Greedy Algorithms for Scheduling Package Delivery with Multiple Drones

Francesco Betti Sorbelli
University of Perugia
Italy
francesco.bettisorbelli@unipg.it

Federico Corò
Missouri University of Science and Technology
USA
federico.coro@mst.edu

Sajal K. Das
Missouri University of Science and Technology
USA
sdas@mst.edu

Lorenzo Palazzetti
University of Florence
Italy
lorenzo.palazzetti@unifi.it

Cristina M. Pinotti
University of Perugia
Italy
cristina.pinotti@unipg.it

ABSTRACT

Unmanned Aerial Vehicles (or drones) can be used for a myriad of civil applications, such as search and rescue, precision agriculture, or last-mile package delivery. Interestingly, the cooperation between drones and ground vehicles (trucks) can even enhance the quality of service. In this paper, we investigate the symbiosis among a truck and multiple drones in a last-mile package delivery scenario, introducing the Multiple Drone-Delivery Scheduling Problem (MDSP). From the main depot, a truck takes care of transporting a team of drones that will be used to deliver packages to customers. Each delivery is associated with a drone's energy cost, a reward that characterizes the priority of the delivery, and a time interval representing the launch and rendezvous times from and to the truck. The objective of MDSP is to find an optimal scheduling for the drones that maximizes the overall reward subject to the drone's battery capacity while ensuring that the same drone performs deliveries whose time intervals do not intersect. After showing that MDSP is an NP-hard problem, we devise an optimal Integer Linear Programming (ILP) formulation for it. Consequently, we design a heuristic algorithm for the single drone case and two more heuristic algorithms for the multiple drone case. Finally, we thoroughly compare the performance of our presented algorithms on different synthetic datasets.

CCS CONCEPTS

• **Mathematics of computing** → *Combinatorial optimization*; • **Computing methodologies → Optimization algorithms**; • **Theory of computation → Scheduling algorithms**.

This work was partially supported by NSF grants CNS-1818942, OAC-1725755, OAC-2104078, and SCC-1952045; and also partially supported by "HALY-ID" project funded by the European Union's Horizon 2020 under grant agreement ICT-AGRI-FOOD no. 862665, no. 862671, and from MIPAAF .

ICDCN 2022, January 4–7, 2022, Delhi, AA, India
© 2022 Association for Computing Machinery.
ACM ISBN 978-1-4503-9560-1/22/01…$15.00
https://doi.org/10.1145/3491003.3491028

KEYWORDS

Drone; truck; last-mile delivery system

ACM Reference Format:
Francesco Betti Sorbelli, Federico Corò, Sajal K. Das, Lorenzo Palazzetti, and Cristina M. Pinotti. 2022. Greedy Algorithms for Scheduling Package Delivery with Multiple Drones. In *23rd International Conference on Distributed Computing and Networking (ICDCN 2022), January 4–7, 2022, Delhi, AA, India.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3491003.3491028

1 INTRODUCTION

Since when the Unmanned Aerial Vehicles (*drones*) have been developed, both the research and industrial communities have started to exploit this new technology in a plethora of applications. In particular, drones can be efficiently and effectively used for search and rescue operations over disastrous areas hit by an earthquake [4, 26], for observing and monitoring crops in precision agriculture contexts [13, 20], for localizing and monitoring missing people [2, 15], or for shipping parcels to customers to accelerate the delivery process in a package delivery context [14, 24, 25, 31, 36]. Interestingly, the collaboration between drones and trucks, relying on their respective capabilities and mobility, can significantly improve the quality of service, especially in the last-mile delivery scenario [9, 12].

In this work, we investigate the cooperation between a *truck* and one or multiple drones for last-mile package delivery. With the help of drones, delivery companies can perform more deliveries and hence extend their revenue and business [23]. In fact, drones can deliver small packages quickly as they can easily traverse difficult terrain [1], often using shorter routes not otherwise possible for trucks. In this scenario, the drones fly (starting from the truck) to the assigned locations and deliver the packages, then leave and meet again with the truck to perform new deliveries. However, significant challenges arise due to such constraints as the drones' high energy consumption and their *current* inability to simultaneously serve multiple customers at a time [10, 17]. The premise under consideration is that a delivery company has to make several deliveries to the customers in a city by relying on a truck carrying a fleet of drones with the same capabilities. In this paper, we study this problem by assuming that the main depot knows the locations of the customers to visit and the consequent roads to travel for the truck.

Therefore, before leaving the depot, the delivery company plans the drones' flying sub-routes to accelerate the deliveries, taking into account not only the energy used by the drones but also the revenue generated. Since drones have limited battery capacity, *a delivery has a cost* in terms of the energy required, and this limits the number of deliveries that a drone can perform. Furthermore, there can be *conflicts among deliveries* if they cannot be accomplished by the same drone, i.e., if the corresponding sub-routes, and hence their time intervals, intersect each other. In addition, *each delivery has an associated reward* that characterizes its importance (e.g., a higher reward means higher priority). Hence, given the truck's route in the city, *the goal is to plan a scheduling for the drones such that the total reward is maximized subject to the constraints of limited drone's battery capacity and the conflicts among deliveries.*

We make the following contributions in this paper.

- We propose an optimization problem, called Multiple Drone-Delivery Scheduling Problem (MDSP), and prove its *NP*-hardness, devising also the optimal Integer Linear Programming (ILP) formulation which is only suitable for small-sized instances.
- We design a heuristic algorithm for the single drone case, and two heuristic algorithms for the multiple drones case, which can be run for larger instances.
- We extensively compare the performance of our algorithms on synthetic datasets.

The remainder of this paper is structured as follows. Section 2 surveys the related works. Section 3 introduces the MDSP showing its *NP*-hardness, and proposes the optimal ILP formulation. Sections 4 devises algorithms for MDSP for the single and multiple drones cases. Section 5 evaluates the algorithms on synthetic datasets. Finally, Section 6 offers conclusions and future research directions.

2 RELATED WORK

This section reviews the literature related to the problem of delivering packages with the help of trucks and drones. We categorize the existing works based on whether the deliveries are done via both trucks and drones, or only by drones in which case the truck is only a carrier.

2.1 Delivery by Both Trucks and Drones

The truck-drone cooperation in a last-mile delivery scenario has been investigated for the first time when the flying sidekicks traveling salesman problem (FSTSP) has been introduced in [21]. The FSTSP is a particular case of the original TSP, where drones take off from the truck, fly to deliver packages to customers, and then go back to the truck in another place. Both the vehicles can perform deliveries, but each has to stop waiting for the other at the rendezvous location. The authors proposed an optimal mixed-integer linear programming formulation (MILP) and two heuristic solutions for solving the FSTSP. A similar approach is also studied by the same authors in [22] for multiple drones.

A greedy heuristic algorithm for the truck-and-drone delivery system is proposed in [7]. The heuristic starts with a solution for the truck only, which comprises nodes (i.e., locations) provided by a TSP instance. Then it greedily builds short sub-paths for the drones by excluding some nodes from the truck's path in order to reduce the overall tour's length in terms of time. Importantly, such a tour must serve all the customers' locations. Finally, the truck's route preserves the order of the nodes from the initial solution, and the drone flies between the neighboring nodes.

In [8], the authors considered a scenario with multiple rechargeable drones. Drones can only carry one package at a time and have to return to the truck's roof to charge their battery after each delivery. It is assumed that the speed is the same for both drones and trucks, but their mobility is different – the drones move according to the Euclidean metric, while the trucks follow the Manhattan metric. The authors presented a heuristic to solve the problem of finding a good schedule for all drones and trucks that minimizes the average delivery time of the packages.

In the delivery system in [28], the truck has a potentially large capacity for carrying packages, but it travels at slow speeds in urban areas due to road intersections or traffic jam. On the other hand, drones are faster and not restricted to street networks, but their range and carrying capacity are limited. So, to minimize the total tour duration to serve all customers the authors proposed an optimal MILP formulation based on timely synchronizing the truck and drone flows. They also introduced a dynamic programming recursion based on an exact branch-and-price approach capable of optimally solving small instances.

The delivery with a truck and drones relying on a rechargeable station for the drones is presented in [16]. The station can furnish a large number of drones, and it is located near customer areas and away from the distribution center. The authors showed that this problem can be divided into TSP and parallel identical machine scheduling (PMS) problems. Through this approach, they successfully reduce the complexity of the problem and obtain an exact solution. A hybrid approach is presented in [35], where the authors propose to simultaneously employ trucks, truck-carried drones, and independent drones to construct a more efficient truck-drone parcel delivery system. A novel routing and scheduling algorithm are proposed to solve the hybrid parcel delivery problem.

2.2 Delivery by Drones Only

A similar work to ours is proposed in [3], where the authors consider a predefined route for the truck, and the problem is to optimize the planning of the drone's flights to/from the truck while serving the customers. Unlike us, however, their goal is to determine the launch and meeting points between drones and trucks. On the other hand, in our work these points are provided in input, and our goal is to determine a subset of deliveries to maximize the revenue. Furthermore, these points are calculated in such a way that the intervals generated do not intersect with each other, thus making all deliveries conflict-free. Importantly, in [3] the authors assume that drones do not have a battery constraint. While their goal is to reduce total makespan (i.e., the time difference between the start and end of a delivery sequence), our goal is to maximize the revenue/reward for deliveries.

The authors in [27] studied the cooperation between trucks and drones from a different point of view. In this work, the truck is only concerned with transporting the drones in order to improve the delivery system and make it more efficient. The objective is to determine a valid combination of the drone's characteristics (i.e., speed and range) to synchronously let them cooperate with the truck avoiding useless stops at locations. Only an optimal MILP formulation is proposed for solving this problem.

In the work in [19], the truck-drone tandem is proposed for cooperatively performing all the required deliveries. Once any delivery has been performed, a drone can immediately recharge its battery. This means that all deliveries are feasible and, accordingly, no scheduling is required.

In [6] a clustering is proposed to find truck's intermediate locations in which it can stop and send drones to deliver packages to near locations. Such clustering aims at minimizing the overall makespan.

3 PROBLEM FORMULATION

In this section, we first introduce the system and delivery models, and then define the Multiple Drone-Delivery Scheduling Problem (MDSP). We also show that MDSP is *NP*-hard.

3.1 System Model

Let A be the 2-D *delivery area* representing our last-mile delivery (application) scenario. Let $\psi \in A$ be the *depot* from where the deliveries start. The position of the depot is located at (x_ψ, y_ψ), assumed to be at the origin of the Cartesian coordinate system in $(0, 0)$. Such a delivery area also comprises roads and customer positions. At the depot, a *truck* is in charge of transporting a fleet of m *drones* $d_1, \ldots, d_m$ with the same capabilities used for deliveries in A. The truck *does not* perform deliveries; it only carries the drones within A.

Let ρ_j be a straight road delimited by two endpoints λ_j and λ_{j+1}, where $j = 0, \ldots, r$ and $\{\lambda_1, \ldots, \lambda_r\} \subset A$. The truck leaves the depot ψ visiting the first endpoint λ_1 along the straight road segment $\rho_0 = \overline{\psi\lambda_1}$, then the next endpoint λ_2 along the segment $\rho_1 = \overline{\lambda_1\lambda_2}$, and so on, up to the last endpoint λ_r, and eventually going back to the depot ψ. These segments make a closed path (*cycle*) C formed by the sequence of endpoints $\lambda_0, \lambda_1, \ldots, \lambda_r, \lambda_{r+1}$ such that $\lambda_0 = \lambda_{r+1} = \psi$. Clearly, the $r + 1$ contiguous roads $\rho_0, \ldots, \rho_r$ define C.

Let $D = \{\delta_1, \ldots, \delta_n\} \subset A$ be the set of n distinct points or locations to be served (i.e., the *customers*) by the drones. Each customer's (delivery) point δ_i has a pair of coordinates $(x_{\delta_i}, y_{\delta_i}) \in A$, for $i = 1, \ldots, n$. Figure 1 illustrates the delivery area A with roads and customers.

3.2 Delivery Model

In this paper we assume that drones can deliver a single package at a time due to stringent payload constraints. A drone's delivery is performed by planning a sub-flight that passes through three points. Specifically, the drones take off (with packages) from the truck which continues to drive in the city, delivers packages to the customers which are waiting for them, and return to the rendezvous locations with the truck again. For each customer's location δ_i, let δ_i^L and δ_i^R be respectively the *launch point* and *rendezvous point* of the drone. Note that δ_i^L and δ_i^R can lie on different roads. In general, δ_i^L lies on road ρ_j while δ_i^R lies on road ρ_z, where $0 \le j \le z \le r$ (see Figure 1). For instance, the truck in Figure 2 travels from λ_j to

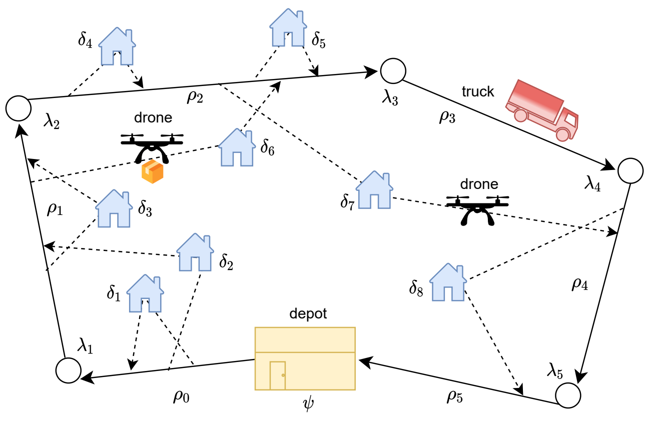

Figure 1: An example delivery area A with 6 roads ρ_j and 8 customers δ_i to serve. The depot ψ is at $(0, 0)$; The truck's path is the solid line, while the drones' paths are the dashed lines.

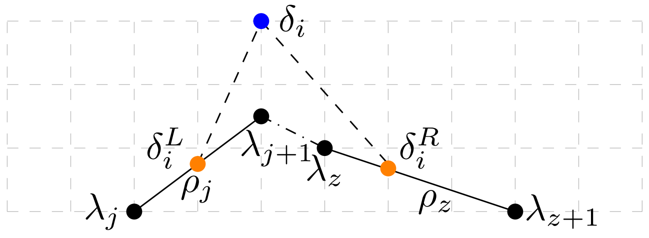

Figure 2: Two non-adjacent roads with a delivery to do: the launch and rendezvous points are highlighted.

δ_i^L carrying the drone, which in turn takes off at δ_i^L flying towards δ_i delivering the package, and finally continues flying towards δ_i^R. In the meanwhile, the truck continues its ground route reaching other endpoints. When both the truck and the drone arrive at δ_i^R, the truck gathers again the drone and they continue to travel up to point λ_{z+1}. Note that, in order to save time, the truck do not travel back and forth for picking up the drones.

Let $w_i \ge 0$ be the *energy cost* (or *weight*) for a drone in terms of energy spent to perform a single delivery δ_i flying from/to the truck. Let $B \ge 0$ be the drone's *energy budget* in terms of battery capacity that limits the number of possible deliveries that it can do. Note that if the drone's current residual energy is not enough for additional flights, it is not possible to perform further deliveries unless a new battery is swapped. Nevertheless, we assume that drones have to rely only on the initial single battery charge.

Let $p_i \ge 0$ be the *reward* for executing a delivery δ_i. The meaning of the reward characterizes *premium users* having higher priority than *regular users*. For instance, delivery companies offer different subscriptions according to the following rule: *the more you pay, the faster you receive your parcels.* In our context, the higher the priority of delivery δ_i, the larger is the reward value p_i. Hence, in order to satisfy the premium users, the scheduling for the drones needs to prioritize the deliveries, guaranteeing first the ones belonging to the premium users (because they have more reward) and second the ones from regular users.

Let $t \geq 0$ denote the *time*. Let $t_0 = 0$ be the *initial time* when the truck leaves the depot at location $\lambda_0 = \psi$, for performing deliveries within the area A. Similarly, let t_{r+1} be the *final time* of the delivery application at $\lambda_{r+1} = \psi$. It is important to observe that the truck travels its route along a predefined sequence of endpoints to be visited exactly once, in a specific direction. Hence, if the truck plans to travel through the endpoints λ_i and λ_j, with $i < j$, then $t_i < t_j$, with $0 \leq i < j \leq r+1$. For any delivery δ_i, let t_i^L be the *launch time* for the drone from point δ_i^L, and let t_i^R be the *rendezvous time* at point δ_i^R, where $t_0 \leq t_i^L < t_i^R \leq t_{r+1}$. Finally, let $\tau_i = t_i^R - t_i^L$ be the *span time* of delivery in δ_i. Consequently, for a given δ_i, let $I_i = [t_i^L, t_i^R]$ define the drone's *delivery interval time* that determines its flying time-window for delivery.

Let $I = \{I_1, \ldots, I_n\}$ be the *interval set*, where $1 \leq i \leq n$, associated with the deliveries within A. Two intervals I_i and I_j are said to be *compatible* if their intersection is empty, i.e., $I_i \cap I_j = \emptyset$, for $i \neq j$; otherwise, the two intervals are in *conflict*. In other words, I_i and I_j are compatible if $t_i^R < t_j^L$ or $t_j^R < t_i^L$. A subset $S \subseteq I$ is said to be *compatible* if $I_i \cap I_j = \emptyset$ for any pair $I_i, I_j \in S$. This means that a drone can perform any subset of such deliveries as long as it has enough battery. Precisely, a given compatible $S \subseteq I$ is *feasible* if the energy cost $C(S) = \sum_{I_i \in S} w_i \leq B$ (the energy budget). The *reward* of a feasible set S is $\mathcal{P}(S) = \sum_{I_i \in S} p_i$.

Recall that m is the number of drones. Two feasible subsets $S_p, S_q \subseteq I$ can be *assigned* to two drones d_p and d_q, with $1 \leq p \neq q \leq m$, if $S_p \cap S_q = \emptyset$. Assuming that $S = \{S_1, \ldots, S_m\}$ consists of m feasible sets assigned to the drones $\{d_1, \ldots, d_m\}$, the *overall reward* $\mathcal{P}(S)$ is defined as the sum of the reward of each drone, i.e., $\mathcal{P}(S) = \sum_{j=1}^{m} \sum_{I_i \in S_j} p_i$.

3.3 Problem Definition

Let us now formally define the delivery scheduling problem.

Problem 1 (**Multiple Drone-Delivery Scheduling Problem (MDSP)**). *Let $\delta_1, \ldots, \delta_n$ be the set of n deliveries, m the number of drones, and B the drone's battery budget. The objective of MDSP is to find a family $S^* = \{S_1^*, \ldots, S_m^*\} \subseteq I$ of m feasible subsets with $S_p^* \cap S_q^* = \emptyset$, for $1 \leq p \neq q \leq m$, such that the overall reward $\mathcal{P}(S)$ is maximized. Specifically,*

$$S^* = \underset{S=\{S_1,\ldots,S_m\}\subseteq I}{\arg\max} \; \mathcal{P}(S)$$
$$\textit{such that } C(S_i) \leq B \; \forall i = 1, \ldots, m$$

In the following we show the *NP*-hardness of MDSP, even for the single drone case.

An alternative way to define the set of deliveries and constraints is to observe that the set of intervals I can be described as an *interval graph*, which is a special case of *chordal graphs*. In MDSP, the set of deliveries D determines the set of intervals I that can be pairwise compatible or in conflict with respect to their starting and ending times. Given an instance of MDSP, we can visualize the intervals and their compatibility along a temporal line. Figure 3 illustrates the intervals corresponding to Figure 1, in which I_2 is in conflict with both I_1 and I_3, whereas I_1 and I_3 are compatible.

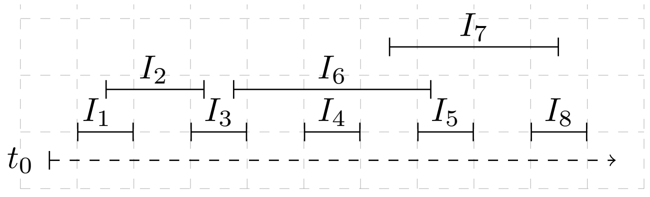

Figure 3: Delivery intervals corresponding to Fig. 1.

A simple way to figure out if two intervals are in conflict or not is by considering the interval graph representation. Formally, in the *undirected interval graph*, $G = (V, E)$, the vertex-set $V = I$, where each vertex uniquely corresponds to an interval $I_i \in I$, for $i = 1, \ldots, n$; and an edge $(I_i, I_j) \in E$ indicates that the deliveries δ_i and δ_j are not compatible, implying $I_i \cap I_j \neq \emptyset$. For example, in Figure 3, I_1 and I_3 can be executed without conflicts, i.e., $(I_1, I_3) \notin E$; while I_2 is not compatible with both I_1 and I_3, implying $(I_2, I_1), (I_2, I_3) \in E$. Note that vertex I_6 has four conflicts because its degree is 4.

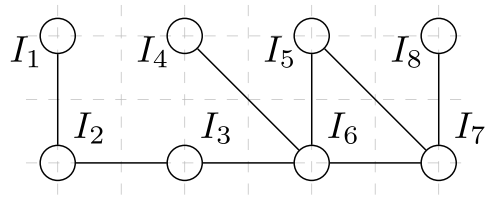

Figure 4: Interval graph G for the example in Fig. 1.

In the following we demonstrate that MDSP is an *NP*-hard problem by showing that the classic 0–1 Knapsack Problem (KP) is a special case of MDSP when intervals are conflict-free.

Theorem 1. *MDSP is NP-hard.*

Proof (Sketch). Our approach is by reduction from KP, which is known to be *NP*-hard [18] and defined as follows.

Given a set $X = \{1, \ldots, n\}$ of n items, each associated with a cost w_i and reward p_i, and a knapsack of capacity c, the KP is to find a subset of X that maximizes the sum of the rewards, satisfying the capacity constraint. Given an instance of KP, we translate it as an instance of MDSP as follows: we first set the energy budget of MDSP equal to the capacity constraint of the knapsack, i.e., $B = c$. Then we create a set of deliveries D starting from the set of items X as follows: for each item $x_i \in X$, create a delivery $\delta_i \in D$ with equal cost and reward. Hence, we can observe that the deliveries are pairwise compatible in the corresponding MDSP instance. Thus, a solution for KP is a solution for MDSP and vice versa. This reduction takes polynomial time. Hence, proven. □

3.4 The Optimal Algorithm

For optimally solving MDSP in the general case with m drones, we can use an ILP formulation. We enumerate the deliveries as $\mathcal{N} = \{1, \ldots, n\}$, and drones as $\mathcal{M} = \{1, \ldots, m\}$. Let $x_{ij} \in \{0, 1\}$ be a decision variable that is 1 if the delivery $j \in \mathcal{N}$ is accomplished by the drone $i \in \mathcal{M}$; otherwise it is 0. Hence, the ILP formulation is formed by an objective function and a few constraints as follows:

$$\max \sum_{i=1}^{m} \sum_{j=1}^{n} p_j x_{ij} \tag{1}$$

subject to:

$$\sum_{j=1}^{n} w_j x_{ij} \le B, \qquad \forall i \in \mathcal{M} \tag{2}$$

$$\sum_{i=1}^{m} x_{ij} \le 1, \qquad \forall j \in \mathcal{N} \tag{3}$$

$$x_{ij} + x_{ik} \le 1, \qquad \forall i \in \mathcal{M}; \forall j,k \in \mathcal{N} s.t.\ I_j \cap I_k \neq \emptyset \tag{4}$$

$$x_{ij} \in \{0,1\}, \qquad \forall i \in \mathcal{M}, \forall j \in \mathcal{N} \tag{5}$$

The objective function in Expression (1) maximizes the overall reward using a fleet of m drones. Constraint (2) states that each drone has an energy budget B in terms of battery; Constraint (3) enforces each delivery to be performed by no more than one drone; and Constraint (4) ensures that deliveries can be performed by the same drone only if they are not in conflict.

Since the above ILP formulation, for solving MDSP, is only suitable for small instances in input, in the following we propose three time-efficient heuristic algorithms suitable for larger instances in input, in scenarios involving a single or multiple drones. From now on, we will interchangeably use the terms 'delivery' and 'interval'.

4 PROPOSED ALGORITHMS

This section proposes a heuristic algorithm for MDSP with a single drone, namely, Max Ratio Single drone (Mr-S), and two heuristic algorithms for MDSP with multiple drones, namely, Max Clique Multiple drones (Mc-M) and Max Ratio Multiple drones (Mr-M).

4.1 The Mr-S Algorithm

The Mr-S is a heuristic to solve MDSP with a single drone, requiring $O(n \log n)$ time and $O(n)$ space. The pseudo-code of Mr-S is given in Algorithm 1.

Algorithm 1: The Mr-S Algorithm

```
1 sort(I) s.t. $\frac{p_1}{w_1} \ge \ldots \ge \frac{p_n}{w_n}$
2 $SOL \leftarrow \emptyset$
3 for $i \in 1, \ldots, n$ do
4     if is-augmentable($I_i, SOL$) then
5         $SOL \leftarrow SOL \cup I_i$
6 return $SOL$
```

In Mr-S we first sort the intervals in non-increasing order by the ratio $\frac{p_i}{w_i}$ of the reward to the energy cost (Line 1, Algorithm 1). Then, we add to the current solution SOL the best possible interval I_i (i.e., with the largest ratio) that does not create conflicts with the already chosen intervals and also evaluating if the residual energy budget is sufficient. This is done via the `is-augmentable` procedure (Line 4).

Note that the greedy technique in Mr-S that solves MDSP can efficiently compute approximated solutions of the KP (i.e., instances of MDSP without interval conflicts) by exploiting the *submodularity property*. Recall that a set function $\mathcal{F} : 2^X \to \mathbb{R}$ is said to be *submodular* if given a finite set $X = \{x_1, \ldots, x_n\}$, for any $S \subseteq T \subseteq X$ and $x \in X \setminus T$ it holds $\mathcal{F}(S \cup x) - \mathcal{F}(S) \ge \mathcal{F}(T \cup x) - \mathcal{F}(T)$. Unfortunately, our objective function $\mathcal{P}(S)$ is not submodular due to the presence of intervals that overlap/intersect each other. If $\mathcal{P}(S)$ had been submodular, it would have been possible to obtain a solution that guarantees at least $1 - e^{-1} \approx 0.63$ of the optimum [33] taking overall $O(n \log n)$ time. For MDSP, due to the presence of conflicts among intervals, the submodularity property does not hold, and hence the approximation ratio of an algorithm that exploits the strategy in Mr-S is unbounded. We can explain this issue by taking into account the example in Figure 5, showing that the solution can be arbitrarily bad with respect to the optimum. Indeed, if we use Mr-S to solve MDSP, in presence of two intervals I_1 and I_2 in conflict, where the span of I_1 is very large, and assuming reward $p_2 = \Delta > 1$ and $0 < \epsilon < 1$, Mr-S would return I_1 (because $\frac{1}{1-\epsilon} > \frac{\Delta}{\Delta+\epsilon}$) gaining a reward of only 1. Therefore, the ratio of the reward obtained by Mr-S to the reward obtained by the optimum is $\frac{1}{\Delta}$. Such a ratio can be arbitrarily small, and hence unbounded. This example can easily be extended to n intervals such that $I_2, \ldots, I_n$ are compatible with each other, and I_1 is in conflict with the other $n-1$ intervals.

The Mr-S algorithm requires $O(n \log n)$ time due to the time required for sorting, and $O(n)$ space.

4.2 The Mc-M Algorithm

The heuristic Mc-M solves MDSP with multiple drones in $O(m(n \log n + h(n))$ time and $O(n)$ space, where $h(n)$ is the time required by a subroutine used in it. The pseudo-code of Mc-M is given in Algorithm 2.

Algorithm 2: The Mc-M Algorithm

```
1 $\hat{I} \leftarrow I, m' \leftarrow m, SOL \leftarrow \emptyset$
2 while $\hat{I} \neq \emptyset$ do
3     $\{S_1, \ldots, S_\omega\} \leftarrow$ create-subsets($\hat{I}$)
4     sort($S_i$) s.t. $\mathcal{P}(S_i) \ge \mathcal{P}(S_{i+1})$
5     $SOL \leftarrow SOL \cup \{S_1, \ldots, S_{\min\{\omega, m'\}}\}$
6     $\hat{I} \leftarrow \hat{I} \setminus \bigcup\{S_i\}$
7     $m' \leftarrow m' - \min\{\omega, m'\}$
8 return $SOL$
```

In Mc-M we sequentially perform a partitioning, depending on the size of the *maximum clique* ω, of the current graph induced by the residual intervals. Then, we assign a drone for each subpartition. This is done via the `create-subsets` procedure (Line 3, Algorithm 2). Finally, we find a global solution comprising the previously computed solutions.

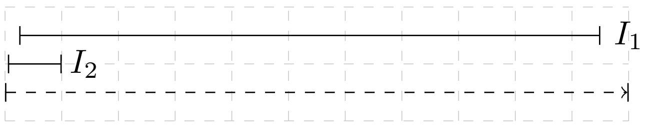

Figure 5: Instance where the submodularity property does not hold for Mr-S. Here, we have $p_1 = 1, w_1 = 1 - \epsilon$ for I_1, and $p_2 = \Delta, w_2 = \Delta + \epsilon$ for I_2.

The set $\hat{I}$ and the number m' of available drones are first initialized (Line 1). Next, until $\hat{I}$ is not empty (Line 2), an optimal partitioning invoking the `create-subsets` procedure is performed to generate $S_1, \ldots, S_\omega$ (Line 3). Since $\hat{I}$ is an interval graph and therefore a chordal graph, the size of the maximum clique ω can be computed in polynomial time [11]. With this knowledge, we divide the set of intervals into several conflict-free subsets based on the number ω. In fact, on a clique of size k all the vertices are in conflict, and hence k drones are needed for conflict-free deliveries. We remark that finding the size of the maximum clique ω in general graphs is *NP*-hard. Nevertheless, if the graph G is specifically an interval graph, ω can be quickly computed in polynomial time.

The most important part of Mc-M resides in the procedure `create-subsets` (Line 3). To efficiently implement `create-subsets`, we initially sort the deliveries (and hence the time intervals) by the launch and rendezvous times in non-decreasing order, into two different ordered sets. Numbers assigned to drones can be labeled as $1, \ldots, \omega$. Then, we consider a pointer i at the beginning of each ordered set. We also create a min-heap data structure initialized with only one element, which is the first number. If the element indexed by the pointer is a launch time, we extract the minimum number from the heap and associate such value with interval i and then move the pointer one position, otherwise, if the element indexed by the pointer is a rendezvous time, we insert the number of the interval i in the heap and move the pointer one position. After the extraction, if the heap is empty, we then insert a new number (i.e., the next available integer not used until now). Having said this, the performance of `create-subsets` can further be improved by taking into account the eventual residual energy budget. Since the returned solution belongs only to one subset, say $S_i \subseteq C_i$, the sum of energy costs in S_i may be less than B. This means that the solution can be augmented by including intervals from other subsets. This approach can be applied selectively by ensuring that the interval to be added to the current solution does not create any conflict. A possible strategy is to greedily pick the *compatible* interval having the maximum ratio between the reward and cost.

Once the partitioning is completed, for each subset C_i, find a subset $S_i \subseteq C_i$ with maximum reward such that $C(S_i) \leq B$. Note that, subsets S_i for each i are feasible solutions. It is important to observe that finding an optimal set S_i is equivalent to solving a KP with budget B on the elements C_i, which is an *NP*-hard problem. However, in our implementation, we take into account only approximated solutions of KP in polynomial time exploiting the submodularity property.

Then, the $\{S_1, \ldots, S_\omega\}$ are sorted in non-decreasing order by the total reward (Line 4). Now, depending on the number of current available groups ω and drones m', we assign the best $\min\{\omega, m'\}$ groups to drones (Line 5). After that, we update the current solution SOL and the number of available drones (Lines 6–7). Finally, the solution is returned (Line 8).

4.2.1 *Random Interval Graphs.* In this section, we provide a few interesting considerations about the value of ω in random graphs. Since the delivery may occur anywhere in the delivery area A, we may consider random interval graphs associated with the delivery intervals I, and study the expected value of ω. If the interval graph is dense, i.e., the number of edges $|E| = O(n^2)$, where n is the number of vertices, then $\omega = O(n)$ [30]. In the general case of random graphs in which the intervals and their associated lengths have equal probability and they are all independent from each other, $\omega = \frac{n}{2} + o(n)$ [29]. Other estimations of ω can also be determined with respect to the length of the intervals. For example, if these interval lengths are uniformly distributed in $[0, r]$, there exist some results. In particular, if $r = f(1/n)$, then $\omega = (1 \pm \epsilon)\frac{\log n}{\log \log n}$, and if $r = f(\frac{\log n}{n})$, then $\omega = O(\log n)$ [30].

The Mc-M algorithm takes $O(m(n \log n + h(n)))$ time for invoking m times the `create-subsets` algorithm, and $O(n)$ space.

4.3 The Mr-M Algorithm

The Mr-M is a heuristic algorithm to solve MDSP with multiple drones, and requires $O(m(n \log n))$ time and $O(n)$ space. The pseudo-code of Mr-M is given in Algorithm 3.

Algorithm 3: The Mr-M Algorithm

```
1 $\hat{I} \leftarrow I$
2 for $i \in 1, \ldots, m$ do
3     $S_i \leftarrow$ Mr-S($\hat{I}$)
4     $\hat{I} \leftarrow \hat{I} \setminus S_i$
5 return $SOL \leftarrow \{S_1, \ldots, S_m\}$
```

In Mr-M we sequentially perform Mr-S (for each drone) on the current set of intervals not assigned to drones yet. After that, we return a global solution which is the union of all the determined solutions in the previous steps.

Initially, the set $\hat{I}$ that characterizes the current residual intervals is initialized with all the deliveries I (Line 1, Algorithm 3). After that, we iteratively (Line 2) invoke Mr-S on $\hat{I}$ (Line 3) for m times (since the fleet of drones), which contains the intervals that have not been assigned to drones yet. At the end of the loop there are intermediate solutions S_i for drone d_i. Consequently, the current set of intervals available for the remaining drones is decreased (Line 4). Eventually, we return the SOL as a global solution which comprises the union of the computed sub-solutions during the previous steps (Line 5).

Table 1 compares the time and space complexities of our proposed algorithms for single and multiple drones. Note that, in this paper, we implemented Mc-M exploiting the submodularity property, and hence its time complexity is $O(m(n \log n))$, and hence polynomial.

Table 1: Comparison between the algorithms.

Algorithm	Time Complexity	Space Complexity
Mr-S	$O(n \log n)$	$O(n)$
Mc-M	$O(m(n \log n + h(n)))$	$O(n)$
Mr-M	$O(m(n \log n))$	$O(n)$

5 PERFORMANCE EVALUATION

In this section, we compare the performance, in terms of obtained reward, of the three proposed heuristics for solving MDSP with a single drone and multiple drones cases.

5.1 The Settings

The last-mile delivery area A is a circle of radius 5 km in which the depot is located at $(0, 0)$. Then, the truck's route is randomly generated inside A. The locations of the $n = \{25, 50, 75, 100\}$ deliveries are uniformly generated at random in A. The truck carries a total of $m = \{1, 3, 5\}$ drones that fly at a constant speed $d_s = 20$ m/s [32]. We set the drone's battery $B = 5000$ kJ [32]. We set the drone's payload to $d_p = 5$ kg [32]. With respect to these parameters, the energy cost w_i for performing a delivery is computed according to the energy model presented in [31, 32], which depends on the distance to travel, the total mass of the drone plus the payload, and the drone's speed. We assume for the truck a whole duration trip of 30000 s = 8 h20 m, which is a reasonable amount of time for a single working day [5].

About the rewards assigned to each delivery, we randomly generate them as an integer number between [1, 100] according to the Zipf distribution [34] varying the θ parameter in $[0, 0.4, 0.8, 1.0]$. When $\theta = 0$, the rewards are uniformly distributed in $[1, 100]$; when $\theta \geq 0.8$, there are a few rewards with a large probability and many rewards occurring a few times. The energy costs and the span times are uniformly generated according to the Uniform distribution, assuming four *span/energy configurations* Σ_i (from Σ_1 with low variability to Σ_4 with high variability). Specifically, in configuration Σ_1 the maximum energy cost is 2500 kJ and the maximum span time is 1500 s, in Σ_2 this pair is 5000 kJ and 10000 s, and so on. Note that, Σ_1 and Σ_2 always admit feasible deliveries, while the other two do not. Moreover, Σ_4 allows very long span times as large as the duration of the truck's tour. Finally, once the span length τ_i is generated, the launch time t_i^L is uniformly generated between t_0 and $t_{r+1} - \tau_i$.

Table 2 summarizes the used parameters.

Table 2: Used input parameters.

Par.	Description	Value	Unit
d_s	drone's speed	20	m/s
d_p	drone's payload	5	kg
B	drone's budget	5000	kJ
n	number of deliveries	$[25, 50, 75, 100]$	–
m	number of drones	$[1, 3, 5]$	–
t_{r+1}	final time	30000	s
p	max reward	100	–
w	max energy cost	2500, 5000, 7500, 30000	kJ
τ	max span time	1500, 10000, 20000, 30000	s
θ	Zipf parameter	$[0, 0.4, 0.8, 1.0]$	–

When evaluating the performance of our algorithms presented in Section 4, we compare Mr-S with respect to the optimum solution in the single drone scenario, and we compare Mc-M and Mr-M with respect to the optimum solutions in the multiple drones scenario. Optimal solutions are obtained from the ILP formulation, which works regardless of the number of drones.

Also, as a reference for comparison (as baseline), we propose three greedy heuristic algorithms for single and multiple drone scenarios. Specifically, we propose the following greedy heuristics that repeatedly select:

GeRt	Greedy Earliest Rendezvous Time: the compatible interval with the earliest rendezvous time t_i^R.
GSw	Greedy Smallest Weight: the compatible interval with the smallest energy cost w_i.
GLp	Greedy Largest Profit: the compatible interval with the largest reward p_i.

In the case of multiple drones, these greedy heuristics are repeated on the residual subset of deliveries not yet assigned to the drones.

5.2 Experiment Results

Figure 6 illustrates the performance, in terms of collected rewards, of our algorithms on synthetic data. Specifically, in the first row, we present the results for MDSP with $m = 1$ (single drone), whereas in the other two rows we illustrate the results for MDSP with $m = 3$ and $m = 5$ drones, respectively. The x-axis reports the four different configurations *span/energy* Σ_i, while the y-axis shows the ratio between the reward reported by any algorithm and that by the optimum algorithm. Clearly, the ratio is less than or equals to 1.

5.2.1 Single drone scenario. When $m = 1$, the best performing algorithms are Mr-S and GLp, which both take into account the rewards in their delivery selection rule. Instead, GSw and GeRt, whose selection rule is unlinked to the rewards, show the worst performance. This trend is particularly emphasized for the Zipf parameter $\theta = 0.8$. In this case, there are a few large-value rewards and many small-value rewards. Thus, GLp is facilitated in their choice. When the rewards are uniformly distributed ($\theta = 0$), Mr-S guarantees better performance than GLp, probably because in its selection it considers also the cost (weight) and not only blindly the reward. Instead, in presence of unbalanced rewards ($\theta = 0.8$), GLp exhibits the best solution because the algorithm can select the most rewardable intervals without any other consideration. The performance jump of GLp is high when $\theta = 0.8$.

Interestingly, Mr-S shows interesting performance with respect to the evaluated configuration. In fact, being completely free in the interval selection and considering both the energy and the reward criteria, Mr-S is the best solution. This holds until the variability is low (e.g., Σ_1), the length of the intervals is short, and the greedy strategy of picking intervals with the largest reward to cost ratio is a winning strategy. In other words, in configurations with low variability Mr-S clearly outperforms GLp regardless of the value of θ and the number of deliveries n. However, with higher variability (e.g., Σ_4), the intervals become larger (as well as the number of intersections) and Mr-S could incur into particularly bad situations like the one depicted in Figure 5. This explains the poor performance of Mr-S when the variability increases. Observe that Σ_1 is the configuration that most likely represents a real-world scenario. Generally, very long intervals for drones (e.g., like those in Σ_4) are not recommended since they create too many intersections. Therefore, even though GLp collects more reward than Mr-S in many configurations, Mr-S shows a very good performance in configurations closer to the reality ($\geq$ 95% of the optimum solution).

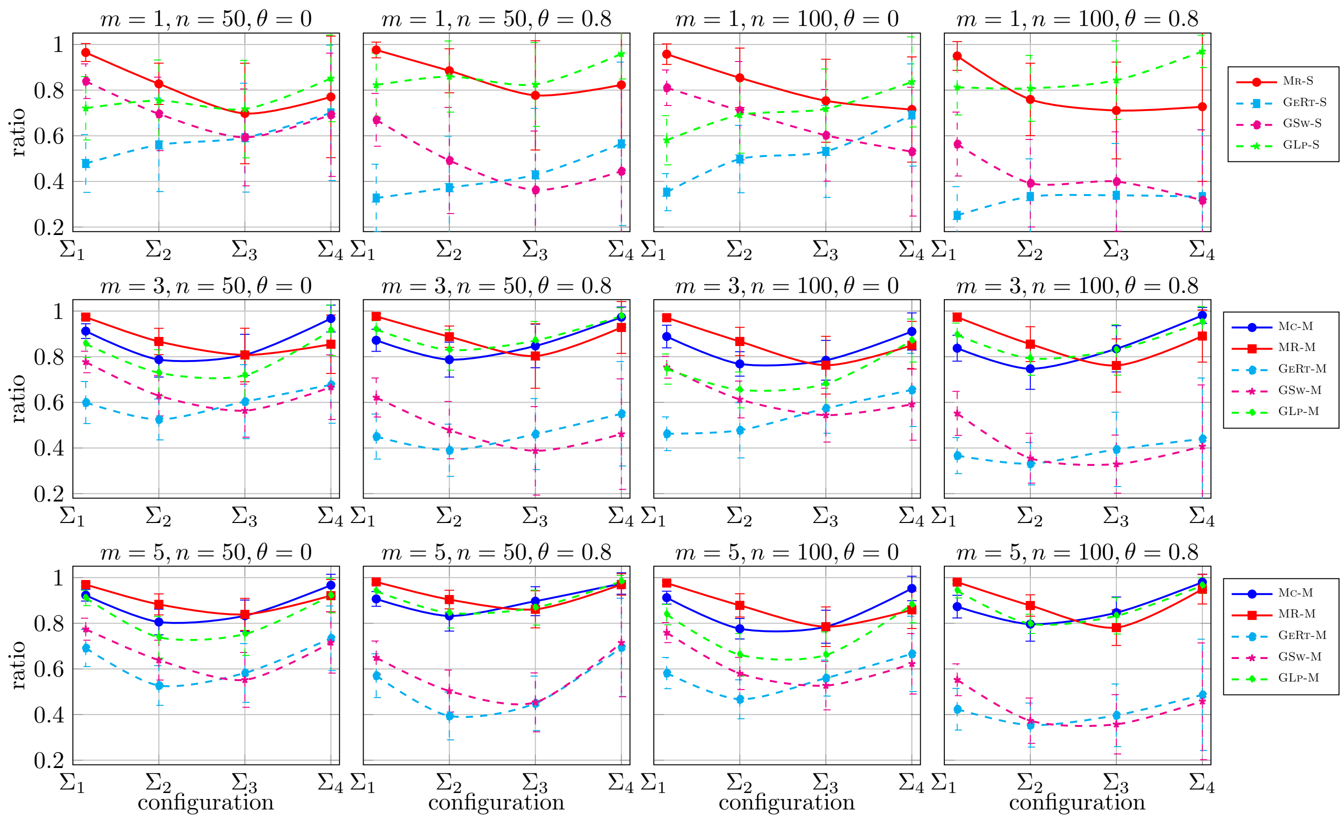

Figure 6: Performance evaluation of our algorithms on a synthetic data-set. The first row compares algorithms with a single drone, while the other two rows compare with multiple drones. Three greedy heuristics, used as a baseline, have suffixes depending on whether they are deployed on a single drone (-S) or multiple drones (-M).

5.2.2 Multiple drones scenario. In the multiple drone scenario, i.e., for $m = \{3, 5\}$, we observe that GERT-M and GSW-M perform poorly, as in the single drone scenario. Accordingly, strategies that pick intervals (deliveries) taking into account either weight or rendezvous time do not perform well at all. The MR-M has a very good performance collecting more than 98% in the lowest variability configurations, i.e., Σ_1. Differently from the single drone case when MR-S was employed, MR-M performs well even when the variability is high. We get this behavior because most likely intervals in conflict can still be assigned to different drones, and the effect of having many intersections is less important. Concerning MC-M, although it does not guarantee any approximation ratio, it performs quite well, and its collected reward is always above 80% of the optimum. It is interesting to observe that both MR-M and MC-M almost always outperform GLP-M. Finally, we observe that the performance of our proposed algorithms with $m = 5$ drones is slightly better than those with $m = 3$. This is probably because the residual intervals can be assigned to other independent drones without conflicts.

6 CONCLUSION

This paper investigated the Multiple Drone-Delivery Scheduling Problem (MDSP) to study the cooperation between a truck and multiple drones in a last-mile package delivery scenario. After showing that MDSP is an *NP*-hard problem, we proposed an optimal ILP formulation that is suitable for small instances in input. Then, for larger instances, we provided three time-efficient heuristic algorithms for the single and multiple drones. Finally, we evaluated the performance of the proposed algorithms on synthetic datasets. As future work, it would be worth designing other effective algorithms as well as searching for guaranteed approximation bounds for them. Moreover, it would be interesting to investigate multi-depot multi-truck scenarios or to allow drones to perform multiple deliveries at the same time or recharge their battery on charging stations. We also plan to study a more realistic dynamic environment, dealing with network communications between the depot and ground-air vehicles, addressing possible delays and new/canceled deliveries. To address these challenges, we plan to develop online strategies to reschedule deliveries on the fly.

REFERENCES

[1] Ross Arnold, Elizabeth Mezzacappa, Melissa Jablonski, Benjamin Abruzzo, and Jonathan Jablonski. 2021. Experimentation for optimization of heterogeneous drone swarm configurations: terrain and distribution. In *Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications III*, Tien Pham and Latasha Solomon (Eds.), Vol. 11746. International Society for Optics and Photonics, SPIE, 523 – 533. https://doi.org/10.1117/12.2585589

[2] Francesco Betti Sorbelli, Cristina M. Pinotti, Simone Silvestri, and Sajal K. Das. 2020. Measurement Errors in Range-based Localization Algorithms for UAVs: Analysis and Experimentation. *IEEE Transactions on Mobile Computing* (2020), 1–1. https://doi.org/10.1109/TMC.2020.3020584

[3] Nils Boysen, Dirk Briskorn, Stefan Fedtke, and Stefan Schwerdfeger. 2018. Drone delivery from trucks: Drone scheduling for given truck routes. *Networks* 72, 4 (2018), 506–527.

[4] Tiziana Calamoneri and Federico Corò. 2020. A Realistic Model for Rescue Operations after an Earthquake. In *Q2SWinet '20: Proceedings of the 16th ACM Symposium on QoS and Security for Wireless and Mobile Networks, Alicante, Spain, November 16-20, 2020*, Cheng Li and Ahmed Mostefaoui (Eds.). ACM, 123–126. https://doi.org/10.1145/3416013.3426453

[5] James F Campbell, Don Sweeney, and Juan Zhang. 2017. Strategic design for delivery with trucks and drones. *Supply Chain Analytics Report SCMA (04 2017)* (2017).

[6] Yong Sik Chang and Hyun Jung Lee. 2018. Optimal delivery routing with wider drone-delivery areas along a shorter truck-route. *Expert Systems with Applications* 104 (2018), 307–317.

[7] Gloria Cerasela Crişan and Elena Nechita. 2019. On a cooperative truck-and-drone delivery system. *Procedia Computer Science* 159 (2019), 38–47.

[8] Rami Daknama and Elisabeth Kraus. 2017. Vehicle Routing with Drones. *CoRR* abs/1705.06431 (2017). arXiv:1705.06431 http://arxiv.org/abs/1705.06431

[9] Dyutimoy Nirupam Das, Rohan Sewani, Junwei Wang, and Manoj Kumar Tiwari. 2021. Synchronized truck and drone routing in package delivery logistics. *IEEE Trans. Intell. Transp. Syst.* 22, 9 (2021), 5772–5782. https://doi.org/10.1109/TITS.2020.2992549

[10] Iman Dayarian, Martin Savelsbergh, and John-Paul Clarke. 2020. Same-day delivery with drone resupply. *Transportation Science* 54, 1 (2020), 229–249.

[11] Fănică Gavril. 1972. Algorithms for minimum coloring, maximum clique, minimum covering by cliques, and maximum independent set of a chordal graph. *SIAM J. Comput.* 1, 2 (1972), 180–187.

[12] Andy M Ham. 2018. Integrated scheduling of m-truck, m-drone, and m-depot constrained by time-window, drop-pickup, and m-visit using constraint programming. *Transportation Research Part C: Emerging Technologies* 91 (2018), 1–14.

[13] Atharva Kadethankar, Neelam Sinha, Vinayaka Hegde, and Abhishek Burman. 2021. Signature Feature Marking Enhanced IRM Framework for Drone Image Analysis in Precision Agriculture. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2021, Toronto, ON, Canada, June 6-11, 2021*. IEEE, 2385–2389. https://doi.org/10.1109/ICASSP39728.2021.9413577

[14] Arindam Khanda, Federico Corò, Francesco Betti Sorbelli, Cristina M. Pinotti, and Sajal K. Das. 2021. Efficient Route Selection for Drone-based Delivery Under Time-varying Dynamics. In *18th International Conference on Mobile Ad-Hoc and Smart Systems (MASS)*. IEEE.

[15] Aakash Khochare, Yogesh Simmhan, Francesco Betti Sorbelli, and Sajal K. Das. 2021. Heuristic Algorithms for Co-scheduling of Edge Analytics and Routes for UAV Fleet Missions. In *40th IEEE Conference on Computer Communications, INFOCOM 2021, Vancouver, BC, Canada, May 10-13, 2021*. IEEE, 1–10. https://doi.org/10.1109/INFOCOM42981.2021.9488740

[16] Sungwoo Kim and Ilkyeong Moon. 2018. Traveling salesman problem with a drone station. *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 49, 1 (2018), 42–52.

[17] Jaihyun Lee. 2017. Optimization of a modular drone delivery system. In *2017 Annual IEEE International Systems Conference, SysCon 2017, Montreal, QC, Canada, April 24-27, 2017*. IEEE, 1–8. https://doi.org/10.1109/SYSCON.2017.7934790

[18] Silvano Martello, David Pisinger, and Paolo Toth. 2000. New trends in exact algorithms for the 0–1 knapsack problem. *European Journal of Operational Research* 123, 2 (2000), 325–332.

[19] Neil Mathew, Stephen L Smith, and Steven L Waslander. 2015. Planning paths for package delivery in heterogeneous multirobot teams. *IEEE Transactions on Automation Science and Engineering* 12, 4 (2015), 1298–1308.

[20] UM Rao Mogili and BBVL Deepak. 2018. Review on application of drone systems in precision agriculture. *Procedia computer science* 133 (2018), 502–509.

[21] Chase C Murray and Amanda G Chu. 2015. The flying sidekick traveling salesman problem: Optimization of drone-assisted parcel delivery. *Transportation Research Part C: Emerging Technologies* 54 (2015), 86–109.

[22] Chase C Murray and Ritwik Raj. 2020. The multiple flying sidekicks traveling salesman problem: Parcel delivery with multiple drones. *Transportation Research Part C: Emerging Technologies* 110 (2020), 368–398.

[23] Kenzo Nonami. 2016. Drone technology, cutting-edge drone business, and future prospects. *Journal of Robotics and Mechatronics* 28, 3 (2016), 262–272.

[24] Lorenzo Palazzetti. 2021. Routing Drones Being Aware of Wind Conditions: a Case Study. In *2021 17th International Conference on Distributed Computing in Sensor Systems (DCOSS)*. IEEE.

[25] Lorenzo Palazzetti, Cristina M. Pinotti, and Giulio Rigoni. 2021. A run in the wind: favorable winds make the difference in drone delivery. In *2021 17th International Conference on Distributed Computing in Sensor Systems (DCOSS)*. IEEE.

[26] Agoston Restas et al. 2015. Drone applications for supporting disaster management. *World Journal of Engineering and Technology* 3, 03 (2015), 316.

[27] Robert Rich. 2020. Inverting the Truck-Drone Network Problem to Find Best Case Configuration. *Adv. Oper. Res.* 2020 (2020), 4053983:1–4053983:10. https://doi.org/10.1155/2020/4053983

[28] Roberto Roberti and Mario Ruthmair. 2021. Exact methods for the traveling salesman problem with drone. *Transportation Science* 55, 2 (2021), 315–335.

[29] Edward R. Scheinerman. 1988. Random interval graphs. *Combinatorica* 8, 4 (1988), 357–371.

[30] Edward R Scheinerman. 1990. An evolution of interval graphs. *Discrete Mathematics* 82, 3 (1990), 287–302.

[31] Francesco Betti Sorbelli, Federico Corò, Sajal K. Das, and Cristina M. Pinotti. 2021. Energy-Constrained Delivery of Goods With Drones Under Varying Wind Conditions. *IEEE Trans. Intell. Transp. Syst.* 22, 9 (2021), 6048–6060. https://doi.org/10.1109/TITS.2020.3044420

[32] Joshuah K Stolaroff, Constantine Samaras, Emma R O'Neill, Alia Lubers, Alexandra S Mitchell, and Daniel Ceperley. 2018. Energy use and life cycle greenhouse gas emissions of drones for commercial package delivery. *Nature communications* 9, 1 (2018), 1–13.

[33] Maxim Sviridenko. 2004. A note on maximizing a submodular set function subject to a knapsack constraint. *Operations Research Letters* 32, 1 (2004), 41–43.

[34] Catriona Tullo and James Hurford. 2003. Modelling Zipfian distributions in language. In *Proceedings of language evolution and computation workshop/course at ESSLLI*. 62–75.

[35] Desheng Wang, Peng Hu, Jingxuan Du, Pan Zhou, Tianping Deng, and Menglan Hu. 2019. Routing and scheduling for hybrid truck-drone collaborative parcel delivery with independent and truck-carried drones. *IEEE Internet of Things Journal* 6, 6 (2019), 10483–10495.

[36] Hojoon David Yoo and Stanislav M. Chankov. 2018. Drone-delivery Using Autonomous Mobility: An Innovative Approach to Future Last-mile Delivery Problems. In *2018 IEEE International Conference on Industrial Engineering and Engineering Management, IEEM 2018, Bangkok, Thailand, December 16-19, 2018*. IEEE, 1216–1220. https://doi.org/10.1109/IEEM.2018.8607829