottleneck such that an increase in the relative velocity yields no further improvement.

Finally, Fig. 9 shows the relative improvement as box plots based on the 10 graphs (that specify the customers and depot) that we used in our experiments. On this figure, we observe that even though the median relative improvement is typically large, the minimum and maximum values can differ significantly. In particular for the cases of $\alpha = 3$, the difference between the maximum and

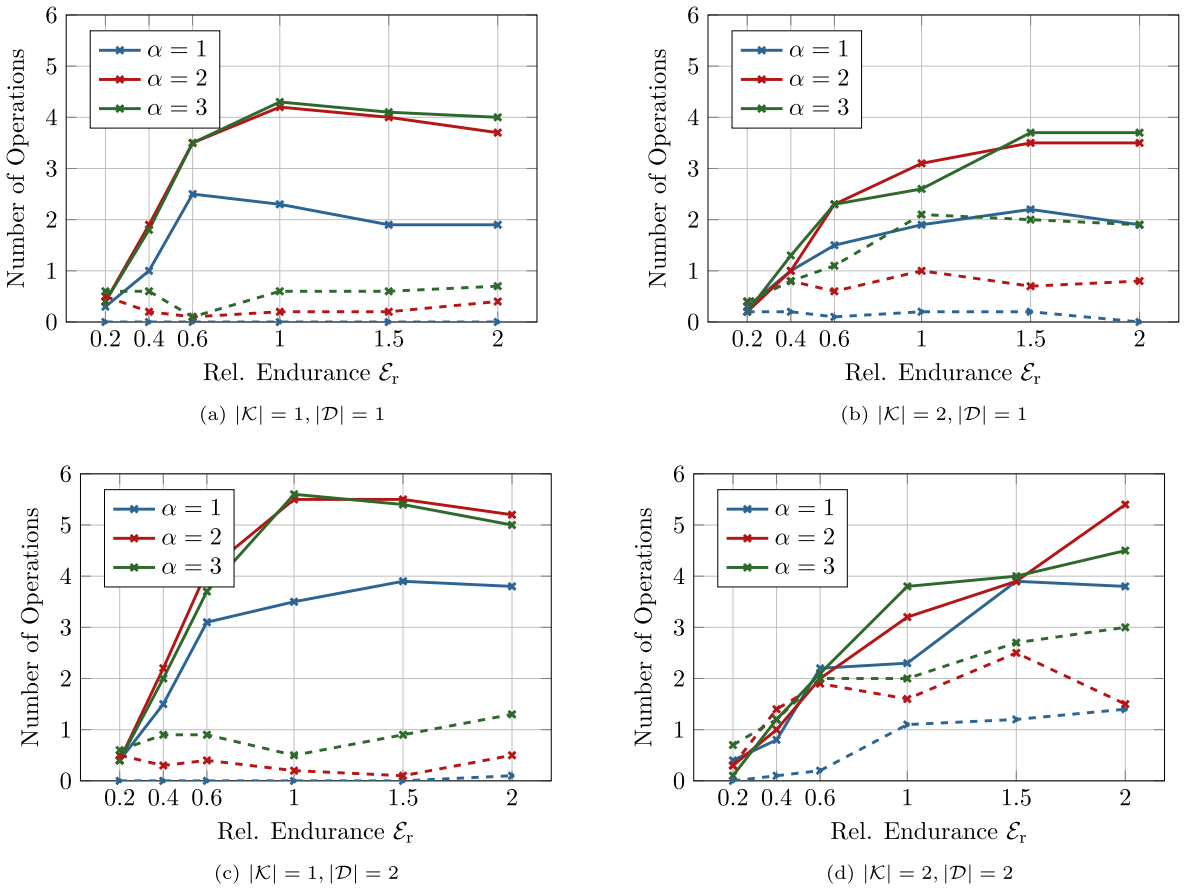


Fig. 8. The average share of acyclic operations (solid lines) and cyclic operations (dashed lines) depending on the number of vehicles $|\mathcal{K}|$, the number of drones per vehicle $|\mathcal{D}|$, the relative endurance $\mathcal{E}_r$, and the relative velocity α (averaged over 10 instances).

minimum relative improvement when comparing different problem instances can be large.

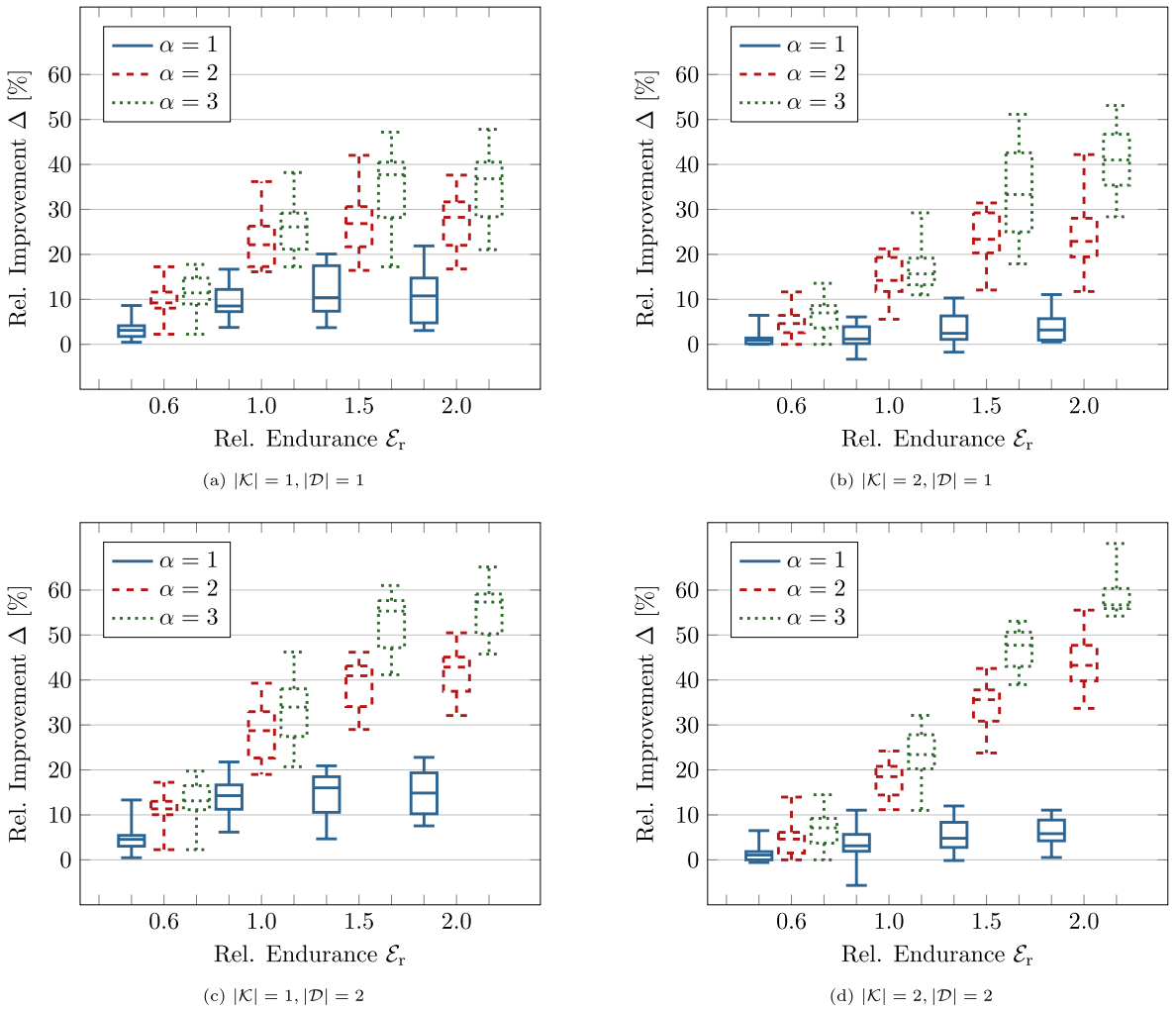
5.1.2. *Small instances used by Murray and Chu (2015)*

To show that our model and heuristic are also well-suited to the assumptions and instances of Murray and Chu (2015), we provide some computational evidence in this section. In these instances, there is a single truck which is equipped with a single drone that follow a Manhattan and Euclidean distance matrix at 25 miles per hour (mph) and 15, 25, or 35 mph, respectively. For the sake of compactness, in this section, we respect these assumptions and do not consider a multi-truck or multi-drone case. Murray and Chu (2015) provide 72 problems that contain 10 customers distributed across an 8-mile square region (80–90% of which are drone serviceable) and 1 depot location i.e., 11 vertices in total (one more vertex compared to the previous section). Among these 72 instances, the endurance of the drone is either 20 or 40 min.

Several minor modifications to the VRPD and DASP MILPs are necessary to match the problem formulation of Murray and Chu (2015). A detailed discussion is provided in Appendix B. Due to the effectiveness of the valid inequalities (refer to Section 5.1.1), we solved the modified MILP through Gurobi Optimizer in the presence of all valid inequalities that are proposed in Section 3.2. For comparative purposes, we also solved the same set of instances through the matheuristic with the three-index formulation using the same set of parameters that was described in the previous section. The detailed results are available in Table D.6 (refer to Appendix D).

Noteworthy, (Murray and Chu, 2015) report that they were unable to solve any of these problems to proven optimality within 30 min of runtime. Compared to this, within 15 min of runtime, we can solve 26 out of 72 instances to optimality. The average MIP gap over all instances remained 16.69% after runtime. As we observed in Section 5.1.1, an increased endurance (i.e., flight time $\mathcal{E}_t$) generally decreases the performance of the MILP solver. This is also evident by the average runtime of 535.0 s and MIP gap of 11.4% (781.9 s and 22.0%) after runtime when the maximum flight time is set to 20 min (40 min).

Table 3 provides some additional information based on the solutions returned by the MILP solver. More precisely, this table contains the average savings Δ (refer to Formula (80)) and the average number of drone operations present in the solution vector. Moreover, information on the root relaxation solution is shown (similar to Table 2). This table highlights that slightly smaller savings of a similar magnitude are possible on these instances compared to the ones observed in Section 5.1.1. Furthermore, the dependence



(a) $|\mathcal{K}| = 1, |\mathcal{D}| = 1$ (b) $|\mathcal{K}| = 2, |\mathcal{D}| = 1$

(c) $|\mathcal{K}| = 1, |\mathcal{D}| = 2$ (d) $|\mathcal{K}| = 2, |\mathcal{D}| = 2$

Fig. 9. The relative improvement Δ as a box plot depending on the number of vehicles $|\mathcal{K}|$, the number of drones per vehicle $|\mathcal{D}|$, the relative endurance $\mathcal{E}_r$, and the relative velocity α. Values taken from Table D.3 (averaged over 10 instances).

Table 3
The savings Δ and the number of drone operations present in the MILP solution vectors to the instances of Murray and Chu (2015) as well as information on the root relaxation solution (averaged over 12 instances).

Drone Speed (mph)	$\mathcal{E}_t = 20$ min						$\mathcal{E}_t = 40$ min					
	Solution			Root			Solution			Root		
	Δ	MIP Gap	Operations	Iterations	Time	$\%_R$	Δ	MIP Gap	Operations	Iterations	Time	$\%_R$
15	2.7%	0.0%	0.7	1,028.5	0.4	58.0%	10.2%	22.0%	1.0	1,123.8	1.4	54.6%
25	15.7%	17.2%	2.1	1,174.8	1.1	52.8%	17.6%	23.9%	2.2	1,298.6	1.7	53.9%
35	23.0%	16.9%	2.9	1,400.7	1.7	54.0%	22.8%	20.3%	3.0	1,417.6	1.8	53.8%
Average	13.8%	11.4%	1.9	1,201.3	1.1	54.9%	16.8%	22.0%	2.1	1,280.0	1.6	54.1%

of the savings on the drone velocity and endurance structurally matches the one observed in Fig. 7.

With respect to the *best known solutions* (BKS) provided by Murray and Chu (2015) and Ha et al. (2018), both our MILP and matheuristic showed favorable performance. More precisely, both approaches were able to generate solutions that were on average within a 0.5% margin to the BKS. Moreover, to the best of our knowledge, we believe that we were able to identify several new and slightly improved BKS through the MILP solver and matheuristic on 13 instances (refer to Table D.6).

5.2. Large instances

Regarding the large instances, we also used the instances provided by Agatz et al. (2018) with 20, 50, and 100 vertices, respectively. In accordance with the explanations in Section 5.1.1, for each size, there are ten instances that are associated with a relative velocity $\alpha \in \{1, 2, 3\}$ and a relative endurance $\mathcal{E}_r \in \{0.2, 0.4, 0.6, 1.0, 1.5, 2.0\}$. For our experiments, we limit ourselves to $\mathcal{E}_r \leqslant 1$ as no significant differences can be observed at larger endurances (see the results in Section 5.1.1). Furthermore, we test with $|\mathcal{K}| \in \{1, 2, 3\}$ and $|\mathcal{D}| \in \{1, 2\}$, i.e., we have one, two, or three trucks that are equipped with either one or two drones each. This yields a total of $3 \cdot 10 \cdot 3 \cdot 4 \cdot 3 \cdot 2 = 2160$ instances. On each instance, we run the matheuristic 5 times using either the three- or two-index formulation of the DASP, yielding a total of 21,600 unique experiments.

In order to achieve an improved runtime behavior, in particular when solving instances with 100 customers while a single truck is present, we use the DASP partition (see Algorithm 2). To this end, a route containing more than 20 vertices is split into multiple (equally large) partitions such that no partition contains more than 20 vertices.

As we have no optimal objective values for these problems, we introduce the following updated metric Δ for studying the benefit of using drones. Given a fixed number of trucks, the updated metric gives insights into the change in makespan through the utilization of drones:

$$\Delta := 100\% - \frac{\text{incumbent objective value}}{\text{incumbent objective value before solving the DASP}} \tag{82}$$

Tables D.7–D.12 show the results from solving the instances with 20, 50, and 100 vertices by using the matheuristic with either the three- or two-index formulation for solving the DASP. Overall, on larger instances the matheuristic also converges very fast. Whereas preliminary results have shown that the three-index formulation typically performs better in solving larger DASPs, when using the DASP partition, there is no significant performance difference in using the three- or two-index formulation for solving the MILPs within the heuristic. Furthermore, similar observations regarding the share of acyclic and cyclic operations can be made, i.e., when the relative velocity is small ($\alpha \in \{1, 2\}$), then the share of cyclic operations also tends to be very small irrespective of the number of drones. Apart from this observation, there is no significant increase in the number of acyclic operations performed when moving from $\alpha = 2$ to $\alpha = 3$.

Fig. 10 visualizes the relative improvement Δ (according to Formula (82)) depending on the number of trucks, drones per truck, instance size, relative velocity, and endurance, respectively, based on the results achieved through the three-index formulation. The basic shape of these curves is very similar to the one that we observed in Fig. 7. That is, on the one hand, increasing the relative endurance $\mathcal{E}_r$ quickly leads to a threshold where a further increase does not yield any changes to the relative improvement. In particular, based on the definition of the relative endurance in Formula (79), for a fixed relative endurance, the larger the instance, the larger the number of possible operations. Therefore, we can observe a lower threshold for larger (or more dense) instances. Furthermore, on the other hand, an increase in the relative velocity leads to a greater relative improvement in all cases. We can also observe that the smaller the number of trucks (or, the larger the number of customers served by each truck) the larger the relative improvement gained by increasing α. These improvements heavily depend on the number of drones. In most cases, the improvement gained by increasing α from 2 to 3 is significantly larger in the presence of two drones than in the presence of a single drone per truck. Finally, we can also observe that substitutability is possible. Therefore, in some cases, a single fast moving drone (e.g., $\alpha = 3$) allows for a similar relative improvement as two slower drones (e.g., $\alpha = 2$ with possibly reduced endurance).

5.3. Comparison to theoretical bounds

As we have discussed in Section 2, Wang et al. (2017) provided several theoretical insights on the VRPD. In this section, we are interested in investigating these theoretical bounds and comparing them to the numerical results of our study. In particular, we are interested in a bound that concerns the relative completion ratio Z_r.

Let $Z(\text{VRPD}_{|\mathcal{K}|,\alpha,|\mathcal{D}|})$ be the optimal objective value of the VRPD for a fleet consisting of $|\mathcal{K}|$ trucks equipped with $|\mathcal{D}|$ drones each. Each drone is assumed to move at a relative velocity α (with unlimited endurance, i.e., $\mathcal{E}_r = 2$). In addition, we denote the optimal objective value by $Z(\text{VRP}^*)$, if we use the same fleet, but with no drones. Wang et al. have introduced the following bound:

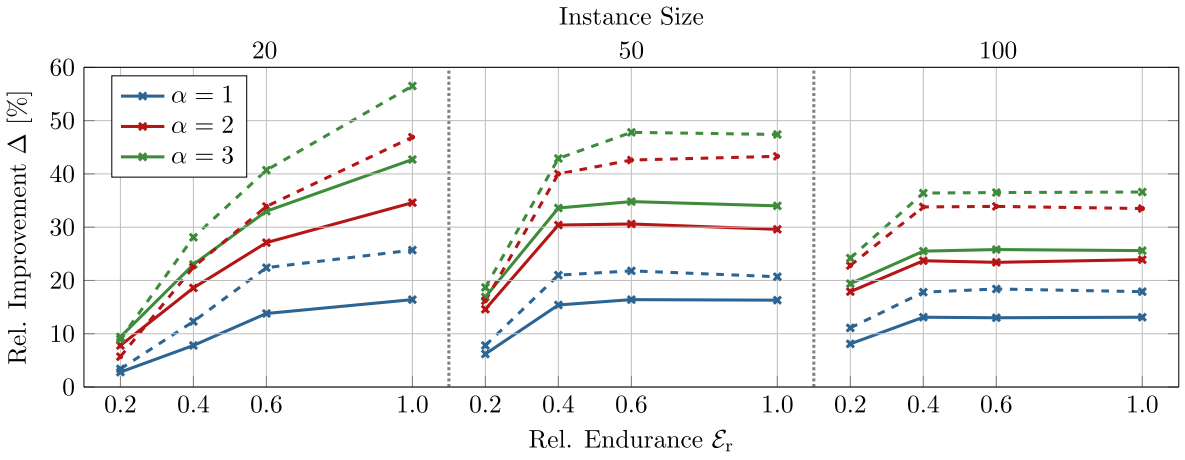
$$Z_{r,\max} = \frac{Z(\text{VRP}^*)}{Z(\text{VRPD}_{|\mathcal{K}|,\alpha,|\mathcal{D}|})} \leqslant \alpha \left|\mathcal{D}\right| + 1. \tag{83}$$

Table 4 highlights the investigated cases and the corresponding results. Depending on the number of drones and their relative velocity, the table shows the bound on the relative completion ratio ($Z_{r,\max}$), the analogous maximum possible savings $\Delta_{\max}$ (see Eq. (80)), and the average savings observed for one and two trucks, respectively, based on Table D.3 for $\mathcal{E}_r = 2$.

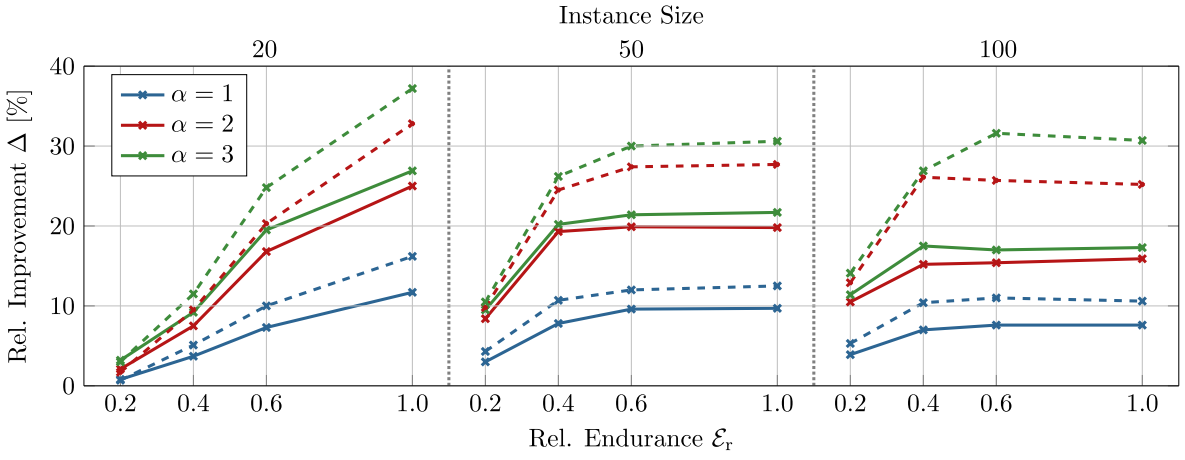
Overall, the savings (or completion ratios) that we observed in our numerical study are significantly smaller than the maximum possible theoretical savings. These observations apply to both, the small instances (see Section 5.1) as well as heuristic solutions to large instances (see Section 5.2).

6. Conclusion

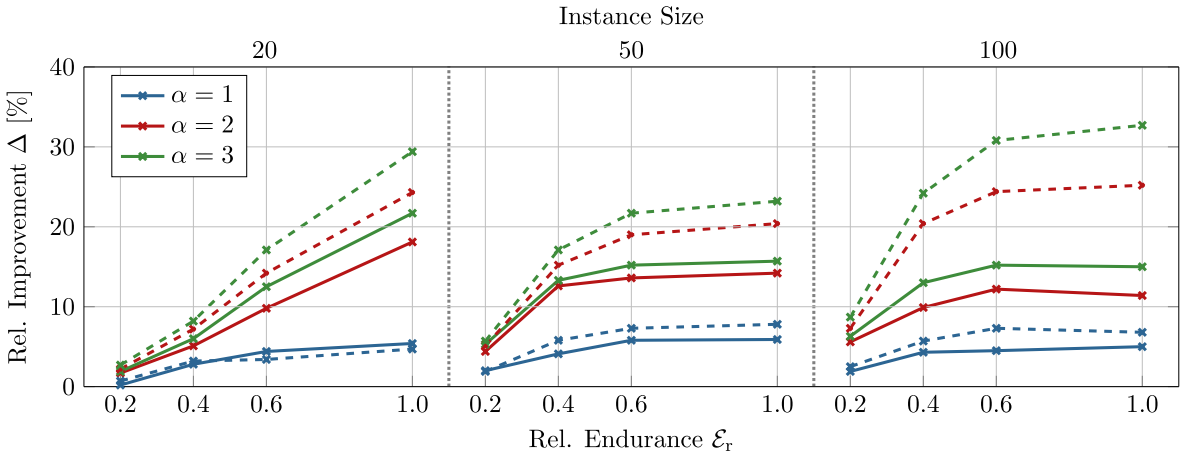
In this paper, we investigated the VRPD. To be more precise, we began in Section 2 with a brief overview of related problems in



(a) $|\mathcal{K}| = 1$ ($|\mathcal{D}| = 1$ as solid lines, $|\mathcal{D}| = 2$ as dashed lines)



(b) $|\mathcal{K}| = 2$ ($|\mathcal{D}| = 1$ as solid lines, $|\mathcal{D}| = 2$ as dashed lines)



(c) $|\mathcal{K}| = 3$ ($|\mathcal{D}| = 1$ as solid lines, $|\mathcal{D}| = 2$ as dashed lines)

Fig. 10. The relative improvement Δ depending on the instance size, number of vehicles $|\mathcal{K}|$, number of drones per vehicle $|\mathcal{D}|$, relative endurance $\mathcal{E}_r$, and relative velocity α based on Tables D.7, D.8, D.9.

Table 4
Theoretical bounds for the cases investigated in this work based on Wang et al. (2017).

$\lvert\mathcal{D}\rvert$	–	1			2		
α	0	1	2	3	1	2	3
$Z_{r,\max} \leqslant$	1	2	3	4	3	5	7
$\Delta_{\max} = 100\% - \frac{1}{Z_r} \leqslant$	0.0%	50.0%	66.66%	75.0%	66.66%	80.0%	85.7%
$\Delta_{\text{avg},\lvert\mathcal{K}=1\rvert}$	0.0%	10.8%	27.5%	35.3%	14.9%	41.2%	55.7%
$\Delta_{\text{avg},\lvert\mathcal{K}=2\rvert}$	0.0%	3.9%	24.1%	40.8%	6.1%	43.7%	58.9%

last-mile delivery that feature trucks and drones which have intensive synchronization requirements. Afterwards, in Section 3 we provided a MILP formulation of the VRPD and derived several valid inequalities. In Section 4, for solving the VRPD, we introduced a matheuristic that can effectively leverage any MILP solver by exploiting the structure of the VRPD. As part of this matheuristic, we introduced the DASP that, given a feasible routing, finds an optimal assignment and schedule of drones such that the makespan is minimized. In this context, we provided two different MILP formulations for the DASP. Finally, in Section 5, we presented the results and observations of our extensive computational experiments. The numerical results indicate that the use of drones can significantly reduce the makespan. Furthermore, we have shown that similar makespan reductions can be achieved with different drone designs. According to the numerical results, our matheuristic is very fast and able to consistently provide optimal or near optimal solutions.

Throughout this work, we have hinted at several potential opportunities for future research. In particular, we have shown that different recharge policies or a mixed drone fleet might be of future research interest (see Appendix A). Furthermore, even though our matheuristic performed very well overall, we believe that it might be interesting to embed it as a component within a more sophisticated metaheuristic. Finally, it might be worthwhile to investigate the existence of more compact MILP formulations.

Acknowledgments

We would like to thank the anonymous reviewers for their insightful comments and suggestions that helped us in improving the quality of this paper.

Appendix A. Possible variations

As we have seen in Section 2, the literature on optimization problems with drones in the context of last-mile delivery is vast and fast growing. Moreover, even though the basic assumptions of these problems are commonly accepted by most researchers, there is no general consensus on more detailed aspects of the problem. These aspects concern questions such as whether cyclic operations are allowed and whether trucks as well as drones follow the same distance metric. In particular, while waiting for a truck to arrive, a relevant question is whether drones can safely land and remain stationary on the ground or if they must continue hovering. This has different implications for distance and time constrained endurance (refer to Appendix C). With the purpose of illustrating the flexibility of the proposed model, as we will see in Appendix A.1–A.6, our model can be easily adapted to cases where different assumptions might apply.

A.1. Limited flight distance without recharge

In this section, we show that it is possible to consider the case where a drone has a limited range of operation and cannot be recharged. In the MILP (1)–(27), this can be done by replacing constraints (24) with the following constraints that account for the total distance covered during all operations for each drone:

$$\sum_{i \in V_L} \sum_{\substack{w \in V_D, \\ i \neq w}} \sum_{\substack{j \in V, \\ w \neq j}} \left(\bar{d}_{iw} + \bar{d}_{wj} \right) y_{iwj}^{kd} \leqslant \mathcal{E}: \forall\, k \in \mathcal{K},\, d \in \mathcal{D}. \tag{A.1}$$

A.2. Limited flight time with instantaneous recharge

If a drone cannot safely remain stationary on its own while waiting for the arrival of a truck, we might consider a maximum flight time including hovering time. To this end, the endurance (in time units) might be limited by a maximum flight time $\mathcal{E}_t$, rather than distance. If we assume that the endurance is recharged after each operation (e.g., through a swap of the battery), we can introduce the following sets of inequalities as a replacement for constraints (24) that consider the maximum flight time $\mathcal{E}_t$ instead:

$$M\left(y_{iwj}^{kd}-1\right)+\left(r_j^{kd}-s_i^{kd}\right)\leqslant \mathcal{E}_t:\quad \forall\, k\in\mathcal{K},\, d\in\mathcal{D},\, i\in V_L,\, w\in V_D,\, j\in V_R,\, \left|\left\{i,w,j\right\}\right|=3, \tag{A.2}$$

$$(\bar{t}_l+\bar{t}_{iw}+\bar{t}_{wi}+\bar{t}_r)\,y_{iwi}^{kd}\leqslant \mathcal{E}_t:\quad \forall\, k\in\mathcal{K},\, d\in\mathcal{D},\, i\in V_L,\, w\in V_D,\, i\neq w. \tag{A.3}$$

In this case, the set of inequalities in (A.2) and (A.3) account for the time spent performing acyclic and cyclic operations, respectively. In particular, in the case of acyclic operations, the hovering time is also considered if the drone arrives before the truck. Note that constraints (A.2) imply that a drone must *immediately* depart from the customer. If a drone can safely remain stationary at the place of delivery for some time, the case discussed in Appendix C applies.

A.3. Limited flight time without recharge

Analogously, we might consider a case where the drone has a limited flight time and an endurance that cannot be recharged. In this case, the following (non-linear) set of constraints might be introduced instead of constraints (24):

$$\sum_{i\in V_L}\sum_{\substack{w\in V_D,\\ i\neq w}}\left(\bar{t}_l+\bar{t}_{iw}+\bar{t}_{wi}+\bar{t}_r\right)y_{iwi}^{kd}+\sum_{i\in V_L}\sum_{\substack{w\in V_D,\\ i\neq w}}\sum_{\substack{j\in V_R,\\ i\neq j,\\ w\neq j}}\left(r_j^{kd}-s_i^{kd}\right)y_{iwj}^{kd}\leqslant \mathcal{E}_t:\ \forall\, k\in\mathcal{K},\, d\in\mathcal{D}. \tag{A.4}$$

These constraints guarantee that the total time spent performing operations (including hovering time) must be less than or equal to the endurance $\mathcal{E}_t$. As an alternative, to provide a linear model, we might introduce additional continuous decision variables q_i^{kd} and $\bar{q}_i^{kd}$ where $k\in\mathcal{K}$, $d\in\mathcal{D}$, $i\in V$ that can take a value in $[0,\,\mathcal{E}_t]$. The variables q_i^{kd} and $\bar{q}_i^{kd}$ indicate the remaining flight time of drone d (that belongs to truck k) when initially retrieved and after departing from vertex i, respectively (we assume that $q_0^{kd}=\mathcal{E}_t$). With these additional variables, we might use the following sets of inequalities instead of constraints (A.4) to provide a linear model:

$$M\left(1-x_{ij}^{kd}\right)+\bar{q}_i^{kd}\geqslant q_j^{kd}:\quad \forall\, k\in\mathcal{K},\, d\in\mathcal{D},\, i\in V_L,\, j\in V_R,\, i\neq j, \tag{A.5}$$

$$q_i^{kd}-\sum_{\substack{w\in V_D,\\ i\neq w}}\left(\bar{t}_l+\bar{t}_{iw}+\bar{t}_{wi}+\bar{t}_r\right)y_{iwi}^{kd}\geqslant \bar{q}_i^{kd}:\quad \forall\, k\in\mathcal{K},\, d\in\mathcal{D},\, i\in V_L, \tag{A.6}$$

$$M\left(1-y_{iwj}^{kd}\right)+\bar{q}_i^{kd}-\left(r_j^{kd}-s_i^{kd}\right)\geqslant q_{jr}^{kd}:\quad \forall\, k\in\mathcal{K},\, d\in\mathcal{D},\, i\in V_L,\, w\in V_D,\, j\in V_R,\, \left|\left\{i,w,j\right\}\right|=3. \tag{A.7}$$

Constraints (A.5) guarantee that for any transition x_{ij}^k that is performed by a truck, the remaining endurance of drone d upon arrival at j must be less than or equal to the remaining endurance upon departure at i. In between arrival and departure, further charge might be used to perform acyclic operations which are accounted for by the set of constraints (A.6). Finally, the inequalities in (A.7) account for a depletion of the drone's battery for performing acyclic operations.

A.4. Limited flight time with non-instantaneous recharge

Similar to the previous section, we might consider a scenario where a drone's battery can be recharged non-instantaneously. In this case, we assume that the battery of the drone might be recharged while the truck is powered (in motion). To this end, we might use the decision variables q_i^{kd} and $\bar{q}_i^{kd}$ that were introduced in the previous section and add the set of constraints (A.6) and (A.7) as well as the following additional constraint:

$$M\left(1-x_{ij}^{kd}\right)+\bar{q}_i^{kd}+\frac{t_{ij}}{\kappa}\left(y_i^{kd}-\sum_{\substack{w\in V_D,\\ i\neq w}}\sum_{\substack{h\in V_R,\\ i\neq h,\\ w\neq h}}y_{iwh}^{kd}\right)\geqslant q_i^{kd}:\ \forall\, k\in\mathcal{K},\, d\in\mathcal{D},\, i\in V_L,\, j\in V_R,\, i\neq j. \tag{A.8}$$

This constraint guarantees that for any transition x_{ij}^k performed by truck k where the drone d is available at i (i.e., $y_i^{kd}=1$) and performs no acyclic operation (i.e., the drone remains stationary on the truck during the transition to j), the endurance might be recharged with t_{ij}/κ, where κ is a parameter that determines the charging rate for each unit of time that the truck is in motion.

If we assume that a drone can also be recharged while a truck remains stationary, it is possible to introduce further continuous

decision variables h_i^{kd} with lower bound 0 that denote the additional waiting time of drone d that remains stationary on truck k at vertex i. Then, we might replace inequalities (21) with the following set of inequalities which guarantee that the drone departure time s_i^{kd} is bound by the release time r_i^{kd}, the time spent performing operations at i, and the waiting time h_i^{kd}.

$$r_i^{kd} + \sum_{\substack{w \in V_D, \\ i \neq w}} \left(\bar{t}_l + \bar{t}_{iw} + \bar{t}_{wi} + \bar{t}_r \right) y_{iwi}^{kd} + h_i^{kd} \leqslant s_i^{kd}: \quad \forall\, k \in \mathcal{K},\, d \in \mathcal{D},\, i \in V. \tag{A.9}$$

With these variables h_i^{kd}, we might replace constraints (A.6) with the following set of constraints that allow a drone to be recharged at a rate of κ' per unit of time that it remains on a stationary truck:

$$q_i^{kd} - \sum_{\substack{w \in V_D, \\ i \neq w}} \left(\bar{t}_l + \bar{t}_{iw} + \bar{t}_{wi} + \bar{t}_r \right) y_{iwi}^{kd} + \frac{h_i^{kd}}{\kappa'} \geqslant \bar{q}_i^{kd}: \quad \forall\, k \in \mathcal{K},\, d \in \mathcal{D},\, i \in V_L. \tag{A.10}$$

Note that similar considerations are easily possible if the endurance is limited by distance (i.e., excluding hovering time) rather than time.

A.5. Heterogeneous drone fleet

Moreover, since each drone is referenced by a specific index, it is easily possible to consider a mixed drone fleet where each truck is equipped with a different number of drones which might have different velocities and endurances. This can be done by introducing specific drone travel times $\bar{t}_{ij}^{kd}$ and endurances $\mathcal{E}^{kd}$.

A.6. Incorporating these variations into the DASP

As a concluding remark, note that the possible variations on MILP (1)–(27), that were implied in Appendix A.1–A.5, might be easily incorporated into either model for the DASP. All variations to the MILP require minor adjustments to constraints (21) and (24). Note that there is a direct correspondence between these constraints and constraints (47), (50), and (51) for the three-index formulation (refer to Section 4.2.1) and constraints (71), (74), and (75) for the two-index formulation (refer to Section 4.2.2). Moreover, most variations require the variables r_i^{kd} and s_i^{kd} from MILP (1)–(27). The equivalent variables are already present in both proposed DASP models as r_i^d and s_i^d. Any additional variables or parameters that were introduced in Appendix A.1–A.5 might be added to the DASP models analogously.

Appendix B provides a guideline on how all MILPs must be adjusted to respect the assumptions of Murray and Chu (2015). This appendix also provides an illustrative example on how, e.g., the endurance constraints must be adjusted to respect the maximum flight time.

Appendix B. Matching the problem formulation of Murray and Chu (2015)

In this section, we describe how the MILP formulations of the VRPD and DASP might be adjusted to respect the problem formulation of Murray and Chu (2015). As outlined in Section 2, several slightly deviating assumptions apply for this problem. More precisely, cyclic drone-operations are prohibited and the overhead times and associated temporal constraints are treated in a slightly different way. In particular, the endurance is measured in units of time rather than units of distance. For a formulation that follows the assumptions of Murray and Chu (2015), we propose MILP (1)–(19), (B.1)–(B.6), (25)–(27).

$$M\left(y_{iwj}^{kd} - 1\right) + s_i^{kd} + \bar{t}_{iw} \leqslant r_w^{kd}: \quad \forall\, k \in \mathcal{K},\, d \in \mathcal{D},\, i \in V_L,\, w \in V_D,\, j \in V_R,\, \left|\left\{i, w, j\right\}\right| = 3, \tag{B.1}$$

$$M\left(y_{iwj}^{kd} - 1\right) + s_w^{kd} + \bar{t}_{wj} + \bar{t}_r \leqslant r_j^{kd}: \quad \forall\, k \in \mathcal{K},\, d \in \mathcal{D},\, i \in V_L,\, w \in V_D,\, j \in V_R,\, \left|\left\{i, w, j\right\}\right| = 3, \tag{B.2}$$

$$r_i^{kd} \leqslant s_i^{kd}: \quad \forall\, k \in \mathcal{K},\, d \in \mathcal{D},\, i \in V, \tag{B.3}$$

$$s_i^{kd} \leqslant s_i^{k}: \quad \forall\, k \in \mathcal{K},\, d \in \mathcal{D},\, i \in V, \tag{B.4}$$

$$a_j^k + \bar{t}_l \sum_{\substack{w \in V_D, \\ w \neq j}} \sum_{\substack{h \in V_r \\ j \neq h, \\ w \neq h}} y_{jwh}^{kd} + \bar{t}_r \sum_{\substack{i \in V_L \\ i \neq j}} \sum_{\substack{w \in V_D, \\ i \neq w, \\ w \neq j}} y_{iwj}^{kd} \leqslant r_j^{kd}: \quad \forall\, k \in \mathcal{K},\, d \in \mathcal{D},\, j \in V_R, \tag{B.5}$$

$$M\left(y_{iwj}^{kd}-1\right)+\left(r_j^{kd}-s_w^{kd}\right)+\bar{t}_{iw}\leqslant\mathcal{E}_t:\quad\forall\,k\in\mathcal{K},\,d\in\mathcal{D},\,i\in V_L,\,w\in V_D,\,j\in V_R,\,\left|\left\{i,w,j\right\}\right|=3. \tag{B.6}$$

In this model, we ask that all variables y_{iwi} where $i\in V_L,\ \ w\in V_D$ are bound to 0 to prevent any cyclic operation from occurring. The updated temporal constraints (B.1)–(B.5) are consistent with the model of Murray and Chu (2015). In particular, the overhead times are treated in a slightly different way through constraints (B.1), (B.2), and (B.5). Finally, constraints (B.6) provide an endurance based on flight time that is coherent with the one proposed by Murray and Chu (2015). Similar adjustments can be made for the DASP models, where it is also necessary to bind all variables y_{iwi} and z_{iw} where $i\in V_L,\ \ w\in V_D$ to values 0. For the three-index formulation, we propose MILP (39)–(44), (B.7)–(B.12), (52)–(54).

$$-M\left(1-y_{iwj}^{d}\right)+s_i^d+\bar{t}_{iw}\leqslant r_w^d:\quad\forall\,d\in\mathcal{D},\,i\in V_L,\,w\in V_D,\,j\in V_R,\,i\prec w\prec j, \tag{B.7}$$

$$-M\left(1-y_{iwj}^{d}\right)+s_w^d+\bar{t}_{wj}+\bar{t}_r\leqslant r_j^d:\quad\forall\,d\in\mathcal{D},\,i\in V_L,\,w\in V_D,\,j\in V_R,\,i\prec w\prec j, \tag{B.8}$$

$$r_i^d\leqslant s_i^d:\quad\forall\,d\in\mathcal{D},\,i\in V, \tag{B.9}$$

$$s_i^d\leqslant s_i:\quad\forall\,d\in\mathcal{D},\,i\in V, \tag{B.10}$$

$$a_j+\bar{t}_l\sum_{\substack{w\in V_D,\\ j\prec w}}\ \sum_{\substack{g\in V_R,\\ w\prec g}}y_{jwg}^{d}+\bar{t}_r\sum_{\substack{i\in V_L,\\ i\prec w}}\ \sum_{\substack{w\in V_D,\\ w\prec j}}y_{iwj}^{d}\leqslant r_j^d:\quad\forall\,d\in\mathcal{D},\,j\in V_R, \tag{B.11}$$

$$M\left(y_{iwj}^{d}-1\right)+\left(r_j^d-s_w^d\right)+\bar{t}_{iw}\leqslant\mathcal{E}_t:\quad\forall\,k\in\mathcal{K},\,d\in\mathcal{D},\,i\in V_L,\,w\in V_D,\,j\in V_R,\,i\prec w\prec j. \tag{B.12}$$

Analogously, for the two-index formulation, we propose MILP (61)–(68), (B.13)–(B.18), (76)–(78)

$$-M\left(1-y_{iw}^{d}\right)+s_i^d+\bar{t}_{iw}\leqslant r_w^d:\quad\forall\,d\in\mathcal{D},\,i\in V_L,\,w\in V_D,\,i\prec w, \tag{B.13}$$

$$-M\left(1-\widetilde{y}_{wj}^{d}\right)+s_w^d+\bar{t}_{wj}+\bar{t}_r\leqslant r_j^d:\quad\forall\,d\in\mathcal{D},\,w\in V_D,\,j\in V_R,\,i\prec w, \tag{B.14}$$

$$r_i^d\leqslant s_i^d:\quad\forall\,d\in\mathcal{D},\,i\in V, \tag{B.15}$$

$$s_i^d\leqslant s_i:\quad\forall\,d\in\mathcal{D},\,i\in V, \tag{B.16}$$

$$a_j+\bar{t}_l\sum_{\substack{w\in V_D,\\ j\prec w}}y_{jw}^{d}+\bar{t}_r\sum_{\substack{w\in V_D,\\ w\prec j}}\widetilde{y}_{wj}^{d}\leqslant r_j^d:\quad\forall\,d\in\mathcal{D},\,j\in V_R, \tag{B.17}$$

$$M\left(\widetilde{y}_{wj}^{d}-1\right)+\left(r_j^d-s_w^d\right)+\sum_{\substack{w\in V_D,\\ i\prec w}}\left(\bar{t}_{iw}y_{iw}^{d}\right)\leqslant\mathcal{E}_t:\quad\forall\,k\in\mathcal{K},\,d\in\mathcal{D},\,i\in V_L,\,w\in V_D,\,j\in V_R,\,i\prec w\prec j. \tag{B.18}$$

Appendix C. Endurance: limited flight distance vs. limited flight time

Within the research community, there is no consensus on the operational constraints that should apply to drones. In particular, this concerns the possibility of *cyclic* drone operations (refer to Table 1) and the way the *endurance* of drones is modelled. Some suggest the application of range-constrained drones (see, e.g., Agatz et al., 2018; Di Puglia Pugliese and Guerriero, 2017), while others propose time-constrained drones instead (see, e.g., Murray and Chu, 2015; Ha et al., 2018). In order to bridge this gap, we introduce the following proposition, which states that there is no problematic difference between the two mentioned cases, if the drone is permitted to delay its departure after serving a customer. Before attending to the proposition, we introduce the following constraints, which can be used if a drone must not immediately depart after serving a customer:

$$M\left(y_{iwj}^{kd}-1\right)+s_i^{kd}+\left(\bar{t}_l+\bar{t}_{iw}\right)\leqslant r_w^{kd}:\quad\forall\,k\in\mathcal{K},\,d\in\mathcal{D},\,i\in V_L,\,w\in V_D,\,j\in V_R,\,\left|\left\{i,w,j\right\}\right|=3, \tag{C.1}$$

$$M\left(y_{iwj}^{kd}-1\right)+s_w^{kd}+\left(\bar{t}_{wj}+\bar{t}_r\right)\leqslant r_j^{kd}:\quad\forall\,k\in\mathcal{K},\,d\in\mathcal{D},\,i\in V_L,\,w\in V_D,\,j\in V_R,\,\left|\left\{i,w,j\right\}\right|=3, \tag{C.2}$$

$$M\left(y_{iwj}^{kd}-1\right)+\bar{t}_{l}+\bar{t}_{iw}+\left(r_{j}^{kd}-s_{w}^{kd}\right)\leqslant\mathcal{E}_{t}:\quad\forall\,k\in\mathcal{K},\,d\in\mathcal{D},\,i\in V_{L},\,w\in V_{D},\,j\in V_{R},\,\left|\left\{i,\,w,\,j\right\}\right|=3,\tag{C.3}$$

$$(\bar{t}_{l}+\bar{t}_{iw}+\bar{t}_{wi}+\bar{t}_{r})\,y_{iwi}^{kd}\leqslant\mathcal{E}_{t}:\quad\forall\,k\in\mathcal{K},\,d\in\mathcal{D},\,i\in V_{L},\,w\in V_{D},\,i\neq w.\tag{C.4}$$

More precisely, constraints (C.1) and (C.2) might serve as a replacement for constraints (20). Through these constraints, we explicitly account for the arrival time r_w^{kd} of a drone d at $w \in V_D$ and its earliest possible release time r_j^{kd} after its return flight towards $j \in V_R$. Given a maximum flight time $\mathcal{E}_t$, the endurance constraints are enforced by (C.3) and (C.4) for acyclic and cyclic operations, respectively, and might serve as a replacement for constraints (24). Note that a similar constraints were also used by Murray and Chu (2015) (refer to Appendix B). Due to constraints (21), in principle, the departure time from w, i.e., s_w^{kd}, can be arbitrarily larger than r_w^{kd}. In the following proposition, we address the relation between distance- and time-constrained endurance assumptions.

Proposition 5. *If a drone can remain stationary at a customer location for a sufficient amount of time, then distance- and time-constrained endurance assumptions are equivalent.*

Proof. Assume that a drone d is retrieved at some vertex j by truck k after an operation (i, w, j). We distinguish two cases: (i) either the drone arrives at the same time as (or after) the truck, (ii) or the drone arrives before the truck.

First, we assume that the drone arrives at the same time as or after the truck, i.e., $s_w^{kd} + \bar{t}_{wj} + \bar{t}_r \geqslant a_j^k + \bar{t}_r$. This case is not problematic. The reason is the fact that, as the truck is already at vertex j, the drone does not need to hover. More precisely, to perform the operation (i, w, j), the drone is in flight for exactly the following amount of time units:

$$(\bar{t}_{l}+\bar{t}_{iw})+(r_{j}^{kd}-s_{w}^{kd})=(\bar{t}_{l}+\bar{t}_{iw}+\bar{t}_{wj}+\bar{t}_{r}).$$

Given its average velocity $\bar{v}$, the drone covers $(\bar{t}_l + \bar{t}_{iw} + \bar{t}_{wj} + \bar{t}_r)\cdot\bar{v} = \bar{d}_{iw} + \bar{d}_{wj}$ distance units during the operation.

Next, we assume that the drone arrives before the truck, i.e., $s_w^{kd} + \bar{t}_{wj} + \bar{t}_r < a_j^k + \bar{t}_r$. Here, if a drone can not safely land, it would need to continue hovering for a time $\delta > 0$ that satisfies the following condition:

$$s_{w}^{kd}+\bar{t}_{wj}+\bar{t}_{r}+\delta=a_{j}^{k}+\bar{t}_{r}.$$

Hovering for δ time units would require a continuous power supply approximately equal to that during straight flight (Dorling et al., 2017). However, the return flight from w can simply be delayed such that the expected arrival time of the drone and truck match. More precisely, it is always feasible to delay the departure time s_w^{kd} from w by the amount δ such that the arrival time of the truck is matched and hovering becomes unnecessary, i.e., $s_w^{kd} + \bar{t}_{wj} + \bar{t}_r = a_j^k + \bar{t}_r$. Note that the makespan remains unaffected. Consequently, analogously to the first case, the drone is in flight for exactly $(\bar{t}_l + \bar{t}_{iw} + \bar{t}_{wj} + \bar{t}_r)$ time units and covers $(\bar{t}_l + \bar{t}_{iw} + \bar{t}_{wj} + \bar{t}_r)\cdot\bar{v} = \bar{d}_{iw} + \bar{d}_{wj}$ distance units during flight.

As a result, if we set $\mathcal{E}_t\cdot\bar{v} \leftarrow \mathcal{E}$ as well as:

$$(\bar{t}_{l}+\bar{t}_{iw})\cdot\bar{v}\leftarrow\bar{d}_{iw}\text{ and }(r_{j}^{kd}-s_{w}^{kd})\cdot\bar{v}=(\bar{t}_{wj}+\bar{t}_{r})\cdot\bar{v}\leftarrow\bar{d}_{wj}.$$

After integrating these transformations into (C.3) and (C.4), we obtain the following inequalities:

$$M\left(y_{iwj}^{kd}-1\right)+\bar{d}_{iw}+\bar{d}_{wj}\leqslant\mathcal{E}:\quad\forall\,k\in\mathcal{K},\,d\in\mathcal{D},\,i\in V_{L},\,w\in V_{D},\,j\in V_{R},\,\left|\left\{i,\,w,\,j\right\}\right|=3,\tag{C.5}$$

$$(\bar{d}_{iw}+\bar{d}_{wi})\,y_{iwi}^{kd}\leqslant\mathcal{E}:\quad\forall\,k\in\mathcal{K},\,d\in\mathcal{D},\,i\in V_{L},\,w\in V_{D},\,i\neq w.\tag{C.6}$$

Note that these inequalities are functionally equivalent to (24), which is the desired conclusion of Proposition 5. □

Remark: Interestingly, the concept of drones that may wait at customer locations is already implicitly present in some works that consider time-constrained drones (see, e.g., Murray and Chu, 2015).

Appendix D. Detailed numerical results

Table D.1 contains the results from solving small instances with 10 customers through Gurobi Optimizer 8.1.0 by using MILP (1)–(27). Moreover, Tables D.2 shows the results when adding VIEQ (30) and (32) to the model. Finally, in Table D.3, the symmetry breaking (34) and knapsack inequalities (37) and (38) were added to the formulation. Each table shows the relative savings Δ (see, Eq. (80)), the MIP gap, the runtime t, the number of acyclic and cyclic operations, #a and #c, respectively, as average values as well as the number of cases in which an optimal solution was found within the runtime limit.

Next, Tables D.4 and D.5 show the results from solving the small instances through the matheuristic (see Algorithm 1) using the three-index formulation (MILP (39)–(54)) and two-index formulation (MILP (61)–(78)) for solving the mathematical program, respectively. In these tables, the relative savings Δ, the time until the final solution is found (t_{best}), the number of changes to the incumbent solution n, and the number of operations are shown as average values. Furthermore, the column labeled "$\leqslant$" highlights how often the matheuristic found a solution with a better or equal objective value to the feasible (in some cases optimal) solutions returned by the MILP solver after runtime (refer to Table D.3). As the matheuristic was run for five times for each parameter vector, therefore, for each optimal solution in Table D.3, the matheuristic might reach this solution up to five times.

The results to the instances of Murray and Chu (2015) are shown in Table D.6. In this table, the *best known solutions* (BKS) were

taken from Murray and Chu (2015) and Ha et al. (2018). The objective value of the MILP and the best solution value after five runs of the matheuristic are then shown relative to the BKS. These results were obtained by adjusting all MILP formulations according to Appendix B.

Finally, Tables D.7–D.12 show the results achieved by solving instances with 20, 50, and 100 customers through matheuristic by solving the MILP through the three- and two-index formulation, respectively. In these tables, as we have no information on any optimal values, the Δ columns are calculated according to Eq. (82). Furthermore, the share of acyclic and cyclic operations are shown as percentages, rather than absolute values.

Table D.1
MILP on instances with 10 vertices.

$\lvert\mathcal{D}\rvert$	α	$\mathcal{E}_r$	$\lvert\mathcal{K}\rvert = 1$						$\lvert\mathcal{K}\rvert = 2$					
			Δ	Gap	t	#a	#c	Opt.	Δ	Gap	t	#a	#c	Opt.
1	1	0.2	0.1%	10.0%	315.7	0.2	0.0	9	0.1%	49.7%	687.1	0.2	0.3	5
		0.4	1.5%	70.0%	779.5	0.8	0.0	3	0.3%	90.0%	881.6	1.1	0.0	1
		0.6	2.6%	100.0%	900.4	2.1	0.0	0	1.2%	100.0%	900.3	1.7	0.0	0
		1	3.1%	100.0%	900.4	2.2	0.0	0	−1.3%	100.0%	901.3	1.1	0.2	0
		1.5	−1.2%	100.0%	900.3	1.1	0.0	0	−0.3%	100.0%	911.2	1.9	0.1	0
		2	−2.4%	100.0%	901.1	1.2	0.1	0	−0.7%	100.0%	903.7	1.5	0.1	0
	2	0.2	0.9%	0.0%	337.7	0.4	0.5	10	0.2%	40.0%	635.9	0.1	0.5	6
		0.4	3.5%	57.1%	730.1	1.9	0.2	4	1.0%	100.0%	900.8	1.4	0.5	0
		0.6	9.5%	96.2%	900.3	3.5	0.1	0	4.6%	100.0%	900.6	1.8	1.1	0
		1	14.1%	100.0%	901.0	4.0	0.6	0	7.2%	100.0%	900.3	2.4	2.5	0
		1.5	15.8%	100.0%	902.4	3.2	0.7	0	20.6%	100.0%	904.4	3.4	1.8	0
		2	14.9%	100.0%	900.2	3.0	1.1	0	20.0%	100.0%	911.7	3.6	1.5	0
	3	0.2	1.4%	0.0%	291.6	0.4	0.6	10	0.3%	41.7%	676.1	0.1	0.5	5
		0.4	4.2%	80.0%	808.4	1.8	0.6	2	1.6%	90.3%	900.7	1.0	0.7	0
		0.6	10.8%	100.0%	900.4	3.4	0.1	0	6.3%	100.0%	900.4	2.0	2.1	0
		1	19.6%	100.0%	900.3	3.7	1.5	0	15.4%	100.0%	901.5	2.6	2.7	0
		1.5	24.8%	100.0%	903.4	3.5	1.8	0	34.3%	100.0%	912.2	3.2	2.8	0
		2	25.1%	100.0%	900.3	3.7	1.1	0	35.8%	100.0%	909.4	3.3	2.7	0
	Average		8.2%	78.5%	781.9	2.2	0.5	38/180	8.1%	89.5%	863.3	1.8	1.1	17/180
1	1	0.2	0.1%	20.0%	371.9	0.4	0.0	8	0.0%	60.0%	752.2	0.2	0.2	4
		0.4	1.7%	80.0%	831.6	1.4	0.0	2	0.3%	100.0%	901.3	1.2	0.1	0
		0.6	3.3%	100.0%	900.4	3.1	0.0	0	0.3%	100.0%	901.6	1.6	0.6	0
		1	4.3%	100.0%	901.2	3.9	0.7	0	1.6%	100.0%	906.5	2.7	1.5	0
		1.5	6.7%	100.0%	900.3	4.1	0.3	0	4.3%	100.0%	900.9	3.5	2.5	0
		2	2.6%	100.0%	904.1	3.9	0.8	0	4.9%	100.0%	900.8	3.5	2.3	0
	2	0.2	0.9%	18.6%	391.1	0.4	0.5	8	0.2%	38.7%	620.7	0.1	0.5	6
		0.4	3.7%	76.4%	824.7	2.3	0.2	2	1.1%	86.1%	873.9	1.4	1.0	1
		0.6	10.1%	100.0%	900.4	4.1	0.5	0	4.3%	100.0%	900.5	2.2	1.7	0
		1	20.4%	100.0%	901.4	4.8	1.0	0	15.3%	100.0%	905.6	2.7	3.2	0
		1.5	30.8%	100.0%	900.3	4.6	2.2	0	31.5%	100.0%	901.1	3.8	3.1	0
		2	34.2%	100.0%	900.4	4.2	1.9	0	41.0%	100.0%	901.1	4.7	2.6	0
	3	0.2	1.4%	10.0%	358.3	0.4	0.6	9	0.3%	48.0%	720.3	0.3	0.6	5
		0.4	4.6%	90.0%	896.6	2.0	0.9	1	2.1%	91.9%	901.1	1.0	1.5	0
		0.6	12.2%	100.0%	900.4	3.9	0.7	0	6.3%	100.0%	902.4	1.4	3.3	0
		1	26.2%	100.0%	902.7	4.5	2.0	0	20.5%	100.0%	904.2	2.5	3.6	0
		1.5	46.1%	100.0%	902.2	4.4	2.5	0	44.7%	100.0%	901.3	3.5	3.4	0
		2	48.2%	100.0%	900.4	4.2	3.0	0	56.2%	100.0%	901.0	3.6	4.5	0
	Average		14.3%	83.1%	804.9	3.1	1.0	30/180	13.1%	90.3%	866.5	2.2	2.0	16/180

Table D.2
MILP + VIEQ (makespan bounds) on instances with 10 vertices.

$\|\mathcal{D}\|$	α	$\mathcal{E}_r$	$\|\mathcal{K}\| = 1$						$\|\mathcal{K}\| = 2$					
			Δ	Gap	t	#a	#c	Opt.	Δ	Gap	t	#a	#c	Opt.
1	1	0.2	0.1%	0.0%	1.8	0.3	0.0	10	0.1%	0.0%	29.7	0.3	0.2	10
		0.4	1.5%	0.0%	8.4	1.1	0.0	10	0.3%	0.0%	141.8	1.2	0.2	10
		0.6	3.4%	0.0%	123.7	2.4	0.0	10	1.4%	7.8%	561.8	1.8	0.1	8
		1	8.4%	33.3%	900.5	2.4	0.0	0	0.9%	57.8%	901.0	2.0	0.2	0
		1.5	10.9%	35.0%	901.9	2.0	0.0	0	3.0%	63.9%	904.4	2.0	0.3	0
		2	10.3%	38.2%	900.8	2.0	0.0	0	5.5%	62.6%	910.5	2.2	0.6	0
	2	0.2	0.9%	0.0%	1.6	0.4	0.5	10	0.2%	0.0%	29.3	0.2	0.6	10
		0.4	3.5%	0.0%	7.5	1.9	0.2	10	1.0%	0.0%	113.8	1.0	0.4	10
		0.6	9.5%	0.0%	41.5	3.5	0.1	10	4.8%	2.0%	445.4	2.6	0.7	9
		1	22.8%	3.5%	377.5	4.3	0.2	9	14.1%	58.0%	902.6	3.4	0.7	0
		1.5	26.2%	41.0%	901.7	3.8	0.2	0	23.8%	65.4%	903.4	4.3	0.7	0
		2	26.1%	39.4%	902.3	4.0	0.0	0	24.8%	63.3%	907.8	3.7	0.7	0
	3	0.2	1.4%	0.0%	2.4	0.4	0.6	10	0.3%	0.0%	35.4	0.3	0.4	10
		0.4	4.2%	0.0%	6.8	1.8	0.6	10	1.7%	0.0%	125.5	1.3	0.9	10
		0.6	11.0%	0.0%	34.1	3.5	0.1	10	6.3%	6.2%	554.4	2.3	1.5	7
		1	26.5%	0.0%	247.4	4.5	0.4	10	18.5%	56.8%	901.7	2.7	2.0	0
		1.5	35.0%	32.9%	875.3	4.1	0.5	1	35.2%	61.1%	841.0	3.8	1.7	1
		2	35.0%	35.5%	871.1	4.3	0.5	1	39.3%	62.3%	917.3	4.0	1.7	0
	Average		13.1%	14.4%	394.8	2.6	0.2	111/180	10.1%	31.5%	562.6	2.2	0.8	85/180
1	1	0.2	0.1%	0.0%	3.4	0.2	0.0	10	0.1%	0.0%	59.1	0.4	0.1	10
		0.4	1.7%	0.0%	49.5	1.7	0.0	10	0.3%	0.0%	299.2	1.1	0.0	10
		0.6	4.8%	1.1%	412.7	3.0	0.0	8	1.5%	30.6%	810.5	2.5	0.2	2
		1	11.9%	46.4%	903.6	3.7	0.0	0	3.3%	65.5%	911.6	2.4	1.2	0
		1.5	16.2%	51.0%	902.1	3.9	0.0	0	4.2%	74.0%	900.6	3.7	1.1	0
		2	13.4%	53.3%	900.3	3.9	0.0	0	5.7%	74.6%	900.5	4.5	1.0	0
	2	0.2	0.9%	0.0%	2.9	0.4	0.5	10	0.2%	0.0%	58.1	0.4	0.3	10
		0.4	3.7%	0.0%	13.4	2.2	0.3	10	1.2%	0.9%	274.7	1.6	0.8	9
		0.6	11.0%	0.0%	86.4	4.0	0.5	10	5.2%	23.2%	802.5	2.7	1.6	2
		1	27.8%	16.9%	778.3	5.4	0.5	3	15.3%	60.8%	901.0	3.5	1.2	0
		1.5	38.8%	50.0%	900.4	5.6	0.3	0	33.3%	73.8%	900.6	4.8	2.1	0
		2	40.5%	51.0%	900.4	5.5	0.4	0	42.1%	71.3%	900.7	4.8	2.3	0
	3	0.2	1.4%	0.0%	2.8	0.4	0.6	10	0.3%	0.0%	50.4	0.4	0.5	10
		0.4	4.6%	0.0%	14.6	2.1	0.8	10	2.1%	0.9%	238.3	1.4	1.3	9
		0.6	12.8%	0.0%	68.6	3.8	0.8	10	7.0%	26.2%	776.9	2.1	2.4	2
		1	33.5%	0.0%	486.5	5.6	0.5	10	22.0%	69.7%	859.4	3.7	1.8	1
		1.5	51.7%	42.9%	870.1	5.6	0.8	1	45.5%	74.1%	900.6	3.8	3.4	0
		2	53.9%	44.1%	884.5	5.0	1.8	1	59.9%	57.7%	900.5	5.1	2.7	0
	Average		18.3%	19.8%	454.5	3.4	0.4	103/180	13.8%	39.1%	635.8	2.7	1.3	65/180

Table D.3
MILP + VIEQ (makespan bounds, symmetry, knapsack) on instances with 10 vertices.

$\|\mathcal{D}\|$	α	$\mathcal{E}_r$	$\|\mathcal{K}\| = 1$						$\|\mathcal{K}\| = 2$					
			Δ	Gap	t	#a	#c	Opt.	Δ	Gap	t	#a	#c	Opt.
1	1	0.2	0.1%	0.0%	1.3	0.3	0.0	10	0.1%	0.0%	35.5	0.3	0.2	10
		0.4	1.5%	0.0%	4.0	1.0	0.0	10	0.3%	0.0%	125.9	1.0	0.2	10
		0.6	3.4%	0.0%	39.1	2.5	0.0	10	1.3%	7.1%	714.9	1.5	0.1	8
		1	9.5%	0.5%	625.7	2.3	0.0	9	1.8%	60.8%	900.3	1.9	0.2	0
		1.5	11.8%	33.5%	900.4	1.9	0.0	0	3.5%	63.9%	907.6	2.2	0.2	0
		2	10.8%	34.7%	900.2	1.9	0.0	0	3.9%	62.5%	926.3	1.9	0.0	0
	2	0.2	0.9%	0.0%	1.4	0.4	0.5	10	0.2%	0.0%	31.4	0.2	0.4	10
		0.4	3.5%	0.0%	3.2	1.9	0.2	10	1.0%	0.0%	138.3	1.0	0.8	10
		0.6	9.5%	0.0%	15.2	3.5	0.1	10	4.8%	10.3%	560.2	2.3	0.6	6
		1	22.8%	0.0%	210.1	4.2	0.2	10	14.5%	56.4%	870.8	3.1	1.0	1
		1.5	27.3%	27.4%	792.6	4.0	0.2	2	23.6%	64.1%	902.0	3.5	0.7	0
		2	27.5%	33.6%	900.5	3.7	0.4	0	24.1%	66.1%	916.8	3.5	0.8	0
	3	0.2	1.4%	0.0%	1.7	0.4	0.6	10	0.3%	0.0%	42.9	0.2	0.4	10
		0.4	4.2%	0.0%	3.3	1.8	0.6	10	1.7%	0.0%	139.8	1.3	0.8	10
		0.6	11.0%	0.0%	13.5	3.5	0.1	10	6.3%	6.9%	528.5	2.3	1.1	8
		1	26.2%	4.8%	168.3	4.3	0.6	9	17.4%	58.6%	857.6	2.6	2.1	1
		1.5	34.3%	33.7%	869.2	4.1	0.6	2	33.9%	67.4%	900.7	3.7	2.0	0
		2	35.3%	28.7%	790.7	4.0	0.7	2	40.8%	62.8%	915.4	3.7	1.9	0
	Average		13.4%	10.9%	346.7	2.5	0.3	124/180	10.0%	32.6%	578.6	2.0	0.8	84/180
1	1	0.2	0.1%	0.0%	1.9	0.4	0.0	10	0.1%	0.0%	66.5	0.4	0.0	10
		0.4	1.7%	0.0%	22.6	1.5	0.0	10	0.3%	0.0%	361.6	0.8	0.1	10
		0.6	4.8%	0.0%	154.3	3.1	0.0	10	1.4%	35.6%	900.8	2.2	0.2	0
		1	14.2%	39.9%	892.1	3.5	0.0	1	3.0%	66.7%	913.1	2.3	1.1	0
		1.5	14.3%	52.5%	900.6	3.9	0.0	0	5.7%	74.4%	900.7	3.9	1.2	0
		2	14.9%	53.3%	900.2	3.8	0.1	0	6.1%	74.6%	900.6	3.8	1.4	0
	2	0.2	0.9%	0.0%	1.7	0.4	0.5	10	0.2%	0.0%	56.3	0.3	0.3	10
		0.4	3.7%	0.0%	5.5	2.2	0.3	10	1.2%	1.2%	319.5	1.0	1.4	9
		0.6	11.0%	0.0%	24.4	4.1	0.4	10	4.9%	36.9%	797.1	2.0	1.9	2
		1	28.3%	25.2%	670.0	5.5	0.2	3	17.9%	59.6%	909.6	3.2	1.6	1
		1.5	39.0%	41.7%	900.3	5.5	0.1	0	34.4%	74.6%	900.6	3.9	2.5	0
		2	41.2%	46.1%	900.8	5.2	0.5	0	43.7%	70.2%	900.6	5.4	1.5	0
	3	0.2	1.4%	0.0%	1.7	0.4	0.6	10	0.3%	0.0%	69.5	0.1	0.7	10
		0.4	4.6%	0.0%	6.1	2.0	0.9	10	2.1%	0.0%	349.1	1.2	1.2	10
		0.6	12.8%	0.0%	30.6	3.7	0.9	10	6.9%	25.3%	730.1	2.1	2.0	4
		1	33.5%	4.3%	413.7	5.6	0.5	7	23.2%	73.6%	848.8	3.8	2.0	1
		1.5	52.7%	23.4%	713.7	5.4	0.9	4	46.6%	74.1%	900.6	4.0	2.7	0
		2	55.7%	20.2%	723.9	5.0	1.3	4	58.9%	65.3%	900.5	4.5	3.0	0
	Average		18.6%	17.0%	403.6	3.4	0.4	109/180	14.3%	40.7%	651.4	2.5	1.4	67/180

Table D.4
Matheuristic with the three-index formulation on instances with 10 vertices.

$\|\mathcal{D}\|$	α	$\mathcal{E}_{rel}$	$\|\mathcal{K}\| = 1$						$\|\mathcal{K}\| = 2$					
			Δ	t_{best}	n	#a	#c	$\leqslant$	Δ	t_{best}	n	#a	#c	$\leqslant$
1	1	0.2	0.1%	3.2	4.3	0.2	0.0	50	0.1%	0.4	2.6	0.3	0.0	50
		0.4	1.5%	11.2	5.2	0.9	0.0	50	0.3%	3.3	3.1	1.1	0.0	50
		0.6	3.4%	17.2	5.5	2.1	0.0	49	1.3%	7.9	3.9	1.8	0.0	50
		1	9.5%	25.4	5.8	2.1	0.0	50	3.5%	13.5	4.4	2.3	0.0	45
		1.5	12.7%	34.4	5.9	2.0	0.0	50	5.8%	7.1	3.8	2.4	0.0	50
		2	12.7%	27.1	5.0	2.0	0.0	49	7.2%	17.9	4.1	2.2	0.0	50
	2	0.2	0.9%	6.1	5.3	0.4	0.5	50	0.2%	1.0	3.0	0.4	0.3	50
		0.4	3.5%	8.5	5.6	1.9	0.2	50	1.0%	5.0	3.1	1.4	0.4	50
		0.6	9.5%	10.2	5.6	3.5	0.1	50	4.6%	10.9	5.1	2.8	0.6	45
		1	22.3%	24.6	5.7	4.2	0.1	42	16.5%	28.9	6.1	3.7	0.4	50
		1.5	27.0%	36.4	6.5	4.0	0.2	43	23.4%	27.4	5.2	4.0	1.0	30
		2	27.4%	36.1	6.5	4.0	0.2	38	23.4%	20.1	5.1	3.7	1.1	27
	3	0.2	1.4%	9.8	5.6	0.4	0.6	50	0.3%	1.3	3.0	0.4	0.4	50
		0.4	4.2%	14.0	6.2	1.8	0.6	50	1.6%	3.0	3.8	1.4	0.6	45
		0.6	10.9%	16.1	6.0	3.5	0.1	50	6.1%	10.0	5.0	2.6	1.0	45
		1	25.9%	40.5	7.8	4.3	0.8	43	19.2%	23.8	6.0	3.5	1.4	41
		1.5	32.2%	33.1	6.4	3.9	1.6	22	31.6%	20.4	5.9	3.3	2.7	18
		2	32.0%	41.7	7.2	3.7	1.6	14	32.8%	22.3	6.3	3.1	2.9	0
	Average		13.2%	22.0	5.9	2.5	0.4	800/900	9.9%	12.5	4.4	2.3	0.7	746/900
2	1	0.2	0.1%	3.0	4.4	0.2	0.0	50	0.1%	0.6	2.1	0.4	0.0	50
		0.4	1.7%	7.0	5.3	1.5	0.0	50	0.3%	2.1	3.1	1.2	0.0	50
		0.6	4.7%	24.3	6.1	2.8	0.0	49	1.6%	14.2	3.9	2.7	0.0	45
		1	14.5%	34.8	5.9	3.7	0.0	48	3.9%	17.3	4.5	3.7	0.1	45
		1.5	19.5%	39.3	5.0	3.4	0.0	50	6.3%	23.4	3.7	4.3	0.3	50
		2	19.8%	33.9	4.2	3.3	0.1	50	7.3%	20.7	3.4	4.4	0.0	50
	2	0.2	0.9%	7.6	4.9	0.4	0.5	50	0.2%	2.4	3.0	0.4	0.2	50
		0.4	3.7%	11.1	5.3	2.2	0.3	50	1.1%	11.1	3.3	1.6	0.6	50
		0.6	11.0%	26.2	6.7	4.0	0.5	50	5.2%	9.1	4.4	3.1	1.1	50
		1	28.6%	29.5	6.1	5.7	0.3	49	21.0%	32.9	6.6	4.3	1.0	45
		1.5	39.5%	41.8	6.4	5.5	0.6	33	33.7%	23.9	5.6	4.8	1.6	24
		2	40.2%	36.4	5.9	5.4	0.7	19	38.1%	30.3	5.3	5.2	0.8	10
	3	0.2	1.4%	11.1	5.1	0.4	0.6	50	0.3%	3.4	3.0	0.4	0.3	50
		0.4	4.6%	12.9	6.1	2.0	0.9	50	1.8%	5.4	3.5	1.6	0.7	45
		0.6	12.8%	24.9	6.8	3.8	0.8	50	6.9%	13.7	4.7	3.0	1.4	50
		1	33.1%	37.5	7.1	5.5	0.6	46	23.1%	23.0	5.9	4.2	1.5	45
		1.5	49.2%	45.9	7.4	4.2	2.8	6	47.2%	22.5	5.3	4.0	2.5	40
		2	51.5%	47.6	6.7	4.2	2.6	5	51.8%	32.7	5.7	4.1	2.5	5
	Average		18.7%	26.4	5.9	3.2	0.6	755/900	13.9%	16.0	4.3	3.0	0.8	754/900

Table D.5
Matheuristic with the two-index formulation on instances with 10 vertices.

$\|\mathcal{D}\|$	α	$\mathcal{E}_{rel}$	$\|\mathcal{K}\| = 1$						$\|\mathcal{K}\| = 2$					
			Δ	t_{best}	n	#a	#c	$\leqslant$	Δ	t_{best}	n	#a	#c	$\leqslant$
1	1	0.2	0.1%	5.3	4.9	0.3	0.0	50	0.1%	0.6	2.3	0.4	0.1	50
		0.4	1.5%	11.1	5.1	1.4	0.0	50	0.3%	1.6	3.0	1.4	0.0	50
		0.6	3.4%	14.4	5.4	2.2	0.0	49	1.3%	7.6	3.8	2.3	0.0	47
		1	9.5%	27.7	5.7	2.1	0.0	50	4.8%	21.9	5.0	2.5	0.0	50
		1.5	12.6%	37.0	5.5	1.9	0.0	49	5.8%	17.8	4.3	2.4	0.0	50
		2	12.7%	38.0	5.2	2.1	0.0	47	7.2%	22.0	5.1	2.1	0.0	50
	2	0.2	0.9%	8.5	5.1	0.4	0.5	49	0.2%	1.2	3.1	0.4	0.3	50
		0.4	3.5%	7.9	5.9	1.9	0.2	50	1.0%	1.1	3.7	1.4	0.4	45
		0.6	9.5%	17.9	6.9	3.5	0.1	50	4.6%	11.0	4.8	2.7	0.7	43
		1	22.2%	32.8	6.1	4.2	0.1	36	16.5%	20.6	6.3	4.1	0.3	49
		1.5	27.0%	42.6	6.5	3.9	0.3	30	23.5%	27.0	5.3	3.8	1.2	28
		2	27.4%	45.8	7.0	4.0	0.2	28	23.6%	21.2	5.3	3.7	1.2	26
	3	0.2	1.4%	15.8	6.0	0.4	0.6	50	0.3%	2.2	3.1	0.4	0.5	50
		0.4	4.2%	17.3	6.2	1.8	0.6	50	1.6%	2.7	3.9	1.4	0.7	45
		0.6	11.0%	14.6	6.0	3.4	0.2	46	6.0%	4.7	5.4	2.6	0.9	40
		1	25.5%	39.1	7.4	4.3	0.6	34	18.9%	22.0	6.2	3.7	1.0	37
		1.5	31.2%	41.4	6.8	4.0	0.9	15	30.6%	13.0	6.2	3.7	1.8	20
		2	31.1%	46.8	6.9	4.2	0.8	3	31.1%	18.1	5.7	3.3	1.8	0
	Average		13.1%	25.8	6.0	2.6	0.3	736/900	9.9%	12.0	4.6	2.3	0.6	730/900
2	1	0.2	0.1%	5.8	5.1	0.2	0.0	50	0.1%	0.9	2.3	0.4	0.2	50
		0.4	1.7%	16.2	5.0	1.7	0.0	50	0.3%	3.4	3.1	1.6	0.0	50
		0.6	4.7%	35.1	6.4	3.0	0.0	48	1.6%	12.8	4.4	3.2	0.1	42
		1	14.5%	38.5	5.9	3.7	0.0	47	4.6%	24.0	5.0	4.1	0.1	48
		1.5	19.5%	45.3	4.8	3.4	0.0	50	6.4%	23.9	5.0	4.5	0.3	45
		2	19.8%	42.1	4.9	3.2	0.0	50	7.3%	39.8	4.8	4.3	0.2	50
	2	0.2	0.9%	6.0	5.3	0.4	0.5	50	0.2%	0.9	2.9	0.4	0.4	50
		0.4	3.7%	14.9	5.6	2.3	0.2	50	1.1%	3.8	3.7	1.4	0.8	45
		0.6	11.0%	21.1	6.5	4.1	0.4	50	5.2%	5.0	4.8	3.1	1.2	45
		1	28.5%	43.9	6.6	5.7	0.3	47	21.0%	30.6	6.1	4.4	0.9	45
		1.5	39.6%	52.3	6.2	5.3	0.6	34	33.7%	32.2	5.9	4.8	1.6	21
		2	40.4%	42.0	6.0	5.6	0.5	19	38.3%	25.0	4.9	5.1	0.9	10
	3	0.2	1.4%	7.4	5.2	0.4	0.6	50	0.3%	1.5	2.9	0.4	0.5	50
		0.4	4.6%	15.7	5.9	2.0	0.9	50	1.8%	3.9	3.9	1.4	1.0	45
		0.6	12.8%	27.0	7.0	3.8	0.8	50	6.7%	10.3	4.9	3.0	1.4	45
		1	33.0%	38.3	7.1	5.5	0.6	45	23.2%	23.2	5.8	4.2	1.5	40
		1.5	48.0%	52.2	6.8	4.9	1.7	3	47.2%	24.4	6.2	3.8	2.6	40
		2	50.4%	52.7	6.9	4.6	1.6	0	51.9%	28.0	5.2	4.3	2.3	4
	Average		18.6%	30.9	6.0	3.3	0.5	743/900	13.9%	16.3	4.5	3.0	0.9	725/900

Table D.6
Detailed results on the FSTSP instances of Murray and Chu (2015) with 11 vertices.

Drone Speed (mph)	Instance	$\mathcal{E}_t$ = 20 min						$\mathcal{E}_t$ = 40 min					
		BKS	MILP			Matheuristic		BKS	MILP			Matheuristic	
			BKS_{rel}	Gap	t	BKS_{rel}	t_{best}		BKS_{rel}	Gap	t	BKS_{rel}	t_{best}
15	37v1	56.5	100.00%	0.0%	116.0	100.00%	12.9	50.6	100.00%	17.0%	900.0	100.00%	27.6
	37v2	53.2	100.00%	0.0%	62.0	100.00%	36.3	47.3	100.00%	0.0%	196.1	100.00%	64.8
	37v3	53.7	100.00%	0.0%	120.2	100.00%	16.7	53.7	100.00%	30.4%	900.0	100.00%	98.2
	37v4	67.5	100.00%	0.0%	510.5	100.00%	4.7	66.5	100.83%	39.4%	900.0	100.00%	83.3
	40v1	49.4	100.00%	0.0%	296.4	100.00%	12.9	46.9	100.00%	9.8%	900.0	100.00%	22.9
	40v2	50.7	100.00%	0.0%	69.1	100.00%	41.5	46.4	102.41%	11.0%	900.0	100.00%	43.2
	40v3	56.1	100.00%	0.0%	325.7	100.00%	50.3	53.9	104.02%	24.3%	900.0	100.00%	11.7
	40v4	69.9	100.00%	0.0%	891.1	100.00%	81.4	67.9	100.71%	37.1%	900.0	100.71%	8.1
	43v1	69.6	100.00%	0.0%	55.3	100.00%	63.7	55.5	100.00%	11.6%	900.0	100.00%	36.0
	43v2	72.1	100.00%	0.0%	54.8	100.00%	56.4	58.1	100.00%	15.5%	900.0	100.00%	37.9
	43v3	77.3	100.00%	0.0%	45.1	100.00%	43.4	69.2	102.43%	29.9%	900.0	100.05%	47.5
	43v4	90.1	100.00%	0.0%	521.1	100.00%	14.6	82.7	101.15%	37.5%	900.0	100.00%	89.5
25	37v5	47.5	108.29%	30.1%	900.0	106.52%	12.6	45.8	104.16%	23.3%	900.0	110.29%	50.9
	37v6	45.1	104.80%	27.9%	900.0	104.80%	39.0	44.6	106.07%	33.7%	900.0	97.76%	106.2
	37v7	49.6	100.00%	26.6%	900.0	100.00%	7.7	46.6	102.51%	30.1%	900.0	100.00%	93.7
	37v8	62.4	100.00%	35.3%	900.0	100.00%	57.7	59.6	101.34%	36.7%	900.0	99.67%	43.3
	40v5	43.5	100.00%	0.0%	139.5	102.51%	22.1	43.5	100.00%	0.0%	162.0	101.19%	19.5
	40v6	44.1	99.71%	0.0%	113.3	99.71%	15.3	44.1	99.40%	0.0%	153.1	99.40%	19.0
	40v7	49.5	99.90%	0.0%	181.1	99.90%	77.3	49.2	100.54%	14.8%	900.0	100.00%	68.3
	40v8	62.3	100.73%	25.1%	900.0	99.92%	82.8	62.0	101.28%	27.5%	900.0	99.95%	46.5
	43v5	53.1	102.86%	12.2%	900.0	104.60%	55.9	52.1	104.44%	28.9%	900.0	100.63%	77.7
	43v6	55.2	100.00%	23.1%	900.0	101.01%	54.2	52.3	105.05%	28.9%	900.0	100.00%	57.9
	43v7	64.4	100.00%	0.0%	240.2	100.00%	63.6	60.7	100.00%	25.9%	900.0	100.00%	79.3
	43v8	77.2	100.00%	25.7%	900.0	100.00%	98.0	73.7	98.97%	36.5%	900.0	98.97%	73.5
35	37v9	42.6	105.80%	29.3%	900.0	105.29%	56.1	42.4	101.25%	23.2%	900.0	105.71%	37.9
	37v10	41.9	101.76%	32.6%	900.0	99.57%	44.7	40.9	104.25%	24.1%	900.0	102.01%	34.3
	37v11	42.9	100.00%	21.3%	900.0	100.00%	28.3	42.9	100.00%	27.6%	900.0	100.00%	72.2
	37v12	56.7	98.24%	0.0%	575.0	98.24%	34.5	55.7	100.00%	35.3%	900.0	100.00%	70.3
	40v9	42.5	100.00%	0.0%	211.6	100.00%	16.1	42.5	100.00%	0.0%	229.2	100.00%	40.9
	40v10	43.1	100.00%	0.0%	122.3	100.00%	44.3	43.1	100.00%	0.0%	204.2	100.00%	26.2
	40v11	49.2	100.00%	0.0%	211.4	100.00%	28.2	49.2	100.00%	0.0%	204.4	100.00%	80.0
	40v12	62.0	100.00%	24.9%	900.0	100.00%	31.6	62.0	100.00%	25.0%	900.0	100.00%	40.1
	43v9	46.9	97.87%	17.9%	900.0	97.87%	108.8	46.9	102.97%	24.0%	900.0	100.00%	40.9
	43v10	47.9	100.00%	17.3%	900.0	97.91%	48.3	47.9	101.28%	21.2%	900.0	97.91%	64.0
	43v11	56.4	101.54%	20.5%	900.0	100.00%	75.6	56.4	101.54%	19.7%	900.0	100.00%	78.0
	43v12	69.2	100.00%	39.1%	900.0	100.00%	111.4	69.2	102.54%	43.0%	900.0	100.00%	59.1
Average			100.60%	11.4%	535.0	100.50%	45.8	53.7	101.37%	22.0%	781.9	100.40%	54.2

Table D.7
Matheuristic with the three-index formulation on instances with 20 vertices.

$\|\mathcal{D}\|$	α	$\mathcal{E}_r$	$\|\mathcal{K}\| = 1$				$\|\mathcal{K}\| = 2$				$\|\mathcal{K}\| = 3$			
			Δ	#a	#c	t_{best}	Δ	#a	#c	t_{best}	Δ	#a	#c	t_{best}
1	1	0.2	2.8%	7.9%	0.7%	38.4	0.8%	6.7%	0.0%	47.9	0.2%	5.5%	0.2%	32.5
		0.4	7.8%	19.7%	0.3%	26.8	3.7%	16.2%	0.0%	76.8	2.8%	16.3%	0.5%	46.7
		0.6	13.8%	22.7%	0.7%	31.4	7.3%	18.3%	0.2%	96.3	4.4%	20.8%	0.5%	34.9
		1	16.4%	20.4%	0.3%	20.3	11.7%	19.9%	0.1%	94.2	5.4%	24.6%	0.3%	70.0
	2	0.2	7.8%	11.9%	6.1%	47.6	2.1%	12.9%	3.1%	96.7	1.7%	11.8%	4.6%	21.4
		0.4	18.6%	33.8%	5.1%	52.7	7.5%	33.2%	3.8%	107.9	5.1%	29.2%	3.8%	42.8
		0.6	27.1%	38.3%	5.1%	60.3	16.8%	38.5%	3.5%	115.7	9.8%	33.1%	6.2%	75.4
		1	34.6%	37.5%	6.3%	68.1	25.0%	41.3%	3.8%	96.3	18.1%	40.8%	4.5%	108.0
	3	0.2	9.4%	12.5%	10.3%	39.7	3.2%	11.4%	6.7%	70.0	1.9%	12.1%	5.9%	30.6
		0.4	23.0%	33.2%	10.6%	74.6	9.2%	32.1%	9.6%	98.7	6.0%	27.2%	8.0%	48.9
		0.6	33.0%	38.6%	10.4%	75.0	19.5%	35.8%	10.3%	98.6	12.5%	31.4%	13.8%	53.5
		1	42.7%	37.4%	13.1%	101.0	26.9%	39.1%	14.5%	89.7	21.7%	36.0%	19.8%	94.6
2	1	0.2	3.4%	12.2%	2.2%	26.2	0.7%	10.1%	0.2%	47.8	0.7%	5.2%	0.0%	33.0
		0.4	12.3%	33.9%	0.9%	29.1	5.1%	28.2%	0.3%	105.8	3.2%	24.4%	0.6%	55.6
		0.6	22.4%	37.5%	0.7%	65.0	10.0%	35.7%	0.2%	104.7	3.4%	33.8%	1.8%	47.9
		1	25.7%	34.5%	0.2%	119.3	16.2%	38.9%	1.8%	86.8	4.7%	40.8%	2.4%	73.5
	2	0.2	5.7%	15.4%	5.4%	29.3	1.7%	13.9%	4.5%	75.2	2.2%	11.9%	4.9%	37.7
		0.4	22.5%	41.9%	4.8%	53.7	9.5%	40.4%	2.0%	79.8	7.2%	35.1%	4.8%	67.1
		0.6	33.9%	54.2%	4.4%	58.0	20.3%	48.9%	4.8%	90.1	14.2%	44.8%	7.5%	64.9
		1	46.6%	53.5%	5.2%	80.9	32.8%	53.2%	8.7%	101.0	24.3%	50.9%	9.6%	90.9
	3	0.2	9.0%	14.2%	11.1%	43.2	3.0%	13.1%	7.9%	71.2	2.7%	12.1%	6.6%	66.2
		0.4	28.1%	40.7%	8.9%	75.3	11.5%	38.1%	7.1%	89.5	8.2%	35.5%	7.6%	60.4
		0.6	40.7%	52.2%	9.2%	75.6	24.8%	44.9%	14.4%	112.9	17.1%	41.9%	13.2%	48.1
		1	56.5%	50.3%	17.7%	125.2	37.2%	49.7%	21.5%	99.4	29.4%	43.4%	26.7%	87.2
	Average		22.7%	31.4%	5.8%	59.0	12.8%	30.0%	5.4%	89.7	8.6%	27.9%	6.4%	58.0

Table D.8
Matheuristic with the two-index formulation on instances with 20 vertices.

$\|\mathcal{D}\|$	α	$\mathcal{E}_r$	$\|\mathcal{K}\| = 1$				$\|\mathcal{K}\| = 2$				$\|\mathcal{K}\| = 3$			
			Δ	#a	#c	t_{best}	Δ	#a	#c	t_{best}	Δ	#a	#c	t_{best}
1	1	0.2	2.5%	9.8%	0.5%	30.3	0.8%	7.7%	0.1%	50.8	0.4%	8.0%	0.3%	26.4
		0.4	8.2%	20.3%	0.4%	25.0	3.5%	17.8%	0.1%	75.9	3.2%	20.8%	1.1%	47.9
		0.6	12.1%	21.8%	0.3%	24.0	6.7%	19.8%	0.5%	86.6	4.4%	22.6%	1.1%	45.5
		1	16.4%	19.9%	0.0%	17.5	11.5%	20.4%	0.0%	91.1	6.3%	23.5%	0.5%	78.6
	2	0.2	6.2%	13.7%	5.3%	51.8	2.0%	13.2%	2.9%	85.3	1.6%	12.3%	5.7%	16.5
		0.4	16.6%	34.3%	4.1%	54.9	7.2%	33.2%	3.9%	84.3	5.0%	27.4%	4.9%	56.2
		0.6	23.9%	38.5%	5.2%	28.8	16.9%	38.2%	2.9%	98.9	10.1%	31.1%	7.4%	80.9
		1	31.5%	39.1%	5.1%	55.9	25.0%	40.7%	3.7%	101.5	17.6%	38.6%	5.9%	89.5
	3	0.2	10.4%	11.3%	10.7%	64.0	3.1%	11.4%	6.4%	74.1	1.8%	11.7%	5.8%	37.0
		0.4	21.7%	33.3%	9.1%	74.7	8.6%	31.7%	7.5%	98.6	5.2%	26.4%	7.8%	60.3
		0.6	33.0%	38.3%	10.6%	90.6	19.5%	35.8%	8.7%	110.5	12.7%	30.1%	12.1%	72.2
		1	39.6%	38.3%	10.7%	83.8	26.6%	39.8%	10.1%	99.2	21.1%	37.6%	13.1%	107.8
2	1	0.2	3.8%	12.7%	1.6%	34.2	0.6%	10.4%	0.1%	61.1	0.7%	9.4%	1.7%	32.5
		0.4	12.3%	34.0%	0.6%	41.5	4.8%	29.9%	0.0%	90.3	3.3%	30.1%	1.9%	60.1
		0.6	21.6%	38.4%	0.2%	59.8	10.5%	35.5%	0.4%	110.7	3.3%	35.1%	2.2%	54.3
		1	25.8%	33.6%	0.1%	93.0	16.0%	38.0%	1.1%	102.3	5.3%	39.1%	3.1%	95.6
	2	0.2	6.6%	15.1%	7.7%	24.5	2.3%	13.7%	5.5%	68.1	2.0%	10.9%	5.8%	28.5
		0.4	21.0%	42.9%	4.4%	66.0	9.4%	39.5%	3.3%	89.3	6.4%	32.3%	8.6%	59.4
		0.6	33.5%	54.0%	4.3%	46.5	19.6%	46.5%	5.5%	96.3	13.7%	41.4%	8.0%	57.4
		1	45.3%	54.9%	4.4%	101.6	32.4%	52.8%	7.6%	105.2	24.0%	49.8%	10.6%	90.5
	3	0.2	11.7%	14.0%	11.8%	62.6	2.7%	11.8%	8.7%	73.0	2.3%	11.2%	8.4%	43.4
		0.4	26.7%	40.8%	10.0%	59.1	11.5%	38.7%	6.9%	77.9	7.8%	32.9%	10.7%	73.6
		0.6	36.8%	52.5%	8.7%	60.6	23.7%	46.1%	10.3%	99.6	16.7%	40.9%	14.2%	54.2
		1	51.9%	53.4%	13.5%	106.5	35.8%	52.1%	15.1%	119.7	28.5%	47.5%	18.5%	98.5
	Average		21.6%	31.9%	5.4%	56.6	12.5%	30.2%	4.6%	89.6	8.5%	27.9%	6.6%	61.1

Table D.9
Matheuristic with the three-index formulation on instances with 50 vertices.

$\|\mathcal{D}\|$	α	$\mathcal{E}_r$	$\|\mathcal{K}\| = 1$				$\|\mathcal{K}\| = 2$				$\|\mathcal{K}\| = 3$			
			Δ	#a	#c	t_{best}	Δ	#a	#c	t_{best}	Δ	#a	#c	t_{best}
1	1	0.2	6.2%	21.7%	0.2%	67.4	3.0%	19.3%	0.0%	92.5	2.0%	17.9%	0.0%	77.2
		0.4	15.4%	23.7%	0.1%	89.7	7.8%	23.6%	0.0%	102.5	4.1%	24.3%	0.0%	104.3
		0.6	16.4%	22.2%	0.0%	97.7	9.6%	23.1%	0.0%	104.9	5.8%	23.9%	0.1%	114.3
		1	16.3%	22.2%	0.1%	70.2	9.7%	23.1%	0.0%	116.1	5.9%	23.7%	0.1%	134.4
	2	0.2	14.6%	31.4%	3.3%	78.7	8.4%	29.2%	2.2%	96.0	4.4%	26.7%	1.8%	74.3
		0.4	30.4%	40.4%	2.9%	85.6	19.3%	40.6%	1.3%	116.7	12.6%	39.1%	2.0%	112.0
		0.6	30.6%	39.1%	3.0%	112.9	19.9%	41.7%	1.8%	90.3	13.6%	40.7%	2.6%	97.9
		1	29.6%	40.0%	2.8%	91.4	19.8%	41.3%	2.0%	108.4	14.2%	41.4%	2.7%	101.1
	3	0.2	16.9%	30.4%	7.7%	94.4	9.5%	28.2%	5.3%	85.3	5.3%	26.9%	5.1%	71.2
		0.4	33.6%	39.6%	9.1%	99.2	20.2%	40.3%	5.3%	85.7	13.3%	38.7%	6.1%	106.1
		0.6	34.8%	38.2%	11.2%	85.2	21.4%	40.7%	6.8%	101.8	15.2%	38.8%	8.6%	107.9
		1	34.0%	38.5%	10.6%	83.3	21.7%	40.6%	6.7%	105.5	15.7%	39.5%	9.6%	106.9
2	1	0.2	7.8%	32.2%	0.5%	87.0	4.3%	28.5%	0.2%	110.4	1.9%	27.1%	0.3%	83.9
		0.4	21.0%	42.6%	0.6%	90.2	10.7%	42.6%	0.7%	112.4	5.8%	42.0%	0.9%	131.0
		0.6	21.8%	42.3%	0.7%	123.9	12.0%	42.9%	0.9%	140.4	7.3%	43.1%	1.5%	127.4
		1	20.7%	42.5%	0.6%	140.1	12.5%	42.9%	0.7%	173.6	7.8%	44.6%	1.3%	128.6
	2	0.2	16.2%	36.2%	5.0%	78.7	9.8%	34.1%	2.5%	91.5	5.1%	31.5%	3.3%	85.9
		0.4	40.0%	56.8%	2.9%	101.6	24.5%	56.6%	2.0%	135.6	15.2%	52.5%	2.6%	91.5
		0.6	42.6%	56.1%	3.3%	86.0	27.4%	57.8%	2.1%	112.2	19.0%	55.5%	4.2%	92.1
		1	43.3%	56.4%	3.9%	91.1	27.7%	57.5%	2.4%	112.3	20.4%	57.6%	3.9%	109.0
	3	0.2	18.7%	34.4%	8.7%	71.1	10.5%	33.1%	6.2%	94.6	5.7%	30.9%	4.8%	86.8
		0.4	42.9%	56.2%	7.5%	91.5	26.2%	55.6%	5.0%	104.3	17.1%	51.2%	6.5%	87.5
		0.6	47.8%	54.8%	10.8%	133.2	30.0%	56.0%	7.7%	115.6	21.7%	53.4%	10.0%	102.0
		1	47.4%	54.2%	11.2%	91.3	30.6%	56.1%	9.5%	106.7	23.2%	55.7%	11.2%	115.9
Average			27.0%	39.7%	4.4%	93.4	16.5%	39.8%	3.0%	109.0	10.9%	38.6%	3.7%	102.1

Table D.10
Matheuristic with the two-index formulation on instances with 50 vertices.

$\|\mathcal{D}\|$	α	$\mathcal{E}_r$	$\|\mathcal{K}\| = 1$				$\|\mathcal{K}\| = 2$				$\|\mathcal{K}\| = 3$			
			Δ	#a	#c	t_{best}	Δ	#a	#c	t_{best}	Δ	#a	#c	t_{best}
1	1	0.2	6.1%	21.6%	0.1%	91.5	2.8%	20.5%	0.0%	90.6	1.8%	17.6%	0.1%	112.3
		0.4	14.4%	24.5%	0.0%	91.2	7.2%	25.2%	0.1%	118.7	4.1%	24.6%	0.1%	116.1
		0.6	14.7%	22.4%	0.1%	96.0	8.2%	24.5%	0.0%	101.2	5.3%	24.0%	0.2%	116.5
		1	14.9%	22.2%	0.0%	79.5	8.5%	23.3%	0.2%	113.9	5.3%	24.1%	0.1%	119.1
	2	0.2	13.9%	30.9%	3.3%	93.1	8.2%	29.3%	2.0%	97.5	4.5%	26.5%	1.6%	76.8
		0.4	29.5%	39.7%	2.9%	81.4	18.9%	40.0%	1.6%	140.4	12.1%	39.0%	1.8%	94.2
		0.6	30.7%	39.4%	3.4%	105.7	19.7%	41.0%	2.1%	108.3	13.4%	39.8%	2.6%	111.6
		1	29.8%	38.9%	3.6%	91.7	19.6%	41.1%	1.9%	109.8	13.9%	40.0%	3.8%	90.8
	3	0.2	16.5%	30.2%	6.5%	85.0	9.2%	29.3%	4.1%	110.5	5.0%	26.5%	4.7%	82.6
		0.4	32.6%	40.3%	8.1%	85.4	20.1%	40.4%	4.7%	117.9	13.3%	38.2%	6.2%	101.1
		0.6	33.4%	38.8%	9.4%	88.1	21.6%	41.3%	6.4%	114.9	14.7%	39.6%	7.5%	89.6
		1	33.6%	39.3%	8.2%	96.6	21.4%	40.9%	5.9%	122.8	15.4%	40.0%	8.2%	112.5
2	1	0.2	6.5%	31.5%	1.2%	82.7	3.3%	30.0%	0.3%	105.1	1.8%	27.5%	0.7%	107.2
		0.4	18.9%	45.3%	0.9%	115.5	9.2%	44.0%	0.9%	118.7	5.4%	42.1%	1.2%	144.2
		0.6	20.2%	42.1%	0.9%	121.5	11.5%	44.7%	1.0%	136.6	6.8%	44.0%	1.1%	167.9
		1	18.9%	42.7%	0.8%	117.7	11.1%	43.5%	0.9%	158.3	7.4%	45.3%	1.0%	174.0
	2	0.2	15.6%	35.4%	5.0%	87.6	9.3%	33.8%	2.7%	104.7	4.9%	31.0%	3.4%	91.9
		0.4	39.0%	56.7%	2.2%	115.2	23.3%	56.1%	1.9%	110.8	15.0%	52.0%	2.9%	88.5
		0.6	42.6%	57.1%	3.4%	108.8	26.9%	57.6%	2.0%	112.7	18.7%	55.1%	3.6%	117.4
		1	41.8%	56.0%	3.0%	85.2	27.0%	57.7%	2.3%	122.1	19.8%	56.2%	3.6%	126.0
	3	0.2	18.6%	33.5%	8.4%	108.8	10.4%	32.9%	5.1%	85.3	5.3%	29.5%	6.7%	98.4
		0.4	41.8%	57.1%	4.6%	107.1	25.1%	54.4%	5.1%	97.8	16.8%	51.4%	5.5%	91.9
		0.6	45.8%	55.7%	8.0%	126.5	29.4%	56.8%	5.9%	105.8	21.0%	52.7%	9.0%	104.0
		1	45.8%	54.7%	9.1%	117.9	29.3%	57.0%	6.6%	103.5	22.6%	55.1%	9.8%	116.6
Average			26.1%	39.8%	3.9%	99.2	15.9%	40.2%	2.7%	112.8	10.6%	38.4%	3.5%	110.5

Table D.11
Matheuristic with the three-index formulation on instances with 100 vertices.

$\|\mathcal{D}\|$	α	$\mathcal{E}_r$	$\|\mathcal{K}\| = 1$				$\|\mathcal{K}\| = 2$				$\|\mathcal{K}\| = 3$			
			Δ	#a	#c	t_{best}	Δ	#a	#c	t_{best}	Δ	#a	#c	t_{best}
1	1	0.2	8.1%	23.8%	0.1%	4.3	3.9%	23.0%	0.0%	3.9	1.9%	22.3%	0.2%	3.3
		0.4	13.1%	22.0%	0.0%	12.5	7.0%	22.6%	0.1%	8.5	4.3%	23.1%	0.1%	6.9
		0.6	13.0%	22.1%	0.0%	12.1	7.6%	22.3%	0.0%	10.6	4.5%	22.6%	0.1%	8.3
		1	13.1%	22.2%	0.1%	15.2	7.6%	22.4%	0.0%	10.8	5.0%	23.0%	0.0%	8.9
	2	0.2	17.9%	38.3%	2.6%	2.4	10.5%	37.0%	2.1%	5.5	5.6%	35.0%	1.7%	3.9
		0.4	23.7%	39.7%	3.0%	4.7	15.2%	40.5%	2.5%	14.9	9.9%	41.0%	1.8%	6.8
		0.6	23.4%	40.2%	3.1%	5.5	15.4%	40.9%	2.4%	8.4	12.2%	41.3%	2.1%	13.7
		1	23.9%	40.1%	2.7%	5.5	15.9%	40.5%	2.4%	12.6	11.4%	41.2%	2.4%	10.3
	3	0.2	19.4%	37.3%	6.8%	2.4	11.4%	36.6%	5.1%	2.7	6.3%	34.3%	4.3%	4.8
		0.4	25.5%	38.7%	9.2%	4.4	17.5%	40.0%	6.9%	9.7	13.0%	39.9%	6.2%	10.9
		0.6	25.8%	38.5%	9.6%	5.1	17.0%	39.9%	7.4%	10.5	15.2%	39.9%	7.0%	23.3
		1	25.6%	39.2%	9.2%	5.2	17.3%	39.9%	7.5%	6.4	15.0%	40.2%	6.4%	11.4
2	1	0.2	11.1%	41.7%	0.6%	11.4	5.3%	38.4%	0.3%	9.5	2.5%	36.1%	0.5%	7.5
		0.4	17.8%	42.6%	0.9%	52.0	10.4%	42.7%	0.6%	32.4	5.7%	42.8%	0.5%	25.2
		0.6	18.4%	42.3%	0.8%	53.5	11.0%	42.7%	0.7%	40.0	7.3%	42.5%	0.8%	35.8
		1	17.9%	42.2%	0.8%	53.9	10.6%	42.5%	0.6%	47.3	6.8%	43.2%	0.6%	30.4
	2	0.2	22.8%	50.9%	2.2%	2.7	12.9%	48.9%	1.7%	2.8	7.3%	45.6%	1.5%	3.9
		0.4	33.8%	56.3%	3.4%	5.2	26.1%	56.5%	3.0%	20.7	20.4%	56.5%	2.2%	22.2
		0.6	33.9%	56.4%	3.5%	5.8	25.7%	57.3%	2.8%	16.4	24.4%	57.3%	3.1%	30.2
		1	33.5%	56.4%	3.6%	6.2	25.2%	57.4%	2.7%	14.1	25.2%	57.2%	3.1%	33.2
	3	0.2	24.2%	49.3%	5.6%	2.8	14.1%	47.6%	4.3%	3.8	8.7%	44.2%	3.6%	5.8
		0.4	36.4%	54.5%	9.2%	4.2	26.9%	55.7%	7.1%	16.7	24.2%	54.3%	7.3%	24.5
		0.6	36.5%	54.5%	9.8%	5.1	31.6%	54.8%	9.3%	31.3	30.8%	54.9%	8.8%	64.3
		1	36.6%	53.7%	10.3%	5.2	30.7%	55.3%	8.2%	25.8	32.7%	55.2%	9.3%	43.7
Average			23.1%	41.8%	4.1%	12.0	15.7%	41.9%	3.2%	15.2	12.5%	41.4%	3.1%	18.3

Table D.12
Matheuristic with the two-index formulation on instances with 100 vertices.

$\|\mathcal{D}\|$	α	$\mathcal{E}_r$	$\|\mathcal{K}\| = 1$				$\|\mathcal{K}\| = 2$				$\|\mathcal{K}\| = 3$			
			Δ	#a	#c	t_{best}	Δ	#a	#c	t_{best}	Δ	#a	#c	t_{best}
1	1	0.2	7.2%	24.8%	0.1%	5.6	3.6%	24.1%	0.1%	4.9	1.8%	22.8%	0.4%	4.2
		0.4	11.6%	22.8%	0.0%	14.5	6.4%	23.2%	0.1%	10.9	4.1%	23.0%	0.3%	8.4
		0.6	11.2%	22.3%	0.0%	15.6	6.7%	22.3%	0.1%	12.8	4.3%	22.0%	0.2%	9.9
		1	11.3%	22.2%	0.0%	15.8	6.6%	22.1%	0.0%	13.1	4.3%	22.1%	0.2%	11.4
	2	0.2	17.6%	38.7%	2.3%	3.1	10.2%	37.4%	1.9%	3.2	5.6%	34.6%	2.7%	2.6
		0.4	23.4%	40.4%	2.7%	5.8	14.9%	40.0%	2.6%	6.3	11.2%	39.3%	3.0%	14.1
		0.6	23.4%	40.2%	2.8%	6.6	15.6%	39.8%	2.3%	10.7	11.0%	39.7%	2.1%	6.8
		1	23.3%	40.1%	3.1%	6.8	15.1%	40.9%	2.0%	8.5	10.9%	40.3%	2.1%	11.0
	3	0.2	19.1%	37.7%	5.8%	3.1	10.3%	35.7%	5.3%	3.8	6.2%	33.7%	5.8%	7.1
		0.4	25.6%	39.3%	8.4%	5.7	16.2%	39.6%	6.3%	7.4	11.5%	39.4%	6.0%	8.1
		0.6	24.6%	39.4%	8.0%	6.8	16.6%	39.1%	6.5%	11.3	13.8%	39.9%	6.3%	16.0
		1	25.7%	39.4%	7.8%	7.3	18.0%	39.7%	6.7%	8.5	13.2%	39.2%	6.4%	12.4
2	1	0.2	10.5%	41.9%	0.8%	17.4	4.6%	40.0%	0.8%	13.3	2.3%	36.4%	0.9%	11.2
		0.4	16.0%	42.9%	0.9%	50.2	9.7%	42.4%	0.8%	40.6	5.5%	40.7%	0.8%	35.3
		0.6	16.5%	41.7%	1.3%	54.4	9.8%	41.9%	0.7%	43.1	6.1%	39.5%	1.1%	37.7
		1	16.6%	42.4%	1.1%	49.2	9.9%	41.6%	0.7%	44.0	6.5%	41.7%	0.9%	41.0
	2	0.2	23.3%	50.4%	2.8%	4.1	13.0%	47.6%	2.0%	4.9	7.9%	43.8%	2.5%	5.9
		0.4	33.0%	56.4%	3.0%	9.2	23.2%	56.0%	2.2%	10.7	19.5%	55.1%	3.0%	23.7
		0.6	34.0%	55.7%	3.2%	9.7	23.1%	56.8%	2.5%	11.9	25.0%	56.0%	2.8%	40.8
		1	33.0%	56.9%	2.6%	10.5	24.0%	56.5%	2.4%	16.5	24.9%	56.6%	2.5%	40.9
	3	0.2	24.7%	49.8%	5.1%	4.0	14.0%	47.3%	4.4%	7.2	7.3%	43.5%	4.5%	3.5
		0.4	36.0%	55.2%	7.7%	7.1	26.1%	55.0%	6.3%	19.9	22.2%	53.9%	5.5%	27.0
		0.6	36.3%	55.0%	7.8%	8.4	30.9%	55.4%	6.9%	26.0	27.6%	55.2%	6.9%	49.0
		1	35.8%	54.5%	8.5%	9.5	31.5%	54.6%	7.5%	32.8	26.5%	54.9%	7.1%	40.8
Average			22.5%	42.1%	3.6%	13.8	15.0%	41.6%	3.0%	15.5	11.6%	40.6%	3.1%	19.5

References

Agatz, N., Bouman, P., Schmidt, M., 2018. Optimization approaches for the traveling salesman problem with drone. Transport. Sci. 52, 965–981. https://doi.org/10.1287/trsc.2017.0791.

Archetti, C., Speranza, M.G., 2014. A survey on matheuristics for routing problems. EURO J. Comput. Optim. 2, 223–246. https://doi.org/10.1007/s13675-014-0030-7.

Bouman, P., Agatz, N., Schmidt, M., 2018. Dynamic programming approaches for the traveling salesman problem with drone. Networks 72, 528–542. https://doi.org/10.1002/net.21864.

Boysen, N., Briskorn, D., Fedtke, S., Schwerdfeger, S., 2018. Drone delivery from trucks: drone scheduling for given truck routes. Networks 72, 506–527. https://doi.org/10.1002/net.21847.

Carlsson, J.G., Song, S., 2018. Coordinated logistics with a truck and a drone. Manage. Sci. 64, 3971–4470. https://doi.org/10.1287/mnsc.2017.2824.

Clarke, G., Wright, J.W., 1964. Scheduling of vehicles from a central depot to a number of delivery points. Oper. Res. 12, 568–581. https://doi.org/10.1287/opre.12.4.568.

Di Puglia Pugliese, L., Guerriero, F., 2017. Last-mile deliveries by using drones and classical vehicles. Optim. Decis. Sci. 217, 557–565. https://doi.org/10.1007/978-3-319-67308-0_56.

Dorling, K., Heinrichs, J., Messier, G.G., Magierowski, S., 2017. Vehicle routing problems for drone delivery. IEEE Trans. Syst. Man Cybern.: Syst. 47, 70–85. https://doi.org/10.1109/TSMC.2016.2582745.

Drexl, M., 2012. Synchronization in vehicle routing-a survey of VRPs with multiple synchronization constraints. Transport. Sci. 46, 297–316. https://doi.org/10.1287/trsc.1110.0400.

Es Yurek, E., Ozmutlu, H.C., 2018. A decomposition-based iterative optimization algorithm for traveling salesman problem with drone. Transport. Res. Part C: Emerg. Technol. 91, 249–262. https://doi.org/10.1016/j.trc.2018.04.009.

Gurobi Optimization, 2019. Gurobi Optimizer Reference Manual. < http://www.gurobi.com > .

Ha, Q.M., Deville, Y., Pham, Q.D., Hà, M.H., 2018. On the min-cost traveling salesman problem with drone. Transport. Res. Part C: Emerg. Technol. 86, 597–621. https://doi.org/10.1016/j.trc.2017.11.015.

Jünger, M., Liebling, T.M., Naddef, D., Nemhauser, G.L., Pulleyblank, W.R., Reinelt, G., Rinaldi, G., Wolsey, L.A. (Eds.), 2010. 50 Years of Integer Programming 1958–2008. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-68279-0.

Lin, S., Kernighan, B.W., 1973. An effective heuristic algorithm for the traveling-salesman problem. Oper. Res. 21, 498–516. https://doi.org/10.1287/opre.21.2.498.

Marinelli, M., Caggiani, L., Ottomanelli, M., Dell'Orco, M., 2018. En route truck-drone parcel delivery for optimal vehicle routing strategies. IET Intell. Transport Syst. 12, 253–261. https://doi.org/10.1049/iet-its.2017.0227.

Murray, C.C., Chu, A.G., 2015. The flying sidekick traveling salesman problem: optimization of drone-assisted parcel delivery. Transport. Res. Part C: Emerg. Technol. 54, 86–109. https://doi.org/10.1016/j.trc.2015.03.005.

Otto, A., Agatz, N., Campbell, J., Golden, B., Pesch, E., 2018. Optimization approaches for civil applications of unmanned aerial vehicles (UAVs) or aerial drones: a survey. Networks 72, 411–458. https://doi.org/10.1002/net.21818.

Poikonen, S., Wang, X., Golden, B., 2017. The vehicle routing problem with drones: extended models and connections. Networks 70, 34–43. https://doi.org/10.1002/net.21746.

Sacramento, D., Pisinger, D., Ropke, S., 2019. An adaptive large neighborhood search metaheuristic for the vehicle routing problem with drones. Transport. Res. Part C: Emerg. Technol. 102, 289–315. https://doi.org/10.1016/j.trc.2019.02.018.

Schermer, D., Moeini, M., Wendt, O., 2018a. A Variable Neighborhood Search Algorithm for Solving the Vehicle Routing Problem with Drones (Technical Report). < https://bisor.de/en > .

Schermer, D., Moeini, M., Wendt, O., 2018b. Algorithms for solving the vehicle routing problem with drones. Lect. Notes Artif. Intell. 10751, 352–361. https://doi.org/10.1007/978-3-319-75417-8_33.

Schermer, D., Moeini, M., Wendt, O., 2019. A hybrid VNS/tabu search algorithm for solving the vehicle routing problem with drones and en route operations. Comput. Oper. Res. https://doi.org/10.1016/j.cor.2019.04.021.

Toth, P., Vigo, D., 2014. Vehicle Routing: Problems, Methods, and Applications. Society for Industrial and Applied Mathematics.

Wang, X., Poikonen, S., Golden, B., 2017. The vehicle routing problem with drones: several worst-case results. Optim. Lett. 11, 679–697. https://doi.org/10.1007/s11590-016-1035-3.