A Bi-objective Routing Problem with Trucks and Drones: Minimizing Mission Time and Energy Consumption

Mahdi Moeini[1(✉)], Oliver Wendt[2], and Marius Schummer[2]

[1] SAMOVAR, Télécom SudParis, ENSIIE, Institut Polytechnique de Paris, 91120 Palaiseau, France
mahdi.moeini@ensiie.fr

[2] BISOR, RPTU Kaiserslautern-Landau, 67663 Kaiserslautern, Germany
wendt@wiwi.uni-kl.de, marius.schummer@knecon.com

Abstract. This paper addresses the *Vehicle Routing Problem with Drones* (VRPD), where the objective consists in creating shortest parcel delivery routes that are operated by a mixed fleet of trucks and drones. After formulating the VRPD as *Mixed Integer Linear Program* (MILP), we introduce a two-phase *Simulated Annealing* algorithm to solve the VRPD. Using benchmark instances, we conducted extensive computational experiments to prove the effectiveness of our algorithm. Additionally, we investigated the impact of using drones, not only on completion time but also on energy consumption of the truck-drone fleet under real-world resembling scenarios. Indeed, by optimizing for minimal makespan as well as minimal energy consumption, and by applying a bi-objective optimization model, the overall observation confirms substantial benefits of using drones on reduction of makespan and energy consumption. Thus, using a combined truck-drone fleet, last-mile logistics may become not only more cost-efficient but also more environmentally friendly and sustainable.

Keywords: Logistics · Vehicle Routing Problem with Drones · Mixed Integer Linear Program · Heuristic · Simulated Annealing

1 Introduction

With an increasing urban population and the rising popularity of e-commerce, the last-mile logistics faces many challenges. Local air and noise pollution, the emission of climate-damaging greenhouse gases, and exuberant traffic congestion are among the increasing external effects of last-mile logistics, which can make home deliveries environmentally damaging, economically inefficient, and opposing to the *Sustainable Development Goals* (SDGs), which are adopted by the United Nations, in particular Goal 11 that is directly linked to the sustainable development of cities[1]. However, thanks to recent innovations, including engine

[1] https://sdgs.un.org/goals.

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2023
O. Gervasi et al. (Eds.): ICCSA 2023 Workshops, LNCS 14106, pp. 291–308, 2023.
https://doi.org/10.1007/978-3-031-37111-0_21

technologies and autonomous vehicles, these challenges might be resolved. In this context, the rapidly evolving drone technology has attracted attention of researchers and practitioners in the logistics sector. Indeed, since drones are not bound to the road network, they constitute an interesting new mode of transportation which makes it possible to escape road traffic, to reach certain destinations quickly, and to reduce environmental and social impacts [2,4]. The well-known technology companies, e.g., Alphabet, Amazon, and DHL, have shown interests in using drones for delivering parcels (see [15] and references therein).

In this context, ideas have emerged to use drones alongside trucks as assisting delivery vehicles. In fact, drones can be used to take parcels autonomously from a truck to a customer or the other way round. In this case, the truck is used as a mobile depot and carrier for drones, where the battery of drones can be replaced or recharged. Moreover, the truck can continue to serve other customers while the drone is airborne. Thus, the total mission time will decrease.

These concepts raise the question of how to plan last-mile logistics routes for such a combined trucks-drones delivery concept. On the one hand, the unique features of trucks and drones enable the possibility for efficiency gains, but on the other hand, they impose limits on the form of generated routes.

In order to obtain the best possible benefits in delivering parcels by a combined truck-drone fleet, several new optimization problems have been introduced, e.g., the *Traveling Salesman Problem with Drone* (TSP-D) [1,3,17,24,25] and the *Vehicle Routing Problem with Drones* (VRPD) [5,15,19–23,27,28], which are extensions to the classical *Traveling Salesman Problem* (TSP) and *Vehicle Routing Problem* (VRP), respectively.

In the context of the VRPD, the contributions of this paper are as follows:

- We introduce a bi-objective truck-drone routing model, and present a two-phase algorithm based on the *Simulated Annealing* (SA) metaheuristic to solve larger instances of the bi-objective VRPD.
- We present the numerical results of our extensive computational experiments on benchmark instances and realistic scenarios, i.e., on instances with parameters that approximate real-world scenarios, in order to investigate the impacts that drone-assisted delivery methods can have on makespan and in particular on energy consumption.

The remainder of this paper is organized as follows: After a brief literature review in Sect. 2, we present our notation and assumptions, a formal description of the VRPD, and its mathematical formulations in Sect. 3. To solve the VRPD, a two-phase Simulated Annealing algorithm is presented in Sect. 4. Section 5 is devoted to the numerical results of our computational experiments covering small and large benchmark instances with and without realistic-scenario assumptions. Section 6 contains our concluding remarks.

2 Literature Review

Detailed reviews on truck-drone delivery concepts can be found, e.g., in [15] and references therein. However, we present a short overview on a selection of papers that are closely related to the context of this paper.

In fact, we investigate a variant of the *Vehicle Routing Problem with Drone* (VRPD), which was introduced by Wang et al. [23,28]. In the VRPD, a set of trucks, where each truck is equipped with identical drones, is tasked to deliver parcels to a given set of customers in the shortest possible time. Compared to the classical vehicle routing problem (VRP), the VRPD has additional features, e.g., the delivery fleet is heterogeneous, one type of vehicles, i.e., the drone, has a limited carrying capacity and a restricted travel distance, and the VRPD requires synchronization constraints [21]. The VRPD was intensively investigated in, e.g., [13,15,19–23,27].

The VRPD can be formulated as a *mixed integer linear program* (MILP), which can then be solved by state-of-the-art MILP solvers, e.g., IBM CPLEX or Gurobi Optimizer [8]. As an extension to the classical VRP, the VRPD is known as an NP-hard problem for which it is impossible to find the global optima within reasonable time, if we use exact methods and the instance size is large [29].

In [22], the authors provide a hybrid approach based on tabu search and *variable neighborhood search* (VNS) [21] to solve the VRPD in presence of *en route operations*, i.e., the drones are able to start and end their mission at some discrete points on the road connecting two customers. In [23], the authors present a matheuristic, which decomposes the VRPD and solves *drone assignment and scheduling problems*. This approach is tested on instances with up to 100 customers. In addition, [23] introduces several classes of valid inequalities and investigates different variants of the VRPD.

In [27], the VRPD is formulated as a MILP, and a branch-and-cut algorithm for the VRPD is derived. The presented algorithm can optimally solve instances with up to 30 customers. The paper shows that using truck-drone tandems can not only improve makespan, but also reduce the fleet size without decreasing the efficiency of transportation systems.

In a recent paper, Masmoudi et al. studied a variant of the VRPD where drones are equipped with a multi-package payload compartment. To solve the problem, the authors use an adaptive multi-start simulated annealing. The algorithm is tested on some benchmark instances and the results are compared with a hybrid genetic algorithm [16].

Even though addressing large instances can currently be done only by heuristics and metaheuristics, these approaches cannot guarantee global optimality of the obtained solutions. However, a well-designed heuristic can provide high-quality solutions in reasonable computation time [16,29]. In this paper, we not only introduce an efficient simulated annealing algorithm to solve the VRPD, but also investigate two kinds of objective functions: *minimizing makespan* and *minimizing energy consumption*. To further bridge the research gap, in our numerical investigations, we use parameters that approximate real-world conditions.

3 The Vehicle Routing Problem with Drones

In this section, we present our assumptions, the notation that we are going to use throughout this paper, and the mathematical formulations of the VRPD.

3.1 Notation and Basic Assumptions

We assume that a depot, a set of customers, and a fleet of trucks are given such that each truck is equipped with a certain number of drones. Starting from the depot, the trucks and the drones have the mission of delivering parcels to the customers in the shortest possible time. Each customer is served exactly once, either by a truck or a drone. On a truck's tour, the assigned drones of the truck can launch from it, serve a customer, and return to the truck. This operation performed by a drone is called a *sortie*. Moreover, we accept the following assumptions on drone sorties and deliveries [3,21,22]:

- A drone can serve exactly one customer per sortie. However, a truck can serve several customers, even while its drones are performing a sortie. In addition, we assume that each truck has sufficient capacity for all deliveries.
- A drone can depart from and return to the same truck. Launching from or landing on a truck is only allowed when the truck is at a customer location or at the depot. The pickup of a drone cannot happen at the same customer location it was launched from, i.e., *cyclic sorties* are not permitted. For the return of a drone to a truck at a certain location, the drone may need to wait for the truck to arrive or the truck should wait for the drone.
- The flight endurance of a drone is limited to $\mathcal{E}$ distance units per sortie. Upon return to a truck, the endurance is restored immediately, which means a drone can be launched from the same node where it returned to the truck. We assume that the trucks have an unlimited endurance.
- Service times for delivering parcels by either drone or truck as well as the service and setup time for drone launching and retrieval are negligible.
- The drones' velocity is α times the trucks' speed. Since drones do not face traffic congestion or road priority rules, we can assume $\alpha \geq 1$.
- Risk-free: Drone deliveries always succeed once the drone has launched.

The objective of the VRPD is to minimize the mission time (*makespan*), which is the duration that it takes the vehicles to serve all customers and to return to the depot afterwards. To formulate the VRPD as a MILP, we need to present the required notation and decision variables [21–23].

Let $G = (V, E)$ be a symmetric complete graph, where V is the set of vertices (nodes) and E is the set of edges. The set $V = \{0, 1, \ldots, n+1\}$ contains n customer locations $V_N = \{1, \ldots, n\}$, and the depot nodes 0 and $n+1$. Both depot nodes, one at the beginning and one at the end of a tour, represent a single location ($0 \equiv n+1$). Moreover, we assume that the set $V_L = V \setminus \{n+1\}$ holds all nodes from which a drone can launch for a sortie, and $V_R = V \setminus \{0\}$ contains all nodes where a drone can be retrieved.

A *drone sortie* is defined by a tuple (i, w, j), where $i \in V_L$ denotes a *launch node*, w is a *customer node* that is served by the drone, and $j \in V_R$ is a *return node*, where the drone rejoins the truck.

We use d_{ij} and $\bar{d_{ij}}$ to denote the distance between nodes i and j for a truck and for a drone, respectively. The average velocity of the trucks and that of the drones is represented by v and $\bar{v}$, respectively, where $\bar{v} = \alpha \cdot v$. The travel time from vertex i to vertex j can therefore be stated as $t_{ij} = d_{ij}/v$ for trucks and $\bar{t_{ij}} = \bar{d_{ij}}/\bar{v}$ for drones. Furthermore, the set $K = \{1, \ldots, K_n\}$, $K_n \in \mathbb{Z}^+$ indicates that $|K| \geq 1$ trucks are available such that each of them is equipped with $D \in \mathbb{Z}^{\geq 0}$ drones.

As decision variables for the model, we use $x_{ij}^k \in \{0, 1\}$ to show whether the edge (i, j) is traversed by truck k. The variables $y_{iwj}^k \in \{0, 1\}$ state if the drone sortie (i, w, j) is part of the route of truck k, meaning that one of the truck's drones launches from node $i \in V_L$, serves customer $w \in V_N$ and returns to the same truck k at node $j \in V_R$ ($i \neq j$, $i \neq w$, $w \neq j$). With $p_{ij}^k \in \{0, 1\}$, we indicate whether node i is served at some point before node j (but not necessarily immediately before j) on the route of truck k. The integer variables u_i^k, having a lower bound of 0, show at which position the customer node i is possibly served through the truck k's route. To further specify the position of drone sorties, the variables $z_{aiwje}^k \in \{0, 1\}$ are used, where a value of 1 indicates that the launching node a of sortie (a, w, e) is placed on the route of truck k before node i, and the return node e comes after node j. This means that the drone is airborne while the truck makes deliveries at customers i and j. The continuous variables $a_i^k \geq 0$ specify the earliest arrival time of truck k at node i.

3.2 Mathematical Formulation of the VRPD

The MILP formulation of the VRPD is given in (1) – (19), where the objective function (1) minimizes the makespan under constraints (2) – (19).

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

$$\text{s.t.} \quad \tau \geq a_{n+1}^k \quad : \forall k \in K, \tag{2}$$

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

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

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

$$\sum_{i \in V_L} \sum_{k \in K} x_{ij}^k + \sum_{l \in V_L} \sum_{m \in V_R} \sum_{k \in K} y_{ljm}^k = 1 \quad : \forall j \in V_N, \tag{6}$$

$$2y^k_{iwj} \leq \sum_{h \in V_L} x^k_{hi} + \sum_{l \in V_N} x^k_{lj} \quad : \forall k \in K, i \in V_N, w \in V_N, j \in V_R, \; i \neq j, i \neq w, w \neq j, \tag{7}$$

$$y^k_{0wj} \leq \sum_{h \in V_L} x^k_{hj} \quad : \forall k \in K, w \in V_N, j \in V_R, \tag{8}$$

$$u^k_i - u^k_j \leq (n+1)(1 - x^k_{ij}) \quad : \forall k \in K, i \in V_N, j \in V_R, i \neq j, \tag{9}$$

$$u^k_j - u^k_i \geq 1 - (n+1)(1 - y^k_{iwj}) \quad : \forall k \in K, i \in V_N, w \in V_N, j \in V_R, i \neq j, \tag{10}$$

$$u^k_i - u^k_j \geq 1 - (n+1)p^k_{ij} \quad : \forall k \in K, i \in V_N, j \in V_R, i \neq j, \tag{11}$$

$$u^k_i - u^k_j \leq -1 + (n+1)(1 - p^k_{ij}) \quad : \forall k \in K, i \in V_N, j \in V_R, i \neq j, \tag{12}$$

Constraints (2) are used to compute the arrival time of the last vehicle at the depot. Constraints (3) and (4) describe the flow conservations for the trucks and drones. The requirement that each truck has to start its route from the depot is stated in Constraints (5). Constraints (6) ensure that each customer is served exactly once by either a drone or a truck. Constraints (7) and (8) state the synchronization requirements on trucks and drones, and Constraints (9) – (12) deal with the order of the customer nodes within the routes. In particular, Constraints (11) and (12) connect the decision variables u^k_{ij} and p^k_{ij}.

$$3z^k_{aiwje} \leq y^k_{awe} + p^k_{ai} + p^k_{je} \quad : \forall k \in K, a \in V_L, i, w, j \in V_N, e \in V_R, \tag{13}$$

$$2 + z^k_{aiwje} \geq y^k_{awe} + p^k_{ai} + p^k_{je} \quad : \forall k \in K, a \in V_L, i, w, j \in V_N, e \in V_R, \tag{14}$$

$$M(x^k_{ij} - 1) + a^k_i + t_{ij} \leq a^k_j \quad : \forall k \in K, i \in V_L, j \in V_R, \tag{15}$$

$$M(y^k_{iwj} - 1) + a^k_i + \bar{t}_{iw} \leq a^k_w \quad : \forall k \in K, i \in V_L, w \in V_N, j \in V_R, \tag{16}$$

$$M(y^k_{iwj} - 1) + a^k_w + \bar{t}_{wj} \leq a^k_j \quad : \forall k \in K, i \in V_L, w \in V_N, j \in V_R, \tag{17}$$

Through Constraints (13)–(14), the decision variables p^k_{ij} and y^k_{awe} are used to determine the values of z^k_{aiwje}. In combination with the objective function, Constraints (15)–(17) compute the earliest arrival time at each node.

$$(\bar{d}_{iw} + \bar{d}_{wj}) y^k_{iwj} \leq \mathcal{E} \quad : \forall k \in K, i \in V_L, w \in V_N, j \in V_R, \tag{18}$$

$$\begin{aligned} M(x^k_{ij} - 1) + \sum_{v \in V_N} y^k_{ivj} + \sum_{l \in V_N} \sum_{\substack{m \in V_R, \\ m \neq j}} y^k_{ilm} + \sum_{\substack{p \in V_L, \\ p \neq i}} \sum_{q \in V_N} y^k_{pqj} \\ + \sum_{a \in V_L} \sum_{w \in V_N} \sum_{e \in V_R} z^k_{aiwje} \leq D \quad : \forall k \in K, i \in V_L, j \in V_R. \end{aligned} \tag{19}$$

Adherence to the drone endurance limit and to the maximum number of drones per truck are ensured by Constraints (18) and (19), respectively.

3.3 Valid Inequalities

Due to the NP-hardness of the VRPD, solving the model (1) – (19) with a standard MILP solver can be quite time consuming, except for small-sized instances

[5,15,19–23,27]. In order to enhance the performance of the MILP solvers, we can restrict the solution space by introducing valid inequalities. The following valid inequalities proposed by [21] have shown to be rather efficient, which is why we use them in our computational experiments too.

$$\sum_{i \in V_L} \sum_{j \in V_R} x_{ij}^k t_{ij} \leq a_{n+1}^k \quad : \forall k \in K, \tag{20}$$

$$\frac{1}{D} \sum_{i \in V_L} \sum_{w \in V_N} \sum_{j \in V_R} y_{iwj}^k (\bar{t}_{iw} + \bar{t}_{wj}) \leq a_{n+1}^k \quad : \forall k \in K. \tag{21}$$

In fact, the inequalities (20) - (21) place lower bounds on the arrival times a_{n+1}^k at the depot at the end of each route. Additionally, we use valid inequality (22), which sets an upper bound on the earliest arrival time a_{n+1}^k of the route for each truck k [21]:

$$\sum_{i \in V_L} \sum_{j \in V_R} x_{ij}^k t_{ij} + \sum_{i \in V_L} \sum_{w \in V_N} \sum_{j \in V_R} y_{iwj}^k (\bar{t}_{iw} + \bar{t}_{wj}) \geq a_{n+1}^k \quad : \forall k \in K. \tag{22}$$

3.4 Alternative Objective Functions

The objective function that is typically used in the VRPD models consists in minimizing the makespan [15,22,23]. However, it is plausible to consider costs minimization as alternative objective function (see [4,15,19], and references therein). Indeed, the usage of drones may be beneficial in terms of energy consumption for the following reasons. First, since aerial drones must be relatively lightweight, they cannot carry heavy batteries, which requires them to be as energy efficient as possible. Second, drones are not affected by road congestion and the related inefficiencies that make drones less pollutant. Hence, in this paper, we are interested in investigating the energy consumption of a VRPD solution too.

Energy Consumption Minimization. Let parameters c and $\bar{c}$ denote the energy consumption per distance unit for a truck and a drone, respectively. Then, by (23) we define an objective function for the VRPD to minimize the totally consumed energy. We denote this alternative VRPD by *VRPD-EC*, where *ec* stands for *energy consumption*.

$$min \quad c \sum_{k \in K} \sum_{i \in V_L} \sum_{j \in V_R} x_{ij}^k d_{ij} + \bar{c} \sum_{k \in K} \sum_{i \in V_L} \sum_{w \in V_N} \sum_{j \in V_R} y_{iwj}^k (\bar{d}_{iw} + \bar{d}_{wj}), \tag{23}$$

$$s.t. \ (3) - (19).$$

In terms of constraints for the VRPD-EC, the arrival times are still required for the synchronization of trucks and drones, but the earliest arrival time at the end depot has no effect on the objective value. Instead, the objective function minimizes the sum of the energy consumed on the individual edges by both trucks and drones.

Bi-Objective Optimization. While minimizing makespan or energy consumption as a single objective function, a generated route is not necessarily optimal in both dimensions simultaneously. Consequently, it might be favorable to generate delivery tours that constitute a meaningful trade-off between routes that are completed quickly but consume more energy versus routes that are the most energy efficient but take a significantly longer time to finish. This goal can be achieved in different ways, e.g., with a *bi-objective* optimization model. In the bi-objective VRPD, denoted by *VRPD-BI*, both objectives of minimizing the energy consumption and minimizing the makespan are included in a single model. To present the mathematical model of the VRPD-BI, let S denote the set of all feasible solutions of the VRPD-BI. Then, we define

$$g(s) = \tau \quad \& \quad h(s) = c \sum_{k \in K} \sum_{i \in V_L} \sum_{j \in V_R} x_{ij} d_{ij} + \overline{c} \sum_{k \in K} \sum_{i \in V_L} \sum_{w \in V_N} \sum_{j \in V_R} y^k_{iwj} (\overline{d}_{iw} + \overline{d}_{wj}),$$

where $s \in S$, $g(s)$ defines the latest arrival time of vehicles at the depot (makespan), and $h(s)$ computes the energy consumption of all vehicles for the VRPD-BI. Using this notation, the bi-objective VRPD is formulated as follows:

$$min \ \{(g(s), h(s)) : \quad s.t. \ (2) - (19)\}. \tag{24}$$

For any $s \in S$, the resulting tuples ($g(s)$, $h(s)$) can be assessed and compared with other results in terms of *Pareto* optimality [26].

4 A Simulated Annealing Approach for the Bi-objective VRPD

Due to the NP-hardness of the VRPD, using exact algorithms for solving the VRPD is limited to small-sized instances. To address large VRPD instances, we need to use heuristics and metaheuristics. In this paper, we develop a two-phase *Simulated Annealing* (SA) algorithm for solving the VRPD. Choosing SA is based on its ability in addressing complex combinatorial optimization problems such as the VRPD. Indeed, SA features a mechanism to escape local optima while at the same time it ensures, theoretically, convergence to a global optimum.

We design a two-phase heuristic, where each phase consists of a separate SA algorithm. The first phase creates a set of truck routes, i.e., VRP solutions, through a conventional SA and by solving the classical VRP without considering any drones [18]. In the second phase, the two-phase heuristic inserts drone sorties into the VRP routes to generate VRPD tours. The VRPD solution with the best objective value is returned as the final result of the algorithm. In the following parts of this section, we present each component of our SA algorithm.

4.1 The Metropolis Criterion

A basic SA starts from an initial solution, uses a temperature parameter to control the moves inside the search space, decreases the temperature, and generates

a solution within a neighborhood of the current solution. If the new solution has a better quality, then it is accepted and, if necessary, the incumbent solution is updated. If the new solution is worse than the current solution, then the SA permits to accept it under some conditions. This is an important feature of the SA, known as the *Metropolis* criterion, which is used to determine whether a solution with inferior quality should be accepted. Even though we solve single-objective as well as bi-objective VRPD models, for the sake of completeness, we formally define only the Metropolis criterion that we use in solving the bi-objective VRPD. The case of single-objective VRPD is defined similarly.

Let Ω denote the solutions set of the bi-objective VRPD, and the tuples $(g(\omega), h(\omega))$ consist of the makespan $g(\omega)$ and the energy consumption $h(\omega)$ at $\omega \in \Omega$. Generally, acceptance rules for multi-objective models can be described in terms of *Pareto* efficiency. More precisely, accepting a solution $\omega^{'}$ over ω with a probability of 1 if $\omega^{'} \succ \omega$ is called *strong criterion*, while the *weak criterion* ensures an acceptance if $\omega^{'} \not\prec \omega$, where the sign $\succ$ stands for dominance of a vector versus another one [26].

In this study, to have a better exploration of the search space, we use the weak criterion as the acceptance rule, i.e., all non-dominated solutions are always accepted. Moreover, we use an extended Metropolis criterion to decide if a new solution that is inefficient with respect to the current solution is accepted, i.e.,

$$P(\text{accept } \omega' \text{ as next solution}) = \begin{cases} \exp(\frac{-\Delta_{\omega,\omega'}}{t}), & \text{if } \omega' \prec \omega \ , \\ 1, & \text{if } \omega' \not\prec \omega \ , \end{cases} \tag{25}$$

where *exp* stands for the exponential function, $\omega \in \Omega$ is the current solution vector, and $\omega' \in \mathcal{N}(\omega)$, i.e., a solution in the neighborhood of ω. Moreover, $\Delta_{\omega,\omega'} = \sqrt{(g(\omega') - g(\omega))^2 + (h(\omega') - h(\omega))^2}$, and t is the temperature parameter at the current iteration (for more details, refer to [18] and references therein).

4.2 Simulated Annealing for Generating VRP Routes

The rational of the first phase of the algorithm is to generate a set of VRP tours, which supposedly constitute a good basis for drone insertions in the second phase. For this purpose, a commonly accepted approach consists in using a pool of different initial solutions, which can increase the chance of finding an optimal or a near-optimal VRPD solution. For this purpose, we start with an initial VRP tour x that is generated through a simple *route-first cluster-second* (RFCS) approach, which consists of applying a *Nearest-Neighbor-Heuristic* (NNH) in the route-generation phase, and an equal splitting by number of trucks in the clustering stage [11,20,21]. Afterwards, we continue by generating a random solution x' in the neighborhood $\mathcal{N}(x)$ for which we use a sequence of *2-opt*, *string cross*, *string relocation*, *string exchange*, and *string mix* operators [11,21]. Then, we check the quality of the new route and decide to accept or discard it by using the Metropolis criterion. We iterate the process until the stopping conditions, e.g., time limit or maximum number of iterations, are met. The set of best solutions found by this approach constitute the pool of initial VRP tours.

4.3 Simulated Annealing for the VRPD

In the second phase of the algorithm, summarized in Algorithm 1, drone sorties are integrated into the truck tours, which are generated in the first phase.

Algorithm 1. SA algorithm of the drone inclusion phase.

for each $x \in X$ **do**
  Set $y \leftarrow x$ and $t \leftarrow t_0$
  while $t \geq t_{min}$ or *iteration limit or time limit is not reached* **do**
    Generate a neighbor valid solution $y' \in \mathcal{N}(y)$
    Calculate objective value of the new route y'
    If required, use the Metropolis criterion to check acceptance of the new route
    if y' is accepted **then** $y \leftarrow y'$
    end if
    if $f(y') < f(y^*)$ **then** $y^* \leftarrow y'$
    end if
    Update the temperature and iteration parameters
  end while
end for
return y^*

The algorithm's input is a VRP solution pool X, and the output is y^*, which is the best of the resulting VRPD solutions. The outer loop iterates over the initial VRP solutions in X to generate VRPD routes and to optimize them through an SA. Furthermore, whenever a neighbor solution with a better objective value is found, y^* is updated to save the overall best VRPD solution.

Compared to the VRP, the solution space of the VRPD is highly constrained. Hence, the selection of a valid $y' \in \mathcal{N}(y)$ can be time-consuming. To avoid excessive search for valid neighbor solutions, we impose a limit on the number of attempts such that in case of failure in finding a valid solution, the algorithm continues with the next VRP tour from the set X.

In producing VRPD routes out of VRP truck tours, generating valid neighbor solutions and updating the temperature parameter t are essential parts of Algorithm 1. In the following, we describe these components.

4.4 Drone Sortie Operators

Assume that a set of VRP truck tours is given. Then, we generate valid VRPD routes by integrating drone sorties into the truck routes through applying two sets of operations: *basic drone operations* and *high-level operations*.

Any basic drone operation consists in performing either a *basic sortie insertion*, or a *basic sortie removal* operation. The *basic sortie insertion* operation is applied on a truck tour for a set of launch, customer, and return nodes to launch a drone from the launch node, serve the customer node, and land on the truck at the return node. An insertion operation is accepted only if the generated

route satisfies all constraints of the corresponding VRPD model. The *basic sortie removal* operation works in a similar but in reverse way, i.e., a drone sortie is removed from a given VRPD route. However, in this case, a validation check is not necessary as a drone sortie can always be successfully removed from a route.

Compared to the basic drone operations, the high-level operations define the neighborhoods of a given VRPD solution. In the following, we explain briefly the different high-level operators that we use in the algorithm.

For a given VRP or VRPD route, the *Insert Random* and *Remove Random* operators select one customer node randomly and insert or remove a drone sortie, respectively. The *Delivery Exchange* operator attempts to switch the delivery mode of a drone customer by turning it into a truck customer and vice versa [20]. The *Sortie Modify* operator attempts to modify a randomly selected existing sortie of a random route. The *Insert Best* is a more elaborated operator, which searches for a drone insertion that would result in the highest *savings* in terms of makespan and/or energy consumption through considering all vertices that could be turned into a drone node by adding a sortie without violating the VRPD constraints. To avoid computational operations, the range of the potential launch and return nodes is limited by a *maximum sortie range* parameter [21].

4.5 Cooling Schedule Determination

Since the cooling schedule has considerable impact on the performance of an SA algorithm, we propose a mechanism to find a suitable cooling schedule based on probabilities and instance characteristics.

A cooling schedule consists of an initial or start temperature t_0, a minimum temperature threshold t_{min} for the end of the cooling process, and a cooling factor $\delta(t)$, which sets the speed of transition from t_0 to t_{min}. Moreover, we denote the probability for hill-climbing moves at t_0 as P_0 and at t_{min} as P_{min}. Due to the Metropolis acceptance criterion, these values affect the probability for an inferior-quality solution to be accepted by the algorithm. Thereby, they influence the way in which the solution space is explored in the course of the algorithm. Indeed, due to the Metropolis criterion (25), we know that:

$$t = \frac{-\Delta_{\omega,\omega'}}{ln(P)}. \tag{26}$$

Hence, given $\Delta_{\omega,\omega'}$ at two neighbor solutions ω and ω', we can set desired acceptance probability values to the parameter P to find the temperature values t_0 and t_{min}. But, since we do not know the $\Delta_{\omega,\omega'}$ values at t_0 and t_{min} beforehand, we adopt a proxy value Δ_{Avg}, which is supposed to represent the average distance between neighboring solutions in the solution space. Therefore, before setting values to t_0 and t_{min}, the value of Δ_{Avg} is computed through preprocessing the solution space of each problem instance [12]. More precisely, the mechanism consists of estimating Δ_{Avg} by executing 100 sequential random moves from an initial solution and aggregating the resulting distances. Although the computed value is just an estimation of the true value, the resulting value supposedly

reflects some characteristics of the considered problem instance. To describe the procedure, we define $\delta(t) = \theta \cdot t$, where $\theta < 1$ is the cooling parameter and t is the current temperature level at a given iteration of the SA. Algorithm 2 provides the pseudocode of the cooling schedule mechanism, where the calculation of Δ_{Avg} in lines $1-6$ is followed by assigning values to t_0, t_{min}, and θ. Moreover, I_{total} and i_{max} are user-specified values indicating the total number of iterations of the heuristic and the number of iterations per temperature step, respectively.

Algorithm 2. Determination of the cooling schedule parameters.

1: Set $\Delta_{Avg} \leftarrow 0$
2: **for** q = 1 to 100 **do**
3: $\omega' \leftarrow$ *neighbor solution* $\in \mathcal{N}(\omega)$
4: $\Delta_{\omega,\omega'} \leftarrow f(\omega') - f(\omega)$ and $\Delta_{Avg} = \Delta_{Avg} + (\Delta_{\omega,\omega'} - \Delta_{Avg})/q$
5: Set $\omega \leftarrow \omega'$
6: **end for**
7: $t_0 \leftarrow -\Delta_{Avg}/ln(P_0)$ and $t_{min} \leftarrow -\Delta_{Avg}/ln(P_{min})$
8: $\theta \leftarrow exp\left(ln(\frac{t_{min}}{t_0})/\frac{I_{total}}{i_{max}}\right)$
9: **return** t_0, t_{min}, and θ

5 Computational Experiments and Numerical Results

We have conducted extensive computational experiments in two main parts: benchmark investigations and realistic scenarios. In this section, we present the numerical results and our observations.

5.1 Settings of Experiments

We selected instances from Agatz et al. [1], which are well-known in the literature, and are in two types: the *uniform* and the *single-center* (for details, refer to [1]). For the sake of an effective presentation of the results, we partition the selected instances into two classes of *small instances*, with between 5 and 9 nodes, and *larger instances* ranging from 10 to 100 nodes. In addition to the number and distribution of customers, the instances are characterized by the following parameters: the number of trucks $|K|$, the number of drones per truck D, the truck speed v, the parameter α (where, drone speed $\bar{v}$ is related to v by $\bar{v} = \alpha \cdot v$), and the relative drone endurance $\mathcal{E}_r$, which is specified as follows:

$$\mathcal{E} = \mathcal{E}_r \cdot max(E) \ : \ \mathcal{E}_r \in [0, 2], \tag{27}$$

where $max(E)$ is the longest Euclidean distance between any two nodes [22]. A value of $\mathcal{E}_r$ close to zero signifies a very limited drone endurance, whereas no endurance limit is imposed with $\mathcal{E}_r = 2$.

Regarding the benchmark experiments, we use the instance parameters $|K| \in \{1, 2, 3\}$, $D \in \{1, 2, 3\}$, $\alpha \in \{1, 2, 3\}$, $\mathcal{E}_r \in \{0.4, 0.6, 1, 2\}$, and $v = 1$. For each

instance size and distribution type (either uniform or single-center), we selected 5 instances. Each of them is combined with the various instance parameters, resulting in $3 \cdot 3 \cdot 3 \cdot 4 = 108$ instance combinations. In total, this results in 5400 VRPD instances for the small benchmark experiments and the same amount of instances for the larger benchmark experiments.

The realistic scenarios, tested on single- as well as bi-objective VRPD, are composed of instances containing 50, 75, and 100 nodes, and we used the parameters $|K| \in \{1, 2, 3\}$, $D \in \{1, 2, 3\}$, and different set of values for α, $\mathcal{E}_r$, and v (for details, refer to Sect. 5.4).

The Simulated Annealing parameters, which affect the efficiency of the algorithm, were determined by preliminary experiments using various problem instances from [1] and the TSPLIB[2]. These parameters are used by the cooling schedule mechanism explained in Sect. 4.5. For the truck route generation SA (respectively, drone inclusion SA), the value of P_0, P_{min}, I_{total} are set to 0.0464, 0.0, and $1000 \cdot n^{1.4}$ (respectively, 0.5, 0.0, and 20000).

We conducted all computational experiments on the *High Performance Computing* (HPC) cluster of the *RPTU Kaiserslautern-Landau* (Germany). The runs were executed on Intel XEON CPUs with a base frequency of 2.60 GHz. On the software side, we used the Java SE 11 JDK on Linux, having access to 32 GB of RAM per process. Since the second phase of our algorithm is the most time-consuming part, we set a time limit of 10 minutes per run. Each instance is ran 5 times, adding up to in total 62100 runs of the algorithm.

To benchmark the obtained values for the small instances, we compare the results of the SA with optimal values of the MILP (1) – (22) formulation of the VRPD. For this purpose, the model is implemented in Java and solved by the commercial solver Gurobi Optimizer in version 8.1.1 [8].

5.2 Small Instances

We used small benchmark instances, with $5 \leq n \leq 9$, to assess the performance of our algorithm in solving the VRPD, for which the solutions obtained by the SA are compared to those provided by the MILP solver Gurobi Optimizer [8].

Table 1 shows the comparative results for the small benchmark instances, broken down by the number of nodes $|V|$. As indicated in the second and third columns of Table 1, the SA algorithm solves a majority of the instances to optimality. These columns present the percentage of instances for which the SA has reached the optimal objective value in at least one and in all of the five runs.

According to the numerical results, even in cases where the algorithm does not reach the optima, the obtained solutions approximate closely the optimal objective values, i.e., with a small optimality gap up to 2.09% on average.

5.3 Larger Instances

Due to the complexity of the VRPD, the MILP solvers cannot find the optima of the larger VRPD instances within a reasonable computation time. Hence, this

[2] Available at http://elib.zib.de/pub/mp-testdata/tsp/tsplib/tsp/.

Table 1. Percentage of small benchmark instances optimally solved by the heuristic, average objective value gap, and runtime comparison with the solver.

$\|V\|$	Opt. found	Opt. in all runs	Gap	Avg. runtime [s]	
				solver	heuristic
5	93.70%	93.70%	0.50%	0.41	0.88
6	96.20%	95.93%	0.35%	1.67	1.64
7	90.19%	89.63%	0.83%	19.85	2.32
8	84.72%	83.06%	1.59%	157.88	2.88
9	71.94%	68.89%	2.09%	998.98	3.36
Total	87.35%	86.24%	1.07%	235.76	2.22

Table 2. Average computation time for each instance size and type as well as percentage of the instances for which the runtime limit (600 s) is reached.

	Instance type	Instance size $\|V\|$				
		10	20	50	75	100
Runtime [s]	single-center	4.41	17.51	170.30	363.94	545.10
	uniform	4.12	16.44	134.52	359.84	476.09
Limit reached	single-center	0.00%	0.00%	1.41%	15.19%	59.81%
	uniform	0.00%	0.00%	1.22%	19.48%	43.41%

section is devoted to presenting our results and observations from solving larger VRPD instances with our SA algorithm. To this aim, for each instance size, Table 2 presents the average runtime (in seconds) as well as ratio of runs for which the time limit of 600 seconds is reached. We observe that, by increasing the number of nodes, the average runtimes exhibit a polynomial growth. In addition, we see that, on average, the computation time for the uniform instances is consistently lower than the time required to solve single-center instances.

5.4 Realistic Scenarios

In our further experiments, we investigated using realistic scenarios in the VRPD. For this end, we used our SA algorithm to solve the MILP formulation of the VRPD as a single- as well as a bi-objective optimization problem.

Assumptions on the Realistic Scenarios. The realistic scenarios are based on both the uniform and single-center instances with 50, 75, and 100 customers, which are originally spread on a plane of 100×100 units in case of the uniform instances, and within or nearby around a circle with a radius of 50 units in case of the single-center instances [1]. With the objective of creating realistic conditions, we converted the customer location data to meaningful values in a physical distance unit. For this purpose, we scaled the instance planes to $\frac{1}{3}$ of

their original values, and treated them to be in kilometers, which makes the instances mimic large urban areas where the customers are situated within an area of about 1100 km^2. This constitutes an approximation of metropolitan areas of large cities, e.g., a bit larger than Berlin.

We used average truck speed values that are based on two empirical surveys, which determine average speed in urban areas worldwide. According to the surveys, the average speed of cars in city centers and urban areas are approximately 20.7 km/h and 31.2 km/h, respectively [7,10]. There is less empirical data available for drone speed levels; however, values between 64 to 80 km/h seem to be realistic for state-of-the-art commercial drones [6]. The flight endurance of these kinds of drones typically ranges between 10 and 50 minutes [9].

Along with these values, we define the *high-traffic* and *low-traffic* road congestion scenarios that correspond to the truck speed v of 20 km/h and 30 km/h, respectively. The velocity of the drones in both scenarios is assumed to be 60 km/h, constituting the low end of the maximum speeds of real drones. The endurance of the drones is assumed to be 20 minutes of flight time, which corresponds to a maximum distance of 20 km at the drone speed of 60 km/h.

To compute the energy consumption of drones, we use a linear model, empirically validated with an ArduCopter Hexa-B hexacopter drone [5], where for a fixed average payload of 1.5 kg, the energy consumption is expressed as follows:

$$E_{drone}(t) = (0.217 \cdot 1.5 + 0.185) \cdot t = 0.5105 \cdot t \text{ [kWh]}. \tag{28}$$

Concerning trucks, we use the energy consumption specification of an electric truck, i.e., Volkswagen e-Crafter, which is available to the mass-market and the corresponding detailed information is publicly available. More precisely, in the *New European Driving Cycle* (NEDC) [14], considering an average speed of 33.4 km/h, Volkswagen e-Crafter's energy consumption with a payload of 975 kg is stated as 21.54 kWh/100 km. Our selected values for low- and high-traffic scenarios are good approximations of these empirical values. Taking into account the empirical data, for a given value of the speed parameter v, the deduced energy consumption of an electric delivery truck is computed as follows:

$$E_{truck}(t) = 21.54 \cdot \frac{v}{100} \cdot t \text{ [kWh]}. \tag{29}$$

Makespan and Energy Consumption Results. Fig. 1 visualizes the results of the experiments with the realistic instances, using the makespan minimization, the energy consumption minimization, and the bi-objective minimization as optimization objective functions. In this figure, the average results, both in terms of makespan and energy consumption, are displayed for the different numbers of trucks, numbers of drones, and (low- as well as high-) traffic scenarios. The VRP tours, i.e., $D = 0$, are used as baseline to measure improvements that we gain by the insertion of drones.

As expected, while minimizing the makespan, there is a significant reduction in total mission time. Moreover, the marginal improvement per vehicle is diminishing for an increasing number of drones per truck and for a growing number

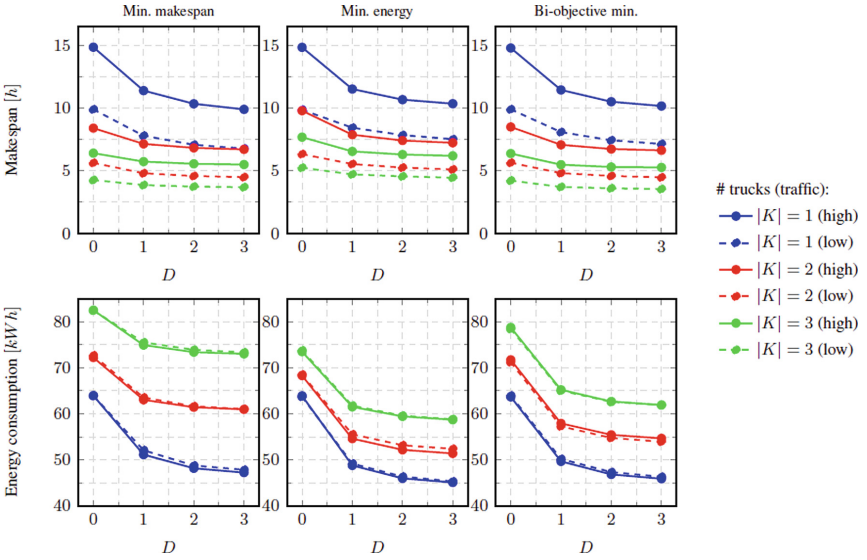


Fig. 1. Makespan and energy consumption results for the realistic scenarios.

of trucks. Indeed, the benefits of drone assistance are more pronounced in the high-traffic than in the low-traffic scenario because drones' trajectories are not affected by road congestion. Moreover, the minimal-makespan routes exhibit a drastic reduction in energy consumption when featuring drone operations.

While minimizing the energy consumption, we observe that the makespan values are slightly higher than what we obtain by using the minimal-makespan objective function (compare the left-side and middle parts of Fig. 1). Indeed, when $D \in \{2, 3\}$, the baseline optimal values (i.e., delivery routes in absence of drones) are significantly higher with the minimal energy consumption objective. This leads to higher overall makespan values; even though, the SA algorithm reduces them by a similar order of magnitude when drone sorties are integrated.

By solving the bi-objective VRPD, we expect to generate solutions that exhibit a compromise between a lower makespan and a reduced energy consumption. In fact, the resulting energy consumption values from the bi-objective optimization model, as visualized in Fig. 1, turn out smaller than the respective results from the makespan minimization model and larger than those obtained by solving the single-objective VRPD minimizing energy consumption. However, we observed that some makespan values of the bi-objective model not only drop below the respective results of the energy-consumption objective, but also smaller than outcomes of the makespan objective, even though the gap is quite small. This means that, with these instances, the makespan objective values are surprisingly dominated by the respective bi-objective results.

Finally, as we observe in Fig. 1, the energy consumed by the truck-drone fleet in high traffic is up to 1% lower than in low traffic, i.e., in terms of energy consumption, using drones may be equally beneficial in both scenarios.

6 Conclusion

This paper focuses on solving the *Vehicle Routing Problem with Drones* (VRPD) with a *Simulated Annealing* algorithm, where the VRPD is studied with different objective functions: minimization of makespan, minimization of energy consumption, and both of them simultaneously as a bi-objective optimization model.

Through extensive computational experiments on benchmark instances, we showed the efficiency of our SA algorithm that integrated an automated cooling schedule mechanism. Moreover, we investigated reductions in mission time and in total energy consumption under realistic scenarios, e.g., realistic vehicle speeds for two congestion levels, and inclusion of energy consumption models for both delivery vehicle types. Comparing the solutions, which are obtained by solving the single- and bi-objective VRPD models, we find a range of different outcomes confirming the benefits of drone-assisted deliveries also in realistic scenarios, not only in improving makespan, but also in reduction of energy consumption.

The presented algorithm solves the bi-objective optimization problem in reasonable computation time; however, developing an efficient exact algorithm is also a plausible future research direction to pursue. The research in this direction is in progress, and the results will be published in the future.

References

1. Agatz, N., Bouman, P., Schmidt, M.: Optimization approaches for the traveling salesman problem with drone. Transp. Sci. **52**(4), 965–981 (2018)
2. Borghetti, F., Caballini, C., Carboni, A., Grossato, G., Maja, R., Barabino, B.: The use of drones for last-mile delivery: a numerical case study in Milan. Italy. Sustainabil. **14**, 1–19 (2022)
3. Murray, C.C., Chu, A.G.: The flying sidekick traveling salesman problem: optimization of drone-assisted parcel delivery. Transp. Res. Part C Emerg. Technol. **54**, 86–109 (2015)
4. Chiang, W.C., Li, Y., Shang, J., Urban, T.L.: Impact of drone delivery on sustainability and cost: realizing the UAV potential through vehicle routing optimization. Appl. Energy **242**, 1164–1175 (2019)
5. Dorling, K., Heinrichs, J., Messier, G.G., Magierowski, S.: Vehicle routing problems for drone delivery. IEEE Trans. Syst. Man Cybern. Syst. **47**(1), 70–85 (2016)
6. Flynt, J.: How fast can drones fly? Max speed of drones. 3D Insider (2018). https://3dinsider.com/drone-speed/
7. kfzteile24 GmbH: Die besten und schlechtesten städte zum autofahren 2017. https://www.kfzteile24.de/die-besten-und-schlechtesten-staedte-zum-autofahren-de
8. Gurobi Optimization: Gurobi Optimizer Reference Manual (2019)
9. Haller, L.: 7 drones that can lift heavy weights (2021). http://www.dronesglobe.com/guide/heavy-lift-drones/
10. INRIX: Inrix 2018 global traffic scorecard. INRIX, Inc. (2019). http://inrix.com/scorecard/
11. Irnich, S., Toth, P., Vigo, D.: The family of vehicle routing problems. In: Toth, P., Vigo, D. (eds.) Vehicle Routing: Problems, Methods, and Applications, chap. 1, pp. 1–33. Society for Industrial and Applied Mathematics (2014)

12. Kaku, I., Xiao, Y., Xia, G.: The deterministic annealing algorithms for vehicle routing problems. Int. J. Smart Eng. Syst. Des. **5**(4), 327–339 (2003)
13. Kloster, K., Vigo, D., Moeini, M., Wendt, O.: The multiple traveling salesman problem in presence of drone- and robot-supported packet stations. Eur. J. Oper. Res. **305**(2), 630–643 (2023)
14. Kroher, T.: VW e-Crafter: Erste Testfahrt mit dem neuen Elektro-Transporter. ADAC ev. (2018). https://www.adac.de/der-adac/motorwelt/reportagen-berichte/auto-innovation/elektro-transporter-vw-e-crafter/
15. Macrina, G., Pugliese, L.D.P., Guerriero, F., Laporte, G.: Drone-aided routing: a literature review. Transp. Res. Part C. Emerg. Technol. 1–25 (2020). https://doi.org/10.1016/j.trc.2020.102762
16. Masmoudi, M.A., Mancini, S., Baldacci, R., Kuo, Y.H.: Vehicle routing problems with drones equipped with multi-package payload compartments. Transp. Res. Part E. Logist. Transp. Rev. **164**(2) (2022)
17. Moeini, M., Salewski, H.: A genetic algorithm for solving the Truck-Drone-ATV routing problem. Adv. Intell. Syst. Comput. **991**, 1023–1032 (2020)
18. Nikolaev, A.G., Jacobson, S.H.: Simulated annealing. In: Gendreau, M., Potvin, J.Y. (eds.) Handbook of Metaheuristics, vol. 146, pp. 1–39. Springer, Boston (2010). https://doi.org/10.1007/978-1-4419-1665-5_1
19. Pugliese, L.D.P., Macrina, G., Guerriero, F.: Trucks and drones cooperation in the last-mile delivery process. Networks **78**(4), 371–399 (2021)
20. Schermer, D., Moeini, M., Wendt, O.: Algorithms for solving the vehicle routing problem with drones. In: Nguyen, N.T., Hoang, D.H., Hong, T.-P., Pham, H., Trawiński, B. (eds.) ACIIDS 2018. LNCS (LNAI), vol. 10751, pp. 352–361. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-75417-8_33
21. Schermer, D., Moeini, M., Wendt, O.: A variable neighborhood search algorithm for solving the vehicle routing problem with drones. Technical Paper (2018). https://bisor.wiwi.uni-kl.de/en/research/publications/
22. Schermer, D., Moeini, M., Wendt, O.: A hybrid VNS/Tabu search algorithm for solving the vehicle routing problem with drones and EN route operations. Comput. Oper. Res. **109**, 134–158 (2019)
23. Schermer, D., Moeini, M., Wendt, O.: A matheuristic for the vehicle routing problem with drones and its variants. Transp. Res. Part C. Emerg. Technol. **106**, 166–204 (2019)
24. Schermer, D., Moeini, M., Wendt, O.: A branch-and-cut approach and alternative formulations for the traveling salesman problem with drone. Networks **76**, 164–186 (2020)
25. Schermer, D., Moeini, M., Wendt, O.: The drone-assisted traveling salesman problem with robot stations. In: Proceedings of the 53rd Hawaii International Conference on System Sciences, pp. 1308–1317 (2020)
26. Serafini, P.: Simulated annealing for multi objective optimization problems. In: Tzeng, G.H., Wang, H.F., Wen, U.P., Yu, P.L. (eds.) Multiple Criteria Decision Making, pp. 283–292. Springer, New York (1994). https://doi.org/10.1007/978-1-4612-2666-6_29
27. Tamke, F., Buscher, U.: A branch-and-cut algorithm for the vehicle routing problem with drones. Transp. Res. Part B: Methodol. **144**, 174–203 (2021)
28. Wang, X., Poikonen, S., Golden, B.: The vehicle routing problem with drones: several worst-case results. Optim. Lett. **11**(4), 679–697 (2017)
29. Yu, S., Puchinger, J., Sun, S.: Two-echelon urban deliveries using autonomous vehicles. Transp. Res. Part E Logist. Transp. Rev. **141** (2020). https://doi.org/10.1016/j.tre.2020.102018