The Dynamic Drone Scheduling Delivery Problem

Giovanni Campuzano(✉), Eduardo Lalla-Ruiz, and Martijn Mes

University of Twente, 7500 Enschede, AE, The Netherlands
{g.f.campuzanoarroyo,e.a.lalla,m.r.k.mes}@utwente.nl

Abstract. Logistics plays an important role in today's last-mile economy. Therefore, companies constantly seek for improving their delivery system towards more efficient and sustainable management of parcel distribution. In this paper, we study the Dynamic Drone Scheduling Delivery Problem. The objective is to minimize the delayed deliveries by a fleet of drones located in a central drone station, taking into account the uncertain arrival of parcels, soft time windows, and energy requirements. We develop a Markov Decision Processes (MDP) formulation and solve it approximately by implementing a value-based Reinforcement Learning (RL) approach. We compare our approach with several heuristic dispatching policies and provide insights into the efficiency of our RL algorithm when facing different delivery scenarios.

Keywords: Drone scheduling · Battery charging · UAV · Last mile · Reinforcement learning

1 Introduction

The last leg of the transportation chain (end-haul), also known as the last mile [20], concerns delivery activities in highly urbanized city centers and controls the reverse and forward flow of goods from producers to consumers. In this context, the logistics sector is currently facing scenarios of unprecedented change with developments in digitization, autonomous vehicles, urbanization, and increasing customer demands. In particular, the rise of e-commerce has resulted in a significant increase in delivery activities, which encompasses up to 30% of the e-logistics costs [23]. To cope with these demanding tasks, companies have been forced to develop more efficient and sustainable parcel distribution systems, to improve their service's quality, diminish greenhouse gas emissions, and reduce operational costs [4].

The industrial sector and scholars have paid close attention to the advancements in artificial intelligence and automation, placing a collaborative effort to develop and include autonomous vehicles in last-mile operations. Consequently, we see innovative delivery systems using non-traditional delivery vehicles, e.g., drones, cargo bikes, and autonomous robots, which provide several advantages to deal with features such as traffic congestion, speed regulations, time windows,

J. de Armas et al. (Eds.): ICCL 2022, LNCS 13557, pp. 260–274, 2022.
https://doi.org/10.1007/978-3-031-16579-5_18

and so forth. In particular, recent advances in technology have increased the popularity of autonomous drones in last-mile delivery. Nowadays, companies are investigating the parallelization the delivery operations by deploying remotely operated drones that help reduce transportation times and logistics costs (e.g., DHL[1], Wings by Google[2], and Air Prime by Amazon[3]). Autonomous drones have also drawn the interest of the transportation research community, where scholars have devoted their efforts to developing combinations of multiple modes of transport that exploit drones' advantages, i.e., avoidance of traffic congestion and infrastructural issues, fast-flying speeds, mobility, and reduced costs [18]. As a result, new research gaps have been identified, where drones can be employed in a wide range of civil applications, for instance, transportation services, emergency and disaster management, agriculture, and industrial warehouses [5]. Nevertheless, in these applications, the main challenge is to cope with the drone's limitations or requirements that might produce unfeasible solutions, e.g., concerning battery life, maximum payload, no-flying zone restrictions, unsafe landing territory, vulnerability to difficult weather conditions, and mandatory signature for receiving packages. For this reason, drones are mainly involved in applications that require ground vehicles or trucks to support drones' delivery operations, for example, the Flying Sidekick Traveling Salesman Problem [10] and the Traveling salesman Problem with Drone [17].

In this work, we study the Dynamic Drone Scheduling Delivery Problem (D-DSDP), where the objective is to optimize the efficient dispatching of a drone fleet from a station for parcel delivery, considering battery levels, time windows, energy consumption, and uncertain parcel arrivals at a central drone station. The main contributions of this work are as follows:

1. We introduce the Dynamic Drone Scheduling Delivery Problem (D-DSDP) and model the stochastic and time-dependent nature of the problem by means of a Markov Decision Processes formulation.
2. We develop a reinforcement learning approach that can solve realistic instances in a reasonable time, using approximate value iteration with a linear value function approximation.
3. We provide insights into the effect of the selection of parcels based on their time windows and drone battery levels and compare the reinforcement learning approach with different assignation heuristics.

The remainder of this paper is structured as follows. Related literature on the D-DSDP is reviewed in Sect. 2. The D-DSDP is formally described and formulated in Sect. 3. A reinforcement learning approach to face the D-DSDP is presented in Sect. 4. The numerical experiments are performed in Sect. 5. Finally, the main conclusions and future works are stated in Sect. 6.

[1] https://www.dhlexpress.be/en/shipping-and-receiving/dhl-parcelcopter/.
[2] https://x.company/projects/wing/.
[3] https://www.amazon.com/Amazon-Prime-Air/b?ie=UTF8&node=8037720011.

2 Literature Review

Literature on drone delivery systems has placed paramount emphasis on transportation systems where drones and trucks work in collaboration [10,11,17,19]. This review, however, focuses on delivery systems where drones are deployed from stations. The reader is referred to [9,12] for reviews on drone applications and drone transportation systems, respectively.

The D-DSDP can be seen as a special case of the Drone Delivery Problem (DDP, [3]), where drones are only allowed to perform round trips to serve customers with their corresponding parcels. The D-DSDP also resembles the Parallel Drone Scheduling TSP (PDSTSP, [10]), with the difference that this delivery system considers only drones. The reader is referred to [13] for a systematic review of the PDSTSP. In [3], authors face the DDP to minimize the makespan and the operational costs. They develop a linear energy consumption model that considers battery and payload to restrict the drone flying range. To solve the DDP, they propose a simulated annealing (SA) algorithm and compare its performance against a MILP formulation. Results show that optimizing the battery weight can provide improvements of up to 80% for the minimum-time DDP. In [2], a MILP formulation and a matheuristic are proposed to solve the multi-objective pick-up and delivery DDP. They consider multiple charging stations and a heterogeneous fleet of drones, where the number of packages that drones can carry is restricted by a maximum weight. A mathematical model that minimizes the maximum distance, number of drones, and the number of batteries is proposed in [22]. This way, the authors develop a MILP formulation that considers drone energy constraints, drone capacity, and time windows to solve instances for up to 10 customers. A pick-up and delivery food distribution case for the DDP is studied in [8], where multiple depots are considered and the demand is predicted in advance. The authors develop a MILP formulation and a heuristic algorithm to solve a dynamic problem. Finally, the reader is referred to [6] for an extensive literature review on truck-and-drone routing problems, mathematical models, problem variants, and heuristic algorithms. Also, the survey [1] provides a relevant literature review on drone distribution systems, which focuses on research issues, solution approaches, and limitations.

We are aware of only one work related to the D-DSDP that studies a delivery system that deploys drones from a central station performing round trips [7]. The authors propose a MILP formulation and a genetic algorithm (GA) for minimizing the number of drones to deliver packages with dynamic arrival and personalized deadlines. Our work mainly differs from it in the solution approach, specific problem features, and the objective function. In our case, first, due to the stochastic nature of this problem, we propose a MDP formulation, which incorporates transportation times, drone energy consumption, battery levels, and uncertain parcel arrivals. Second, we incorporate into our decision space drone charging policies at the central station, while in [7] drones are only allowed to swap the battery for a fully charged one. Third, we focus on minimizing the total costs of the delayed parcels, since we study scenarios with a high-parcel

flow to compare our approach with different heuristic policies, whereas in [7] the authors focus on minimizing the number of drones at the station.

As we can observe, literature on drone stations where drones perform round trips for parcel deliveries is being developed. In this regard, we have found a research gap regarding drone delivery systems from a drone station that consider uncertain parcel arrival, drone energy consumption, transportation times, and time windows, when minimizing delayed deliveries. Therefore, in this paper, we aim at filling this research gap by developing a reinforcement learning approach to solve the D-DSDP with the above-mentioned features and outperform myopic policies.

3 Mathematical Model

3.1 Problem Description

In the D-DSDP, a fleet of drones is located in a central station to deliver parcels in the form of round trips. The central drone station faces uncertain parcel arrivals, and the drones have to meet time window and energy consumption requirements. Figure 1 illustrates a drone station with a fleet of drones and a set of known delivery locations.

The D-DSDP consists of a finite horizon, representing a day, where time is discretized in consecutive time periods $t \in \mathcal{T} = \{1, 2, ..., T\}$, from now on called stages. A set of drones $v \in V = \{1, 2, ..., V\}$ should perform round trips to deliver parcels $j \in J = \{1, 2, ..., J\}$ from a central station to their corresponding customers. As shown in Fig. 1, the distribution area is split into different distance classes $d \in \mathcal{D} = \{1, ..., D\}$, which determine the distance of customers to the central drone station. Furthermore, parcels arrive from outside the system according to a stochastic process with a rate $\lambda_{d,t}, \forall d \in \mathcal{D}, t \in \mathcal{T}$. Every parcel is known in the system since it is picked-up by the truck. The time between pickup of the parcel and delivery at the central drone station is indicated by a release period $r \in \mathcal{R} = \{0, ..., R\}$. Once the release period is equal to zero, the corresponding time window $k \in \mathcal{K} = \{0, ..., K\}$ starts decreasing one unit per stage. That is, time windows are related to a parcel release period r and release periods are related to the current stage t. For instance, a parcel that has $r = 2$ and $k = 1$ can be delivered in only one stage, i.e., we first wait two stages from $r = 2$ to $r = 0$, and then there is a time window of one stage to transport the parcel from $k = 1$ to $k = 0$.

Drones have one unit-load capacity and different battery levels $b \in \mathcal{B} = \{0, 1, 2, ...B\}$, where $b = 0$ means the drone has ran out of battery and can only be sent to recharge the battery. Moreover, transporting a parcel $j \in J$ to a customer class $d \in \mathcal{D}$ takes d units of time and battery level. For instance, if a parcel is sent at time $t = 1$ with $d = 2$, and $b = 3$, this means the drone that was sent to $d = 2$ becomes available at time $t = 3$, i.e., $t + d$, and with a battery level of $b = 3 - 2 = 1$. Drones can only charge their battery at one of the Q charging stations at the central station. We assume that charging the battery by one level requires one time unit, i.e., from b to $b + 1$ when going from stage t to $t + 1$. We

also assume that the external arrival process of parcels to customer class $d \in \mathcal{D}$ at stage $t \in \mathcal{T}$ is independent of the external arrival process of other parcels at other stages.

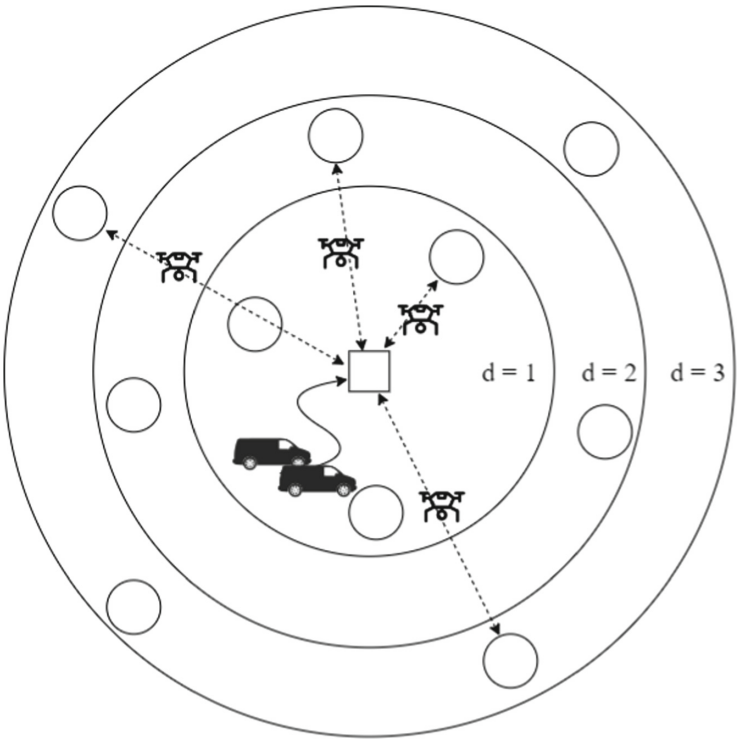

Fig. 1. Illustration of the D-DSDP.

The D-DSDP features two transportation modes. First, we have drones performing round trips from the station to customers. We do not consider transportation costs for drone deliveries, since every parcel has a fixed destination and the system should always bear those costs. Second, if a parcel is not transported by the end of its time window, we consider an alternative transportation mode, i.e., a manned vehicle that performs this transport quickly and at a high cost defined as C^L. Table 1 provides a description of the sets and parameters of the D-DSDP.

3.2 Markov Decision Process Formulation

States: Each period t corresponds to a *stage* in the MDP. Thus, stages are discrete and consecutive. Furthermore, at each stage t, there are $J_{t,d,r,k}$ parcels with distance class d, release stage r, and time window length k at the drone station, and $V_{t,r,b}$ drones with release stage r and battery level b to transport parcels. The state of the system S_t consists of the number of each parcel and

Table 1. Parameters of the D-DSDP.

$\mathcal{T}$:	set of time periods
$\mathcal{D}$:	set of customer classes
$\mathcal{K}$:	set of time periods until expiration of the time window
$\mathcal{R}$:	set of time period until arrival of the parcel
$\mathcal{B}$:	set of battery levels
Q :	number of charging stations at the drone station
$\lambda_{j,t}$:	arrival rate of parcel $j \in J$ at time $t \in \mathcal{T}$
C^L :	Cost of alternative manned transport for late parcels
S_t :	state of the system at period $t \in \mathcal{T}$
V_t :	number of drones available in the system at time $t \in \mathcal{T}$
J_t :	number of parcels in the system at time $t \in \mathcal{T}$

drone type at stage t, as seen in (1). We denote the state space of the system by $\mathcal{S}$, i.e., $S_t \in \mathcal{S}$.

$$S_t = \Big[(J_{t,d,r,k}, V_{t,r,b})\Big]_{\forall d \in \mathcal{D}, r \in \mathcal{R}, k \in \mathcal{K}, b \in \mathcal{B}} \tag{1}$$

Decisions: At each stage $t \in \mathcal{T}$, we have to decide (i) how many drones should transport parcels from the central station and (ii) how many drones should charge the battery. This decision depends on the release time of parcels, the battery level of drones, and the number of drones available to be used. We use variables $\mathcal{X}^V_{t,d,k,b}$ and $\mathcal{X}^C_{t,b}$ to represent the number of drones used to transport parcels for a customer with a distance class d, with time window k, battery level b, and the number of drones sent to charge the battery at a current battery level b, respectively. The decision x_t consists of all decision variables at stage t, as seen in (2), subject to constraints (3)-(6), which define the feasible space of $\mathcal{X}_t$.

$$\mathcal{X}_t = \Big[(\mathcal{X}^V_{t,d,k,b}, \mathcal{X}^C_{t,b})\Big]_{\forall d \in \mathcal{D}, k \in \mathcal{K}, b \in \mathcal{B}} \tag{2}$$

$s.t.$

$$\sum_{k \in \mathcal{K}} \sum_{d \in \mathcal{D}} \mathcal{X}^V_{t,d,k,b} + \mathcal{X}^C_{t,b} \leq V_{t,0,b} \qquad \forall b \in \mathcal{B} \setminus \{0\} \tag{3}$$

$$\sum_{\substack{b \in \mathcal{B} \setminus \{0\}, \\ b \geq d}} \mathcal{X}^V_{t,d,k,b} \leq J_{t,d,0,k} \qquad \forall d \in \mathcal{D}, k \in \mathcal{K}, d \leq k \tag{4}$$

$$\sum_{b \in \mathcal{B} \setminus \{B\}} \mathcal{X}^C_{t,b} \leq Q \tag{5}$$

$$\mathcal{X}^V_{t,d,k,b},\ \mathcal{X}^C_{t,b} \in \mathbb{Z} \cup \{0\} \qquad \forall d \in \mathcal{D}, k \in \mathcal{K}, b \in \mathcal{B} \tag{6}$$

Constraints (3) establish that the maximum number of drones used can not be larger than the drones available. Constraints (4) state that the drones used to transport parcels can not exceed the number of parcels at the drone station and their battery levels should meet the energy requirements, i.e., $b \geq d$. Constraint (5) restrict the maximum number of drones that can be sent to charge the battery at any time period t. Constraints (6) define the nature of the variables.

Stochastic Processes: The transition from S_{t-1} to S_t is influenced by the decision $x_t \in \mathcal{X}_t$ and the exogenous information. Note that the parcel arrival rate and its characteristics, i.e., the time window length, release time, and customer destination, are determined by a probability distribution. To model this stochastic process, we introduce $\hat{J}_{t,d,r,k}$ to represent the number of parcels that arrive at the station with destination d, release time r, and time window k. The exogenous information W_t at stage t consists of all the *new information* represented by $\hat{J}_{t,d,r,k}$, as seen in (7).

$$W_t = \left[\left(\hat{J}_{t,d,r,k}\right)\right]_{\forall d \in \mathcal{D}, r \in \mathcal{R}, k \in \mathcal{K}, b \in \mathcal{B}} \tag{7}$$

A state S_t at stage t occurs as the result of the state of the previous stage S_{t-1}, the decision of the previous stage x_{t-1} plus the exogenous information captured in W_t that became known between the stages. The transition of the tasks is set by the time window k of the parcel, relative to the release time r, the number of parcels transported in the previous stage $t-1$, and the random arrival of new parcels. All of these factors, and index relations, are used to capture the transition of the system. We represent them using the transition function S^M, as shown in (8).

$$S_t = S^M\left(S_{t-1}, x_{t-1}, W_t\left(\hat{J}_t\right)\right) \qquad \forall t \in \mathcal{T} |\ t > 0 \tag{8}$$

Transition of Transportation Tasks' States:

$$J_{t,d,0,k} = J_{t-1,d,0,k+1} - \sum_{b \in \mathcal{B} \setminus \{0\}} \mathcal{X}^V_{t-1,d,k+1,b} + J_{t-1,d,1,k} + \hat{J}_{t,d,0,k} \qquad \forall d \in \mathcal{D}, k \in \mathcal{K} \setminus \{K\} \tag{9}$$

$$J_{t,d,0,K} = J_{t-1,d,1,K} + \hat{J}_{t,d,0,K} \qquad \forall d \in \mathcal{D} \tag{10}$$

$$J_{t,d,r,k} = J_{t-1,d,r+1,k} + \hat{J}_{t,d,r,k} \qquad \forall d \in \mathcal{D}, r \in \mathcal{R} \setminus \{0, R\}, k \in \mathcal{K} \tag{11}$$

$$J_{t,d,R,k} = \hat{J}_{t,d,R,k} \qquad \forall d \in \mathcal{D}, k \in \mathcal{K} \tag{12}$$

Constraints (9) define the number of parcels that become available to be transported at a given period t with a time window k. Constraints (10) define the number of transportation tasks that become available with a maximum time

window. Constraints (11) define the transportation tasks that become available with a waiting time r at a stage t. Constraints (12) ensure that the parcels with the maximum waiting time R at a stage t are given by the exogenous information arriving to the system.

Transition of Drones' States:

$$V_{t,0,b} = V_{t-1,0,b} - \sum_{k\in\mathcal{K}}\sum_{d\in\mathcal{D}} \mathcal{X}^V_{t-1,d,k,b} - \mathcal{X}^C_{t-1,b} + V_{t-1,1,b+1} + \mathcal{X}^C_{t-1,b-1} \quad \forall b \in \mathcal{B}\setminus\{0,B\} \tag{13}$$

$$V_{t,0,B} = V_{t-1,0,B} - \sum_{k\in\mathcal{K}}\sum_{d\in\mathcal{D}} \mathcal{X}^V_{t-1,d,k,B} + \mathcal{X}^C_{t-1,B-1} \tag{14}$$

$$V_{t,r,b} = V_{t-1,r+1,b+1} + \sum_{k\in\mathcal{K}} \mathcal{X}^V_{t-1,r,k,b} \quad \forall r \in \mathcal{R}\setminus\{0,R\}, b \in \mathcal{B}\setminus\{0,B\} \tag{15}$$

$$V_{t,R,b} = \sum_{k\in\mathcal{K}} \mathcal{X}^V_{t,R,k,b} \quad \forall b \in \mathcal{B}\setminus\{0\} \tag{16}$$

Constraints (13) establish the number of drones available to transport a parcel at a given battery level. Constraints (14) define the number of drones that become available at a maximum battery level. Constraints (15) define the transition of the release time for drones that are transporting a parcel. Constraints (16) set the number of drones that have a maximum release time R at each stage t.

Bellman's Equation: The objective function $C(S_t, x_t)$ at time period t depends on the number of drones used to transport parcels and the use of the alternative transportation mode. We define a variable $z_{t,d}$ as the number of parcels transported to customers with a distance class d by the alternative transportation mode at stage t. This variable is constrained by the transportation tasks' state $J_{t,d,r,k}$ and the number of drones used to transport parcels $\mathcal{X}^V_{t,d,k,b}$, as seen in (18). This way, the costs of the decisions can be defined as a function of x_t and S_t, as shown in (17).

$$C(S_t, x_t) = \sum_{d\in\mathcal{D}} C^L \cdot z_{t,d} \tag{17}$$

where,

$$z_{t,d} = J_{t,d,0,0} - \sum_{b\in\mathcal{B}\setminus\{0\}} \mathcal{X}^V_{t,d,0,b} \quad \forall d \in \mathcal{D} \tag{18}$$

Bellman's equation enables the sequential minimization of the expected costs over the horizon, i.e., the sum of (17) over all $t \in \mathcal{T}$, since there is uncertainty in the arrival of parcels, and thus the states. We aim to find the policy that minimizes the logistics overhead costs over our planning horizon. Therefore, we

define a policy $\pi \in \Pi$ is a function $\pi : S_t \rightarrow x_t$ that maps each state to a corresponding decision. The optimal policy π^* may be found by solving the Bellman optimality equations for each state, as shown in (19). In particular, the transition function S^M allows us to write the expectation in terms of the current state, the corresponding decision, and the given realization. The set of all realizations is denoted by Ω, i.e., $W_t \in \Omega$, $\forall t \in T$, where for each realization $\omega \in \Omega$ there is a probability p_ω^Ω.

$$V_t^{\pi^*}(S_t) = \min_{x_t \in X(S_t)} \left(C(S_t, x_t) + \sum_{\omega \in \Omega} \left(p_\omega^\Omega \cdot V_{t+1}^{\pi^*}\left(S^M(S_t, \mathcal{X}_t, \omega)\right)\right)\right), \quad \forall S_t \in \mathcal{S} \tag{19}$$

The probability p_ω^Ω depends on the realization $\omega \in \Omega$ in three ways. First, it depends on the total number of the arriving parcels, see Equation (22). Second, it depends on the probability that the $\hat{J}_{t,d,r,k}$ parcels will have customer destination class d, release-time r and time-window length k. Third, it depends on a multinomial coefficient β [15] that counts the ways of assigning the total number of arriving parcels j to variable $\hat{J}_{t,d,r,k}$. This coefficient is necessary since the order in which parcels arrive does not matter and their characteristics are allowed to *repeat*. With these three aspects, the probability p_ω^Ω can be computed using (20).

$$p_\omega^\Omega = \beta \cdot p_j^J \cdot \prod_{r \in \mathcal{R}, k \in \mathcal{K}, d \in \mathcal{D}, b \in \mathcal{B}} \left((p_d^{D^J} p_r^{R^J} p_k^{K^J})^{\hat{J}_{d,r,k}^\omega} \right) \tag{20}$$

where,

$$\omega = \left[(\hat{J}_{t,d,r,k}^w) \right]_{\forall, r \in \mathcal{R}, k \in \mathcal{K}, d \in \mathcal{D}, b \in \mathcal{B}} \tag{21}$$

$$j = \sum_{r \in \mathcal{R}, k \in \mathcal{K}, d \in \mathcal{D}} \hat{J}_{d,r,k}^w \tag{22}$$

$$\beta = \frac{j!}{\prod_{r \in \mathcal{R}, k \in \mathcal{K}, d \in \mathcal{D}} \left(\hat{J}_{d,r,k}^w ! \right)} \tag{23}$$

The optimal policy for D-DSDP can be found by solving (19) using dynamic programming. Due to the curses of dimensionality, this is not possible for realistically sized instances. Hence, we develop a reinforcement learning approach to solve this MDP formulation in Sect. 4.

4 Reinforcement Learning Approach

In this section, a detailed description of the RL approach to solve the D-DSDP is presented. This approach is suitable to derive high quality decision policies for

MDP problems that cannot be solved exactly. Such a decision policy is a mapping of states to actions, where the actions account for their long-term effects. More specifically, we use the approximate value iteration algorithm [21]. Algorithm 1 shows the iterative process of the value-based procedure to approximately explore the solution space and converge to a cost approximation of each state of the system. This algorithmic approach works as a combination of Monte Carlo simulation and the Value Iteration algorithm by fitting a linear regression over a set of features F to determine the decision policy π.

Algorithm 1: Approximate Value Iteration Algorithm

Data: $(N, F, \epsilon, \gamma, \overline{V}, \hat{v}, \theta^f)$
1 $\overline{V}, \hat{v}, \theta^f \leftarrow$ *Initialize()*, $\forall f \in F$
2 $n = 1$
3 **while** $n < N$ **do**
4 **for** $t < \mathcal{T}$ **do**
5 **if** $t > 0$ **then**
6 $\hat{v}_t = \min_{x \in \mathcal{X}_t} \left\{ C(S_t, x_t) + \gamma \, \overline{V}_t \left(S^{M,x}(S_t, x_t) \right) \right\}$
7 $\theta_t^f = \theta_{t-1}^f - H_t \phi_{t-1}^f \left(\overline{V}_{t-1}(S_{t-1}^x) - \hat{v}_t \right), \ \forall f \in F$
8 $\tilde{x}_t \leftarrow \epsilon$-*greedy*$(\mathcal{X}_t)$
9 $S_t^{\tilde{x}} = S^{M,\tilde{x}}(S_t, \tilde{x}_t)$
10 $\phi_t^f \leftarrow$ *Compute*$(S_t^{\tilde{x}}), \ \forall f \in F$
11 $W_{t+1} \leftarrow$ *Random*(Ω)
12 $S_{t+1} = S^M(S_t^{\tilde{x}}, W_{t+1}^n)$
13 $t = t + 1$
14 $n = n + 1$
15 return $\theta^f \ \forall f \in F$

The input data is the number of iterations N, the feature set F, the value ϵ, the learning rate γ, and the initial values for $\overline{V}$, $\hat{v}$, and θ^f. The algorithm starts by setting initial values for $\overline{V}, \hat{v}, \theta^f$, and n (lines 1–2). The iterative process begins in line 3, where a whole horizon $\mathcal{T}$ is run at each iteration n (line 4). We elaborate on the concept of *post-decision* state S_t^x, which refers to the state of the system just after a decision x_t has been made and before the *next-stage* exogenous information W_{t+1} affects the system. The algorithm transitions from state S_t into the *post-decision* state S_t^x by the transition function $S^{M,x}(S_t, x_t)$. If the current stage is larger than 0, the downstream cost $\hat{v}_t$ is computed based on the current policy, that is, make decision x_t (lines 5 - 6). The downstream cost $\hat{v}_t$ represents the direct reward $C(S_t, x_t)$ given x_t, plus the approximated downstream costs of the *post-decision* state (or estimated downstream value) $\overline{V}_t$, i.e., one-step look-ahead with a bootstrap estimate [21]. Then, the linear-regression feature weights $\theta^f \ \forall f \in F$ are updated by computing the error between the previous stage estimated downstream value $\overline{V}_{t-1}$ and the current stage downstream cost $\hat{v}_t$ (line

7). The optimization matrix H_t is implemented to update the feature weights θ^f. For a comprehensive explanation on the optimization matrix we refer to [16]. To properly balance the exploration and exploitation of the decision space, in line 8 the ϵ-greedy decision policy is applied to choose $\tilde{x}_t \in \mathcal{X}_t$ [14]. Then, system transitions into the *post-decision* state $S_t^{\tilde{x}}$ by applying the transition function $S^{M,\tilde{x}}$ (line 9). After this, the features $\phi_t^f \ \forall f \in F$ are computed and stored based on the information provided by $S_t^{\tilde{x}}$ (line 10). Once the exogenous information for the next stage W_{t+1} has arrived (line 11), the algorithm transitions into the next state S_{t+1} by applying S^M (line 12). This process, i.e., from lines 5 - 13, is repeated at every stage $t \in \mathcal{T}$. Then, once n reaches value N, the algorithm returns the learned weights $\theta^f \ \forall f \in F$ to determine the policy π through the value function approximation $\overline{V}_t(S_t^x)$.

5 Numerical Experiments

5.1 Experimental Design

The experiments were executed on a computer equipped with a 1.90 GHz Intel(R) Core(TM) i7-8665U, 16 GB of RAM, and running Windows 10 in 64-bit mode. The instance sets used for the feature selection and the experiments are described in Table 2. In order to select a proper set of features for our RL algorithm, we analyzed the predictive power of a wide set of features. We first ran a long simulation and stored the features' values for all encountered states. Next, for each of the encountered states, we sum the observed costs over the states encountered in the subsequent ten stages, and chose the feature set resulting in the lowest error and the highest predictive power explaining the subsequent 10 stages costs. The final set of selected features for the RL algorithm are the number of drones available per battery level, the number of parcels per release time, the urgent parcels, i.e., that cannot wait one more stage, the non-urgent parcels, the non-urgent parcels per distance class, the total number of parcels in the system, the number of newly arrived parcels, the average distance class over all parcels, and the total travel time to transport all parcels to their corresponding customers. The training times of the RL algorithm were 282 and 644 s for the small and large instance sets, respectively.

5.2 Comparison of the RL Approach with Heuristic Strategies

These experiments study the effectiveness of our RL approach in comparison to four different operational strategies. The procedure of the algorithms is described as follows. The first algorithm chooses an *aleatory decision* at every stage. The second is a *transport-first heuristic* that always prioritizes the transport of parcels and sends drones to charge the battery when they either run out of battery or need to wait. The third is a *charge-first heuristic* that always prioritizes charging the battery of the drones and sends them to transport parcels only when the battery stations are occupied or the batteries of the drones are fully charged.

Table 2. Instance settings.

Set	Meaning	Small instances	Large instances
$\mathcal{T}$:	set of time periods	96	96
$\mathcal{D}$:	set of customer classes	3	3
$\mathcal{K}$:	set of time periods until expiration of the time window	6	6
$\mathcal{R}$:	set of time period until arrival of the parcel	4	4
$\mathcal{B}$:	set of battery levels	10	10
V_t :	number of drones available in the system at time $t \in \mathcal{T}$	10	20
Q :	number of charging stations at the drone station	10	15
$\lambda_{j,t}$:	arrival rate of parcel $j \in J$ at time $t \in \mathcal{T}$	10	20
C^L :	Cost of a delayed parcel	1	1

The last algorithm is a *versatile heuristic* that charges a predefined percentage of drones based on the average battery level of the fleet. Then, the remaining drones are sent to deliver parcels if possible. The results of the different heuristics and the RL algorithm for small and large sets of instances are provided in Table 3, where we consider probabilities $P = \{1/3, 1/3, 1/3\}$ for parcels to be delivered at a customer location $\mathcal{D} = \{1, 2, 3\}$. Furthermore, costs, standard deviations, and horizon running times (time required to run a simulation of 96 stages) are representative of 2000 replications and rounded to the next integer for the final cost, to the second decimal for the deviation, and to the third decimal for the horizon time.

Table 3. Performance comparison of heuristic strategies and the RL algorithm.

Instance size	Small instances			Large instances		
Algorithm	Final cost	Standard deviation	Horizon time (s)	Final cost	Standard deviation	Horizon time (s)
Aleatory heuristic	536	+/−6.80	0.009	1118	+/−14.36	0.015
Transport-first heuristic	516	+/−6.66	0.009	1115	+/−14.28	0.016
Charge-first heuristic	518	+/−6.68	0.007	1118	+/−14.14	0.014
Versatile heuristic	490	+/−6.24	0.007	1097	+/−14.17	0.014
RL algorithm	**447**	**+/−5.59**	**0.111**	**1060**	**+/−13.81**	**0.136**

Results show that the RL algorithm outperforms all the heuristics providing an objective value of 447 and 1060 for the small and large instances, respectively. As expected, the aleatory heuristic is the algorithm that has the worst performance for both sets of instances, which is the same as the charge-first heuristic for the large set of instances. For the whole set of heuristics, we can see standard deviations larger or equal to 6.24 and 14.14, corresponding to the small and large instance sets, whereas the RL algorithm results in smaller standard deviations of 5.59 and 13.81. This shows that the dispersion of the RL algorithm's performance with respect to its objective value is more stable and solid in comparison to the heuristic approaches, but it has larger computational times. In contrast to the heuristics, the RL algorithm needs to evaluate all possible decisions, requiring computing the feature values of all corresponding post-decisions

states. However, computation times are still low enough to support online decision making, especially when taking into account that the reported times below 1 s include 96 decision moments and time required for running the simulation. Consequently, we conclude that the RL algorithm is able to explore the solution space and learn from the tested scenarios to make decisions that outperform different operational strategies within reasonable computational times.

5.3 Experiments on Different Instance Scenarios for the RL Approach

The last set of experiments studies how the performance of the best RL algorithm of Sect. 5.2 changes when studying different configuration scenarios. To carry out the experiments, we use the small instance set described in Table 2, which we refer to as *basic instance set*, and vary the time-window length, the number of charging stations, the number of battery levels, and the probabilities $P = \{1/9, 3/9, 5/9\}$ that a given parcel has to be delivered at a customer location $\mathcal{D} = \{1, 2, 3\}$. Regarding the latter, opposed to the original equal probability distribution representing a situation with higher customer density in the city center, the adjusted distribution assumes an equal density, hence more customers at the outskirts of the city. Results are provided in Table 4, where costs, standard deviations and running times are representative of 2000 replications and rounded to the next integer, for the final cost, and to the second decimal, for the deviation.

Table 4. Different instance settings for the RL algorithm.

Instance type	Final cost	Standard deviation	Experiment purpose
Basic instance set	447	+/−5.59	-
Variation 1, $\mathcal{K} = 4$	486	+/−6.17	Smaller time windows
Variation 2, $Q = 5$	472	+/−6.20	Less charging stations
Variation 3, $\mathcal{B} = 5$	464	+/−6.06	Smaller batteries
Variation 4, P	566	+/−7.21	More distant customers

From Table 4, we see that the all the variations increase the final costs and the standard deviations in comparison with the basic instance set. The first variation shows that as the time windows become smaller, parcels can spend less time at the drone station. As such, the instance set becomes more challenging since parcels might not be transported in time. Regarding the second variation, a reduction in charging stations will cause a situation where drones are not able to charge their battery on time for parcels requiring higher energy levels. Results on the third variation show that smaller batteries result in higher final costs. The last variation increases the costs when most of the customers are located on the outskirts. This is more challenging since the RL algorithm should find a policy that allows it to meet time window constraints for distant destinations and, at the same time, to reserve more drones with higher energy levels.

6 Conclusions

We introduced the Dynamic Drone Scheduling Delivery Problem (D-DSDP), which incorporates time windows, release times, charging stations, energy consumption features, and stochastic parcel arrival into the decision-making.

We address the D-DSDP by providing a Markov Decision Processes model and developing an Approximate Value-Iteration Reinforcement Learning algorithm. We carry out two different sets of experiments. In the first set of experiments, we compare the RL algorithm with four dispatching heuristics that resemble decision-making of human planners. Results show that our algorithm is able to outperform the heuristic strategies at the cost of increasing computational times. In the second set of experiments, we test how different instance settings affect the performance of the RL algorithm. We find higher operational costs when the maximum time-window length, the number of charging stations, and the energy levels of the drone batteries are reduced and when most of the customers are located in the outskirts of the city.

With regards to future research, we aim at studying other extensions such as picking up parcels from the customer locations to transport them to the drone station, as single or combined trips. In addition, the D-DSDP can also be extended to an infinite horizon setting.

Acknowledgements. This work was funded by the Chilean National Agency for Research and Development (ANID)/Scholarship Program/DOCTORADO BECAS CHILE/2019 - 72200288.

References

1. Benarbia, T., Kyamakya, K.: A literature review of drone-based package delivery logistics systems and their implementation feasibility. Sustainability **14**(1), 360 (2021)
2. Coelho, B.N., et al.: A multi-objective green UAV routing problem. Comput. Oper. Res. **88**, 306–315 (2017)
3. 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)
4. Jennings, D., Figliozzi, M.: Study of road autonomous delivery robots and their potential effects on freight efficiency and travel. Transp. Res. Rec. **2674**(9), 1019–1029 (2020)
5. Khoufi, I., Laouiti, A., Adjih, C.: A survey of recent extended variants of the traveling salesman and vehicle routing problems for unmanned aerial vehicles. Drones **3**(3), 66 (2019)
6. Liang, Y.J., Luo, Z.X.: A survey of truck-drone routing problem: literature review and research prospects. J. Oper. Res. Soc. China **10**, 343–377 (2022). https://doi.org/10.1007/s40305-021-00383-4
7. Liu, C., Chen, H., Li, X., Liu, Z.: A scheduling decision support model for minimizing the number of drones with dynamic package arrivals and personalized deadlines. Expert Syst. Appl. **167**, 114157 (2021)
8. Liu, Y.: An optimization-driven dynamic vehicle routing algorithm for on-demand meal delivery using drones. Comput. Oper. Res. **111**, 1–20 (2019)

9. Macrina, G., Pugliese, L.D.P., Guerriero, F., Laporte, G.: Drone-aided routing: a literature review. Transp. Res. Part C Emerg. Technol. **120**, 102762 (2020)
10. 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)
11. Murray, C.C., Raj, R.: The multiple flying sidekicks traveling salesman problem: parcel delivery with multiple drones. Transport. Res. Part C Emerg. Technol. **110**, 368–398 (2020)
12. Otto, A., Agatz, N., Campbell, J., Golden, B., Pesch, E.: Optimization approaches for civil applications of unmanned aerial vehicles (UAVS) or aerial drones: a survey. Networks **72**(4), 411–458 (2018)
13. Pasha, J., et al.: The drone scheduling problem: a systematic state-of-the-art review. IEEE Trans. Intell. Transp. Syst. (2022)
14. Powell, W.B., Ryzhov, I.O.: Optimal Learning, vol. 841. Wiley, Hoboken (2012)
15. Riordan, J.: An Introduction to Combinatorial Analysis. Princeton University Press, Princeton (2014)
16. Rivera, A.E.P.: Anticipatory freight scheduling in synchromodal transport (2018)
17. Roberti, R., Ruthmair, M.: Exact methods for the traveling salesman problem with drone. Transp. Sci. **55**(2), 315–335 (2021)
18. Rojas Viloria, D., Solano-Charris, E.L., Muñoz-Villamizar, A., Montoya-Torres, J.R.: Unmanned aerial vehicles/drones in vehicle routing problems: a literature review. Int. Trans. Oper. Res. **28**(4), 1626–1657 (2021)
19. Schermer, D., Moeini, M., Wendt, O.: A b ranch-and-cut approach and alternative formulations for the traveling salesman problem with drone. Networks **76**(2), 164–186 (2020)
20. SteadieSeifi, M., Dellaert, N.P., Nuijten, W., Van Woensel, T., Raoufi, R.: Multimodal freight transportation planning: a literature review. Eur. J. Oper. Res. **233**(1), 1–15 (2014)
21. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. MIT press, Cambridge (2018)
22. Troudi, A., Addouche, S.A., Dellagi, S., Mhamedi, A.E.: Sizing of the drone delivery fleet considering energy autonomy. Sustainability **10**(9), 3344 (2018)
23. Wang, X., Zhan, L., Ruan, J., Zhang, J.: How to choose "last mile" delivery modes for e-fulfillment. Math. Prob. Eng. **2014** (2014)