Design and Assessment of a Fleet Management Strategy for UAV Pickup and Delivery Networks

Nikki Kamphuis

Design and Assessment of a Fleet Management Strategy for UAV Pickup and Delivery Networks

by

Nikki Kamphuis

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on 12 April 2024.

Student number:	4572637
Project duration:	November 2022 – April 2024
Thesis committee:	Dr. B.F. Lopes Dos Santos TU Delft, Chaiman
	Dr. O.A. Sharpans'kykh TU Delft, Supervisor
	Dr. M.J. Ribeiro TU Delft, Supervisor
	Dr. J. Ellerbroek TU Delft, Examiner

TU Delft

Acknowledgements

A few years ago, a joke that my friends and I shared was, ’Afstuderen? Nooit!’ translating to ’Graduating? Never!’. At the end of this journey, this thought might not have always been a joke while writing my thesis, especially with the (personal) hardships I encountered. To me, a large part of the thesis was having my very own project in which I could decide the direction and show initiative. I am grateful for this learning opportunity and for being able to develop and reflect on myself freely.

However, I am even more grateful for my support system during this process. To start, I would like to thank my supervisors dr. Alexei Sharpans’kykh, dr. Marta Ribeiro and ir. Maurits Dogterom for their enthusiasm and contributions to the project and their compassion regarding my personal situation. While I did not always look forward to the update meetings, I always left them with new motivation to work on the project.

I would also like to thank my parents for their continuous support and help in critically making my own choices. This thesis is dedicated to you, Dad. Furthermore, I would like to thank my boyfriend, Max, for being so sweet and supportive and believing in the finished product from the start while jokingly reminding me that resigning is also possible, even on the brink of graduation. Next to him, I would also like to thank my roommates, to whom I could vent about everything with a cup of tea and who have been so understanding. Finally, I would like to thank all my friends, particularly Tessa, for their support, fun outings, and study period.

To close, I am forever grateful for my experiences in Delft, which have shaped me into who I am today. To any student reading this, please enjoy student life to the fullest of its potential (whatever that may be to you), and don’t be afraid to ask stupid questions.

Nikki Kamphuis
Delft, March 2024

Contents

List of Figures

List of Tables

List of Abbreviations

ADP	Approximate Dynamic Programming
BuGP	Bundle Generation Problem
CBAA	Consensur-Based Auction Algorithm
CBBA	Consensus-Based Bundle Algorithm
CTR	Control Tower Region
DDP	Drone Delivery Problem
IA	Instantaneous Assignment
ICAO	International Civil Aviation Organization
ID	In-schedule Dependencies
KPI	Key Performance Indicator
LSCP	Location Set Covering Problem
MAA	Medical Air Assistance
MCLP	Maximal Covering Location Problem
MDP	Markov Decision Process
MDS	Medical Drone Service
MEXCLP	Maximum Expected Covering Location Problem
MILP	Mixed Integer Linear Programming
MIP	Mixed Integer Programming
MLP	Multi-Layer Perceptron
MRTA	Multi-Robot Task Allocation
MT	Multi-Task
ND	No Dependencies
NZa	Dutch Healthcare Authority
PDP	Pickup and Delivery Problem
pTeSSI	Probabilistic Temporal Sequential Single-Item auction
RIVM	National Institute for Public Health and Environment
RL	Reinforcement Learning
SDVRP	Stochastic Dynamic Vehicle Routing Problem
SPDP	Simultaneous Pickup and Delivery Problem
SR	Single-Robot

SSC	Sequential Single-Cluster
SSI	Sequential Single-Item
ST	Single-Task
STN	Simple Temporal Network
STNU	Spatial Temporal Network with Uncertainty
TA	Time-extended Assignment
TAT	Turn Around Time
TeSSI	Temporal Sequential Single-Item
TPR	Third-Party Risk
TSP	Travelling Salesman Problem
UAS	Unmanned Aerial System
UAV	Unmanned Aerial Vehicle
VRP	Vehicle Routing Problem
VTOL	Vertical Take-Off and Landing
XD	Cross-schedule Dependencies

Introduction

In recent years, the evolution of Dutch healthcare practices has underscored the importance of innovative solutions to foster sustainable healthcare delivery. Recognizing this imperative, a collaborative effort between PostNL Health and ANWB Medical Air Assistance in the Netherlands seeks to address this gap by leveraging a fleet of drones to transport medical materials.

While the companies are actively engaged in safety testing for drone technology, they are concurrently developing a comprehensive business case for their proposed service, Medical Drone Service. This endeavor involves careful considerations, from identifying potential customer segments to assessing daily operational costs to inform pricing strategies.

Seeking expert input, the companies contacted Delft University of Technology to devise a tailored strategy aligned with their unique network requirements. Consequently, we developed a fleet management strategy characterized by its adaptability to unpredictable demand. The company provided valuable insights throughout the process through bi-weekly update meetings and active participation in milestone events such as kick-off, midterm, and green light meetings. Enabling this company to embark on its journey holds promise for fostering sustainable innovations in healthcare.

This thesis report is structured as follows: Part I presents the scientific paper detailing our research findings. Part II encompasses a comprehensive literature review that provides a theoretical foundation for our study. Finally, Part III offers additional insights and results from our research efforts.

I

Scientific Paper

Design and Assessment of a Fleet Management Strategy for UAV Pickup and Delivery Networks

Nikki Kamphuis*

Delft University of Technology, Delft, The Netherlands

Abstract

The Dutch healthcare sector wrestles with rising costs, staff shortages, and increased demand due to healthcare centralization. This, coupled with worsening traffic congestion, underscores the need for efficient solutions like drone-based medical transport. This paper addresses the need for effective fleet management strategies tailored to the unique demands of the healthcare environment. Specifically, it seeks to develop an adaptive strategy that accommodates the network's expansion and the inherently stochastic and urgent nature of pickup and delivery orders associated with medical transport. Utilizing an agent-based model formulation, the paper introduces a novel approach combining adapted temporal sequential single-item auctions for allocation and scheduling with reinforcement learning for drone repositioning to optimize fleet management. Our findings highlight the strategy's consistent efficiency across various demand scenarios, maintaining performance within predefined limits. Notably, the repositioning module significantly enhances the fleet utility and the fraction of served orders, albeit at the expense of increased cost per delivery. Conversely, the reallocation module causes minimal performance improvement. Under heightened stochasticity introduced by urgent orders, the strategy maintains stable costs per delivery while fleet utility and order fulfillment rates decline. Additionally, our investigation underscores the increasing benefits of repositioning in more stochastic scenarios. Moreover, exploring hybrid fleets reveals that while short-range high-payload drones can reduce cost per delivery, they compromise overall fleet utility and order fulfillment rates. Furthermore, we identify the under-utilization of payload capacity in scenarios with orders weighing up to 2 kilograms for a drone with a payload of 10 kilograms.

1 Introduction

Confronted with escalating service demands and a contracting workforce, the Dutch healthcare sector is urgently seeking innovative solutions and enhanced support to maintain its services. The Dutch Healthcare Authority (NZa) has recognized the resource strains and the imperative for strategic and operational reforms, identifying the centralization of services as an essential tactic(NZa-Magazines 04, 2022). However, the move towards centralization increases the necessity for medical transport, underscored by the 3.5 million medical deliveries made annually (Dogterom, 2023). This need emerges amid escalating traffic congestion, which saw a significant surge of 17% in 2023 alone, further burdening the already overloaded Dutch road network (ANWB Verkeersinformatie, 2023).

Simultaneously, the increasing need for rapid and efficient delivery solutions has stimulated interest in drone-assisted delivery, which avoids common issues like traffic congestion and enables access to remote locations, reducing the environmental impact of deliveries. Moreover, the carbon-neutral nature of drones further reduces their environmental impact (Demir et al., 2022).

Operational cases using drone technology have already been established, setting a precedent for further implementation of this application. For instance, the company Zipline stands out as a front-runner in this field, particularly renowned for delivering medical supplies to remote areas like Rwanda. This pioneering company has successfully showcased the potential of drones to revolutionize healthcare logistics by delivering medical supplies to remote and underserved communities, thereby demonstrating the transformative impact and feasibility of drone technology in critical services (Zipline, 2023).

Building on the promising capabilities of drone-assisted deliveries, Medical Drone Service (MDS), a drone-based delivery service, presents a vital solution for the Dutch healthcare industry, addressing the pressing need for swift and dependable medical transport. By guaranteeing cost-effective and reliable drone deliveries of medical supplies, the ambitions of MDS align with sustainable healthcare practices. The challenge, particularly for companies like MDS, revolves around learning efficient fleet management to maintain operational efficiency while preserving the commitment to high service quality standards. This balance is crucial in an environment characterized by fluctuating and un-

*Msc Student, Sustainable Air Transport, Faculty of Aerospace Engineering, Delft University of Technology predictable demand, where reliability is paramount due to the critical nature of medical transport.

Existing drone pickup and delivery strategies are often developed with simpler, static environments in mind, lacking the flexibility to adapt to dynamic, real-time variations. This stands in contrast to adaptive ground-vehicle-based research, which mainly serves ride-hailing services. However, these are limited to processing single requests sequentially, a notable shortcoming. Additionally, insights from traditional vehicle pickup and delivery research underscore the limitations of fully centralized approaches, highlighting their potential to constrain scalability. This highlights a crucial research gap: the need for scalable, adaptive algorithms capable of handling real-time changes, like fluctuating demand and environmental conditions, while efficiently managing multiple orders simultaneously.

This framework is proposed within an agent-based model formulation, which is advantageous due to its inherent modularity. This feature enables the model to mirror the complexities of the network accurately and offers opportunities for decentralization. The choice of auction mechanisms for allocation and scheduling is grounded in their scalability and capacity to yield near-optimal results efficiently. Moreover, the incorporation of reinforcement learning techniques for drone repositioning is motivated by their adaptability to unforeseen scenarios and proficiency in making informed decisions based on anticipated future conditions. Together, these novel approaches promise to dynamically adjust fleet management to accommodate changing operational demands and environmental conditions. This strategic combination aims to significantly reduce operational costs and boost efficiency within the critical domain of medical logistics, demonstrating a tailored response to the unique challenges presented in this sector.

The structure of this paper is organized as follows: Initially, section 2 delves into a detailed examination of the operational conditions and the rationale behind the selection of our techniques. Following this, section 3 outlines the agent-based model formulation, elaborating on each component. Section 4 then offers an in-depth look at our fleet management strategy. We conduct experiments detailed in section 5 to evaluate the strategy's effectiveness, with the outcomes presented in section 6 .section 7 summarizes the results and reflects on the initial hypotheses and puts forth recommendations, and section 8 discusses the conclusion.

2 Background

This section starts with a problem description, stating all relevant details about the operational process of the drone delivery service for medical transport. Then, a literature study on similar problems is presented, indicating the gap in current literature. Finally, we provide the reasoning behind the selected approach.

2.1 Problem Description

The operational process of MDS is the foundation of our model. This process is organized as follows: When an order is received, the MDS operations center assigns it to a suitable drone. Before the drone departs and arrives at the pickup location, safety checks such as weather assessments are made to determine if conditions permit flight. Although these pre-flight and landing checks are crucial in practice, they are omitted in our simulation for simplification.

Following the safety assessment, the order is loaded onto the drone. A battery life check is then conducted to ensure the drone has sufficient power for the flight, which, if necessary, leads to a battery swap. Once the delivery is completed, the drone undergoes another battery check and is sent to an appropriate station to await the next task.

MDS aims to deploy drones with a range of approximately 100 kilometers at a speed of 90 kilometers per hour and a maximum payload of 3 kilograms. This capacity enables the possibility of handling multiple deliveries simultaneously. For scenarios where drone delivery is not feasibledue to weather conditions, operational constraints, or insufficient capacitya backup network of cars is available to guarantee service continuity.

Orders are categorized into three categories based on their urgency: urgent, semi-urgent, and same-day. Urgent orders are to be delivered within one hour of receipt, semi-urgent orders within two to four hours, and same-day orders by the end of the day they are placed. This urgency categorization adds a layer of unpredictability to the operation, necessitating robust management strategies to uphold high service standards. Accordingly, these urgency levels play a significant role in our simulation model, influencing how deliveries are prioritized and managed.

Thus, the task is to devise a fleet management strategy that satisfies the following requirements (denoted as R_n):

- R_1 **- Modular**: Given that MDS is in its pilot phase and thus subject to significant growth, the model must support easy modification, including the expansion, adjustment, and removal of variables.
- R_2 **- Explainable**: As the model informs future decision-making, it must be transparent, allowing easy understanding of its processes and decisions.
- R_3 **- Scalable**: MDS intends to create a nationwide network covering all sorts of medical partners. Therefore, the fleet management strategy must emphasize scalability.
- R_4 **- Efficient**: Given the capabilities of the intended drone, the strategy must be able to integrate orders to facilitate efficient operations.
- R_5 **- Adaptable**: The strategy must be capable of adjusting to the stochastic nature of order ar-

rivals, ensuring robust performance in a dynamic environment.

2.2 Research Context

Literature relevant to the problem can be split into two categories: literature on drone-based pickup and delivery problems and ground-vehicle-based pickup and delivery problems.

Starting with drone-based problems, Macrina et al. (2020) comprehensively discusses the drone delivery problem (DDP). Their discussion shows that complexities faced by MDS, like managing dynamic conditions and meeting strict time requirements in a multi-region environment, still need to be explored. The tendency of current research on the drone delivery problem to focus on small-scale scenarios underscores a gap in addressing more complex operational frameworks.

For instance, Liu (2019) introduces a rolling horizon optimization algorithm for on-demand meal delivery with drones. They recommend a shift away from centralized decision-making models, citing enhanced scalability and an improved capacity to navigate uncertainties. Similarly, Huang et al. (2022) create a methodology for task allocation and scheduling drones, utilizing an iterative heuristic. They underscore the importance of looking into inter-region delivery systems.

Furthermore, Campuzano et al. (2022) examine managing drone fleets for time-sensitive orders. Their research, focused on the balance between dispatching drones and recharging to minimize delays, primarily revolves around operations from a single central hub, indicating a limitation in scope that may only partially translate to broader, more decentralized networks.

Zhen et al. (2023) tackle the complexities of large-scale assignment and routing problems with a mixed-integer programming (MIP) formulation and column generation heuristic. Despite their contributions, they acknowledge a significant oversight in not incorporating dynamic order assignment optimization based on real-time demand fluctuations.

Lastly, the application of Q-learning for dynamic drone dispatching by Chen et al. (2022b) shows promise for same-day deliveries. However, they note limitations in scalability and adaptability to large-scale operational changes, which reflects a common theme across existing research.

The collective insights from these studies highlight a critical gap in the literature on the DDP: a lack of comprehensive strategies that are both scalable and adaptive to the specific and complex needs of networks like MDS.

Turning our attention to the literature on ground-vehicle-based pickup and delivery, research on dynamic ride-sharing is very relevant to the problem posed by MDS.

Beirigo et al. (2022) and Kullman et al. (2021) delve into mobility on demand, respectively introducing an approximate dynamic programming and deep reinforcement learning formulation for dispatching and rebalancing vehicles or incorporating third-party vehicles to sustain service levels or optimizing for cost. They argue that these methods surpass reactive optimization strategies. However, research on ride-sharing is limited in usefulness due to the disability to process more than a single request at a time.

Agatz et al. (2012) discuss consolidation of orders through traditional optimization and note the advantages of decentralized matching algorithms for large-scale networks, pointing towards the need for more agile and adaptable operational frameworks. Arslan et al. (2018) work on a rolling horizon algorithm for crowd-sourced delivery and underscore a gap in using predictive data for order arrivals, indicating a direction for future research.

Hildebrandt et al. (2023)'s proposal to integrate reinforcement learning with traditional vehicle routing problem solvers for stochastic dynamic vehicle routing problems (SDVRP) hints at the potential for a predictive, comprehensive approach to addressing the complexities of SDVRPs.

In conclusion, while research on ride-hailing services is promising, it faces a notable limitation in applicability to MDS by processing single requests sequentially. Additionally, insights from traditional vehicle pickup and delivery research highlight the drawbacks of fully centralized approaches, underscoring their potential to limit scalability. These conclusions combined with the insights drawn from research on the DDP underscores the usefulness of our research objective: to design and assess an adaptive strategy for large-scale pickup and delivery networks.

2.3 Selection of Methods

Various techniques have been evaluated to develop the vehicle management strategy. The following subsections provide a detailed overview of the considerations for each component of this strategy.

2.3.1 Agent-Based Modelling

The decision to use an agent-based modelling techniques was made to meet requirements R_1, R_2, and R_3. Agent-based models are favored for their bottom-up approach, which enhances modularity. New agents with unique properties can be easily integrated into the model. Additionally, these models are inherently decentralized, aligning with previously mentioned strategies for enhancing scalability. This decentralization contributes to the system's flexibility and capacity to better reflect complex scenarios. Lastly, agent-based modeling stands out for its dual capability to conduct local and system-wide analyses, enhancing our ability to understand the model's dynamics in detail. Such an understanding is crucial for accurately interpreting outcomes and making well-informed decisions (Macal and North, 2009). To implement agent-based modelling, Python's Mesa library is used. Mesa enables users to quickly develop agent-based models with pre-built core components like spatial grids, agent schedulers, or custom implementations.

2.3.2 Task Allocation and Scheduling

The selection process for a task allocation and scheduling method suitable for the MDS network focused on the need for a method that satisfies the requirements set in section 2.1.

In Multi-Robot Task Allocation (MRTA) research, strategies have evolved from centralized to decentralized, auction-based methods. Highlighted by Khamis et al. (2015) and Korsah et al. (2013), auction-based methods merge centralized planning's reliability with distributed systems' adaptability, efficiently managing dynamic environments and uncertainties while ensuring robustness by avoiding single failure points. Though they might not always reach the optimal outcomes of optimization strategies in complex scenarios, these auction-based approaches offer scalability and (near-)optimal solutions as proven by Lagoudakis et al. (2005), embodying a favorable compromise for design modularity required in R_1.

Consequently, different auction mechanisms were investigated to identify the most appropriate method to meet our specific requirements. The Sequential Single-Item (SSI) and Sequential Single-Cluster (SSC) auctions stood out for effectively balancing computational demands with utilizing inter-task synergies (Koenig et al., 2010; Heap and Pagnucco, 2013). The Temporal Sequential Single-Item (TeSSI) auction developed by Nunes and Gini (2015), an extension of SSI, was particularly appealing due to its capability to handle temporal constraints. However, the method does not allow *flexible pickup and delivery operations* in which the delivery task does not directly follow the pickup task.

The Adapted TeSSI auction, developed by Chen et al. (2022a), was selected for its ability to intricately schedule and optimize the routes between pickup and delivery points, unlike the original TeSSI auction. This method efficiently arranges all pickup and delivery tasks, improving the MDS network's overall scheduling and routing efficiency, thus satisfying R_4. This method will be enhanced with a feature that permits the decommitment from tasks to cater to the dynamism within the MDS network as required by R_5, allowing for greater flexibility and responsiveness to changes.

2.3.3 Repositioning

Vehicle repositioning is essential in emergency healthcare, taxi services, and bike-sharing sectors to ensure an efficient match between supply and demand. The strategic placement of resources such as ambulances or taxis can significantly enhance service responsiveness and customer satisfaction by minimizing wait times and ensuring availability in high-demand areas. Given the nature of the urgent orders in the MDS network, we deem it necessary to include repositioning in our vehicle management strategy to make our strategy more adaptable (R_5).

Studies, including those by Sayarshad and Chow (2015), highlight the superiority of non-myopic, forward-looking models over simpler, myopic alternatives for their ability to anticipate and prepare for future demand. This foresight is crucial for drone repositioning within the MDS network, ensuring drones are strategically placed in anticipation of peak demand periods. Therefore, our focus is primarily on non-myopic methods that evaluate expected future demand, enabling more informed and proactive decision-making to optimize drone positioning for upcoming needs.

Among the various repositioning strategies examined, real-time relocation models, compliance tables, and reinforcement learning approaches present unique advantages and challenges. Inspired by ambulance relocation strategies, real-time models offer rapid response to changing demands but are computationally intensive, often requiring precomputed solutions to mitigate delays (Bélanger et al., 2019). Compliance tables are less demanding computationally, thanks to precalculation, but their static nature hampers dynamic adaptability, leading to potential inefficiencies in vehicle use (van Barneveld, 2016).

Reinforcement learning introduces a dynamic where an agent learns to make decisions by adapting its actions based on received feedback in the form of rewards or penalties. Within this realm, the Monte Carlo Policy Gradient method stands out. It updates the agent's decision-making policy towards higher rewards based on the outcomes of entire episodes without requiring a model of the environment.

This reinforcement learning approach is particularly promising, balancing adaptability and computational efficiency. It facilitates incremental improvements and system adjustments more smoothly than the rigid frameworks of approximate dynamic programming, real-time optimization, and compliance tables (Schmid, 2012; Nasrollahzadeh et al., 2018). These benefits make it a strong candidate for effectively repositioning drones within the network.

Consequently, the Monte Carlo policy gradient is identified as the most suitable technique for vehicle repositioning drones within the MDS network, offering a good blend of adaptability, efficiency, and strategic foresight essential for managing the dynamic demands of the network.

3 Model Formulation

Agent-based modeling techniques are applied to simulate the operational environment of MDS and assess the capabilities of the devised strategy. This class of modeling methods is particularly suited for the problem at hand due to its modularity. This section provides an in-depth exploration of the model components, encompassing pre-processing procedures, underlying assumptions, and the specifications governing the environment and the agents.

3.1 Pre-processing

The model incorporates pre-computed routes for drones and cars, calculated through the methodology established by van Haasteren (2022).

For drones, van Haasteren employs an A* algorithm capable of minimizing either distance or third-party risk (TPR). These two optimization criteria define routes as either being *fast* or *safe*. The environment is discretized into a grid for the TPR computation, and a risk value is computed for each cell. This risk assessment is based on four failure types: ballistic descent, uncontrolled glide, parachute descent, and flyaway. By considering the probability of each failure occurrence and the population density of each cell, a corresponding risk value is assigned to that cell. Consequently, the A* algorithm utilizes the assigned risk values for navigation.

Van Haasteren employs the BING Maps Distance Matrix API for cars. Through this API, the model calculates the anticipated travel time and distance for each hour throughout the week, considering the variability in traffic conditions. The routes generated by the API are designated as *safe* routes. For *fast* routes, the effect of using lights and sirens is incorporated using a speeding factor.

Although van Haasteren conducted experiments involving a combination of the two route types, this study operates under the assumption that exclusively safe routes are employed. The assumption is built on the expectation that risk-free drone routes will be the standard for air traffic control in the foreseeable future. The presence of cars in the network is solely for backup purposes; thus, using fast routes is not deemed necessary.

3.2 Assumptions

Key assumptions have been established to refine the study's focus and practical applicability. These assumptions lay the groundwork for studying UAV operational dynamics within the network.

3.2.1 Energy Consumption

In the UAV research domain, the intricacies of energy consumption often serve as a focal point as demonstrated by Dorling et al. (2017) and Troudi et al. (2018). However, this research adopts a simplified approach using a maximum range in this research to limit the scope.

This study assumes a battery swap is viable at each client location within the network. Consequently, this assumption leads us to the critical premise that each flight leg must be constrained to a distance shorter than the maximum range achievable by the UAV. It is important to note that while each flight leg is limited in range, the cumulative sum of these combined legs for a given order may surpass the individual UAV's maximum flight range.

3.2.2 Backup Network

MDS has implemented a contingency plan by incorporating a backup network of cars to fulfill orders in instances where no drones are available. These cars are assumed always to be positioned at an origin when needed, and their presence is abundant within the model. Despite the undesirable nature of relying on cars instead of drones, the associated costs are integrated into the fixed costs of the network. Consequently, their influence on key performance indicators (KPIs) is limited.

3.2.3 Observability

Due to the centralized organization of the network, full observability is assumed. This means communication barriers are nonexistent, and the coordinating agent is fully aware of all drone schedules and positions. The centralized structure facilitates seamless coordination and presents an opportunity to maximize overall network performance.

3.2.4 Maintenance

Excluded from the scope of this study are maintenance events for drones, which occur once every 200 flight hours. This timeframe significantly surpasses the planning horizon under consideration, allowing us to concentrate on operational aspects without the influence of routine maintenance occurrences.

3.3 Environment Specification

In our agent-based model, we construct the environment by representing (a portion of) the Netherlands as a bi-directional graph. Within this graph, each node indicates either a customer of MDS or a designated MDS hub for drones (potentially located at a customer location). The connections between these nodes are assigned weights that could signify various factors such as travel time, risk, emissions, or distance between the nodes. These weights are used to determine the details of a single flight, which serve as input to the fleet management strategy. Please note that the weights depend on the vehicle type, car or drone. The environment can be classified as fully observable.

3.4 Agent Specifications

In this section, we discuss the different types of agents within the network. Among these agents, the drone and car agents are dynamic in nature as their primary role involves the delivery of packages. On the other hand, the command center agent and customer agents fall under the category of static agents. These static agents predominantly engage in planning activities rather than dynamic, on-the-move operations. A full overview of agent interactions is found in Figure 1.

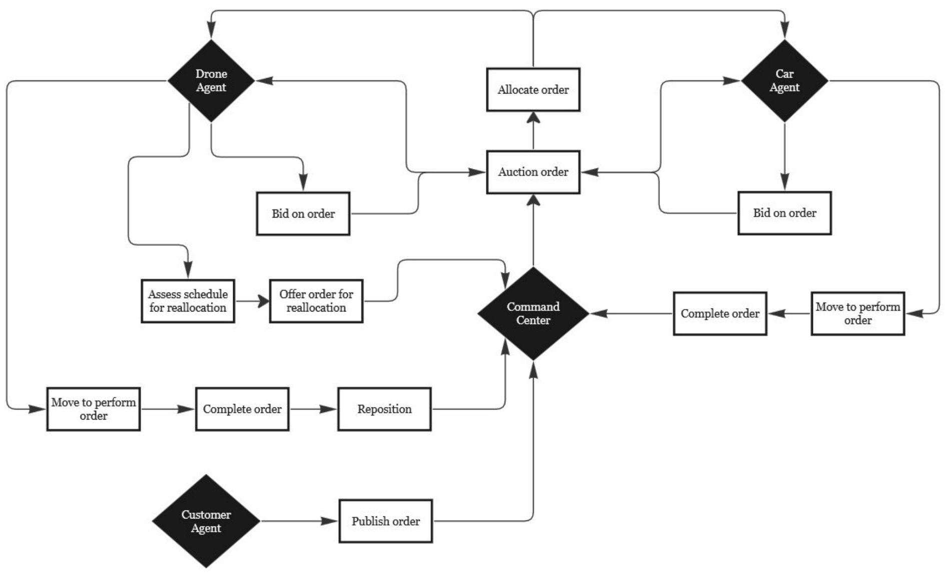


Figure 1: Agent interactions based on agent-based model formulation.

3.4.1 Drone Agent

The drone agent is the primary agent responsible for deliveries within the MDS network. These agents travel to the order's point of origin, retrieve the order, and subsequently transport it to its designated destination. Hence, drone agents are involved in the network's planning and repositioning aspects.

In our model, the specifications for the standard drone within the network are closely aligned with those used by MDS, ensuring relevance and applicability. The key specifications of the UAV are outlined in detail in Table 1.

Table 1: Overview of specifications standard drone MDS.

Specification	**Value**	**Unit**
Speed	90	km/h
Maximum payload	3	kg
Range	100	km

We outline the properties of the drone agent below:

- *Bidding property:* Each drone agent can formulate a bid when competing in auctions for order allocation. The determination of bids is based on a method selected by the end user of the model. The primary method is outlined in section 4.1.2.

- *Moving property:* Drone agents are equipped to navigate over the links of the environment. The speed of these agents is assumed to be constant, resulting in the consistent movement of a fixed distance during each time step. Vital aspects of the moving property include departure and arrival, during which orders are processed in the model. Furthermore, a mandatory turnaround time of 5 minutes is imposed, requiring each drone to wait at least 5 minutes before initiating another departure.

- *Reallocating property:* Drone agents can assess their flight schedules to identify orders suitable for reallocation. Upon determining a suitable order, the agent presents it to the command center for potential reallocation efforts. Orders are only reallocated when the total cost is reduced, ensuring a cost-effective approach.

- *Repositioning property:* Following the completion of a mission and arrival at the destination, a drone agent in correspondence with the command center can relocate to another position only when the drone agent has no further flights planned.

3.4.2 Car Agent

Car agents function as the secondary delivery entities within the network, serving as backups in scenarios where drone agents cannot fulfill an order within the stipulated deadline. Given their role as secondary vehicles, car agents operate with reduced properties but are consistently available, ensuring a car is always ready to execute an unallocated order within the model. We outline the following properties for car agents:

- *Bidding property:* Car agents can formulate bids for order allocation when participating in secondary auctions. Bids are computed using routing information extracted from BING Maps Distance Matrix API. A car agent bids the sum of the traveling costs towards the origin and the destination. If the agent cannot meet the order's deadline, it will not provide a bid.

- *Moving property:* Car agents also navigate the links of the environment. Although their speed may vary during a journey, an average speed is assumed, reflecting the total duration provided by the routing API. Consequently, the movement remains consistent, rendering a fixed distance during each time step. Like drone agents, the processing of orders occurs during the departure and arrival phases.

3.4.3 Command Center Agent

The command center Agent plays a vital role in maintaining a comprehensive overview of the network, overseeing all aspects of order and vehicle agent management through strategic planning actions. The properties inherent to the command center are as follows:

- *Order management property:* The command center manages the coordination of orders. It receives new orders from clients and publishes them. Upon the completion of an order, the center closes the order.

- *Auctioning property:* The command center allocates new orders by auctioning them to vehicle agents. Various auction types can be employed to facilitate this allocation process, offering flexibility and adaptability to different scenarios.

- *Repositioning property:* With a clear understanding of where agents are located in the network and knowledge of anticipated demand, the command center determines a new location for vehicle agents considering repositioning maneuvers.

3.4.4 Customer Agent

Customer agents in the medical drone service network represent various medical entities, including hospitals, out-patient clinics, laboratories, and blood banks. These entities engage with the drone service to facilitate the transportation of medical packages.

- *Ordering property:* One key functionality of customer agents is their ability to create orders for shipping medical material. When creating an order, customers provide information such as the package's mass and the desired delivery deadline. The process of order publication follows a Poisson process.

  The Poisson process models the occurrence of events over time. In this case, the events are the creation of orders for package delivery. The rate parameter λ of the Poisson process is determined by the average number of flights per day for each link in the network. The average number of flights is calculated based on the demand requirements and pre-determined relations. To ensure a realistic simulation, there is a constraint on the maximum average number of flights, set to 5 flights per leg. The average number of flights per leg is used to sample the time between the arrivals of consecutive flights. This time represents the duration between successive orders being created by the customer agent.

4 Fleet Management Strategy

We introduced a multi-agent system with various agent properties in the previous section. Within this system, the drone agents have three key properties: bidding, reallocation, and repositioning, constituting the core elements of our fleet management strategy. This section further elaborates on these components. Each component is thoroughly explained in the subsequent subsections

4.1 Allocation

To allocate resources effectively, we utilize an enhanced version of the Adapted TeSSI allocation mechanism initially introduced by Chen et al. (2022a). Unlike the traditional TeSSI auctions, our Adapted TeSSI mechanism diverges by replacing the simple temporal network with a simultaneous pickup and delivery optimization framework. This modification allows for more flexible pickup and delivery operations, where the delivery task does not directly follow the pickup task. For instance, a drone can first visit two pickup destinations, before going to a delivery destination. Consequently, we gain the ability to intricately schedule orders, leading to more optimal routes.

We extend the formulation of the Adapted TeSSI optimization problem by introducing drone-related constraints and constraints specific to the MDS usecase.

4.1.1 Adapted TeSSI Auction Mechanism

The operational workflow of the auctioneer is detailed in Algorithm 1, where each task is directly assigned to a vehicle agent following their publication.

The auctioneer conducts two sequential auctions to allocate the order. First, the auctioneer tries to assign an order to the set of drone agents, D, in the initial auction. During this auction, drone agents bid based on their increase in itinerary costs, conditioned on their ability to fulfill the order within the specified deadline.

If no drone agent can meet the order's deadline, the auctioneer proceeds to a second auction involving the set of car agents, C. In this auction, car agents bid their lowest cost for executing the order. It is critical to note that drones and cars operate on distinct scheduling mechanisms, a topic further explained in section 4.1.2. As car agents are assumed to be abundantly available, the algorithm concludes after this stage.

Upon identifying the auction winner, the corresponding order is incorporated into the agent's set of orders, denoted as O. Simultaneously, the agent's previous schedule is substituted by the newly computed schedule generated during the bidding process. Note that flights already in progress will remain in the new schedule.

Algorithm 1 Pseudo algorithm for auctioneer

```
1: Input:
   $D$: Set of drone agents
   $C$: Set of car agents
   $O_a$: Set of orders in schedule of agent $a$
   $o$: Order to be allocated
2: Output:
   $s_{d/c}$: New schedule of either agent $d \in D$ or $c \in C$
3: $lowestBid = M$
4: $winner$, $winnerSchedule$ = $None$, $None$
5: for $d \in D$ do
6:    $bid$, $newSchedule$ = $d$.makeBid($d$.currentSchedule, $o$)
7:    if bid < lowestBid then
8:       $lowestBid = bid$
         $winner, winnerSchedule = d, newSchedule$
9:    end if
10: end for
11: if $lowestBid < M$ then
12:    $o$.assignedVehicle = $winner$
       $winner$.replaceSchedule($winnerSchedule$)
       $winner$.orders = $O_{winner} \cup o$
13: else
14:    for $c \in C$ do
15:       $bid, newSchedule$ = $c$.makeBid($c$.currentSchedule, $o$)
16:       if bid < lowestBid then
17:          $lowestBid = bid$
             $winner, winnerSchedule = c, newSchedule$
             $winner$.orders = $O_{winner} \cup o$
18:       end if
19:    end for
20: end if
```

4.1.2 Scheduling Mechanisms and Bids

We utilize two separate scheduling methods to manage car and drone agents within the system. Given the abundant availability of car agents and their minimal impact on the model's key performance indicators, aside from network availability, we opt for a relatively simple scheduling approach outlined by van Haasteren (2022). In this method, cars submit bids representing the cost of traveling to an order's origin and destination points, guaranteeing they can meet the order's deadline. Orders are only combined when they share the same origin and destination. The auction is won by the agent with the lowest bidding cost.

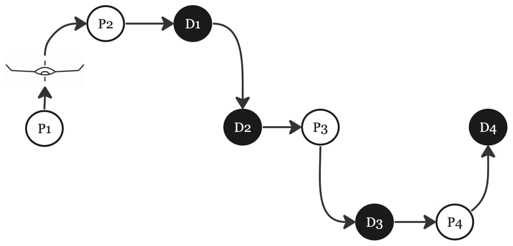


Figure 2: Example of drone agent schedule solution, pickup locations are shown in white, while delivery locations are shown in black.

On the other hand, drone agents follow a more intricate scheduling procedure. When incorporating a new order into their itinerary, drone agents solve a version of the Simultaneous Pickup and Delivery Problem (SPDP). This approach determines the sequence of origins and destinations to be visited. An illustrative solution to the SPDP in the context of the MDS network is presented in Figure 2. The diagram showcases that a destination does not have to succeed its corresponding origin directly, provided that the precedence relationship between the origin and destination is upheld. Furthermore, the combined mass of the orders on board during each flight leg cannot exceed the payload capacity.

When an order is received, the drone agent identifies the moment in its schedule when a complete cycle of deliveries is finished. At this point, there might be some tasks that are pending but not yet initiated. From this moment forward, it reoptimizes the schedule for the pending tasks and the newly published task.

The underlying mathematical model is based on the formulation by Chen et al. (2022a), which is extended to suit the premise of the MDS network. The formulation of the problem is as follows:

Parameters

- n: number of pickup nodes
- $\tilde{n}$: number of delivery nodes coupled to a pickup node

- P: set of pickup nodes, $P = \{1, ..., n\}$
- D: set of delivery nodes coupled to a pickup node, $D = \{n+1, ..., n+\tilde{n}\}$
- $\{0, 2n+1\}$: virtual start and end nodes
- V: set of all nodes, $P \cup D \cup \{0, 2n+1\}$
- A: set of all arcs $\{(i,j) : i,j \in V, i \neq j, i \neq 2n+1, j \neq 0, (i,j) \neq (0, 2n+1)\}$
- $G(V, A)$: Bi-directional graph, containing the nodes on V and arcs on A
- C: payload capacity of a drone in grams
- c_{ij}: cost of traversing arc (i,j)
- d_i: deadline associated with node i
- l_{ij}: distance associated with travelling arc (i,j)
- m_i: mass to be collected at node i
- R: range of the drone agent
- TAT: turnaround time for a drone agent
- TAT_i: turnaround time associated with node i, dependent on previous movement
- t_{ij}: time in minutes associated with traversing arc (i,j)
- t_{max}: maximum time order can spend on board of the drone in minutes

Decision Variables

- X_{ij}: Binary variable equal to 1 if arc (i,j) is used and equal to 0 if not
- Q_i: Integer variable stating the mass of the UAV at node i. Mass is discretized to multiples of 10 grams and $\max(Q_i) = (C/10)$
- B_i: Integer variable stating time of arrival at node i, $0 < B_i \leq 1440$.
- Z_i: Binary variable to prevent subtours, equal to 1 if t_ij and TAT_i are both zero

Objective Function

The objective is to integrate the order into the current schedule with minimal impact on the total schedule cost. Therefore, we re-optimize the sequence of pickup and delivery destinations, including the new order, to minimize the total schedule cost, as depicted in Equation 1. Here, c_{ij} represents the cost of traveling from one node to another. Thus, c_{ij} can be zero if both nodes are at the same location.

$$\min \sum_{(i,j)\in A} X_{ij} c_{ij} + \sum_{i \in \{P+D\}} Z_i \tag{1}$$

Constraints

This section outlines the constraints that govern the limitations and connections within the SDPD optimization problem. Equation 2 establishes that only one arc can enter each node, except for the starting node. Similarly, Equation 3 asserts that only one arc can exit each node, excluding the end node. Moving forward, Equation 4 ensures mass continuity, while Equation 5 prevents exceeding mass capacity. Additionally, the sequence of pickup and delivery nodes is enforced by Equation 6, guaranteeing that every pickup node is visited before its corresponding delivery node. The turnaround time at each node is defined by Equation 7, which specifies a turnaround time of 0 when the preceding node was at the same location or was the starting node. Moreover, Equation 8 maintains time continuity and prevents the formation of subtours. Equation 9 guarantees that each node is visited before its deadline. Lastly, Equation 10 ensures that the drone agent's range is respected, Equation 11 specifies the maximum time on board, and Equation 12 mandates that the drone is empty upon completing the schedule.

$$\sum_{i \in V \setminus \{2n+1\}} X_{ij} = 1 \quad \forall j \in V \setminus \{0\} \tag{2}$$

$$\sum_{j \in V \setminus \{0\}} X_{ij} = 1 \quad \forall i \in V \setminus \{2n+1\} \tag{3}$$

$$Q_j \geq Q_i + m_j X_{ij} \quad \forall (i,j) \in A \tag{4}$$

$$\max(0, m_i) \leq Q_i \leq \min(C, C - m_i) \quad \forall i \in V \tag{5}$$

$$B_i \leq B_{i+n} \quad \forall i \in P \tag{6}$$

$$TAT_i = \sum_{j \in V \setminus \{2n+1\},\, l_{ji} \neq 0} X_{ji} TAT \quad \forall i \in V \setminus \{2n+1\} \tag{7}$$

$$\text{If } j = 0 \text{ and } X_{ji} = 1 \text{ with } l_{ji} \neq 0, \text{ then } TAT_i = 0$$

$$B_j \geq (B_i + t_{ij} + TAT + Z_i) X_{ij} \quad \forall (i,j) \in A \tag{8}$$

$$Z_i \geq 1 - TAT_i \quad \text{if } l_{ij} = 0$$
$$Z_i = 0 \quad \text{otherwise}$$

$$B_i \leq d_i \quad \forall i \in V \tag{9}$$

$$X_{ij} l_{ij} \leq R \quad \forall (i,j) \in A \tag{10}$$

$$B_{i+n} - B_i \leq t_{max} \quad \forall i \in P \tag{11}$$

$$Q_{2n+1} = 0 \tag{12}$$

4.2 Reallocation

We have devised a reallocation mechanism to ensure allocated orders remain optimally assigned as new orders emerge. However, continuously re-auctioning every pending order at each time step incurs high computational costs. Although only a few orders are typically published and auctioned per time step, the backlog of pending orders can quickly accumulate. To address this challenge, we have developed a heuristic method for selecting orders for reallocation.

When a drone agent becomes idle, it performs a single re-evaluation of its current schedule. During this process, an order for re-auctioning is identified based on the proportion of costs contributed to the overall schedule cost. Specifically, the order with the highest cost fraction exceeding a pre-established threshold is selected for reallocation. Additionally, analysis has revealed that urgent orders are unsuitable for reallocation and, therefore, excluded from consideration. Once an order is selected for reallocation, it is auctioned to the other agents using the procedure described in the previous section, and a winning agent is selected. However, the results from the auction have yet to be definitive. Reallocation only occurs if the combined cost of the original schedules of both the auctioning agent and the winning agent exceeds the total cost of the new schedules. After this is confirmed, the order is transferred to the winning agent, and the respective schedules are updated.

The reallocation algorithm is outlined in Algorithm 2. Given the schedule of the reallocating drone agent S_1 and a threshold value, the algorithm outputs the updated schedules S_1 and S_2 for the reallocating drone agent and the winning drone agent, respectively.

The algorithm iterates through the orders in S_1, excluding urgent orders, and computes the cost fraction for each order. It then selects the order with the highest cost fraction for potential reallocation. If the cost fraction of the selected order exceeds the specified threshold, the algorithm proceeds with reallocation.

During reallocation, the algorithm removes the selected order from S_1 and re-auctions it to determine the new schedules S_{1_new} and S_{2_new} for both agents involved. If the total cost of the original schedules exceeds that of the new schedules, the algorithm updates S_1 and S_2 accordingly.

This reallocation mechanism optimizes the allocation of orders while mitigating computational costs, ensuring efficient operation in dynamic environments.

Algorithm 2 Reallocation Mechanism

```
 1: Input:
      $S_1$: Schedule of the reallocating drone agent
      $threshold$: Threshold for initiating reallocation
 2: Output:
      $S_1, S_2$: Updated schedules of the reallocating drone agent and the winning drone agent
 3: $orderForAllocation, highestCostFraction \leftarrow$ None, None
 4: for each $order$ in $S_1$ do
 5:     if $order.urgency \neq$ urgent then
 6:         $costFraction \leftarrow$ computeCostFraction($order$)
 7:         if $costFraction < highestCost$ then
 8:             $orderForAllocation, highestCostFraction \leftarrow order, costFraction$
 9:         end if
10:     end if
11: end for
12: if $highestCostFraction > threshold$ then
13:     $S_{1\_old} \leftarrow S_1$
14:     $S_{1\_new} \leftarrow$ removeOrderFromSchedule($order$)
15:     $S_{2\_new}, S_{2\_old}, newAgent \leftarrow$ reAuction($order$)
16:     if cost($S_{1\_old} + S_{2\_old}$) > cost($S_{1\_new} + S_{2\_new}$) then
17:         $S_1 \leftarrow S_{1\_new}$
18:         $S_2 \leftarrow S_{2\_new}$
19:     end if
20: end if
```

4.3 Repositioning

To effectively maneuver drone agents in dynamic environments and anticipate long-term consequences, we employ reinforcement learning (RL). RL is well-suited for this task due to its ability to adapt to changing conditions, make forward-looking decisions, and handle unforeseen scenarios, making it an ideal solution for the expanding MDS network.

Specifically, we utilize the REINFORCE algorithm developed by Williams (1992), a policy gradient learning approach. This algorithm employs a parameterized policy updated through stochastic gradient ascent (Sutton and Barto, 2018). These updates rely on numerous samples of the gradient of the performance measure to match the actual expectation of the performance measure, essentially making this a Monte Carlo method. However, REINFORCE may suffer from high variance, leading to slow learning. To mitigate this, a baseline is introduced. Although other policy gradient methods, like the actor-critic method, have lower variance, they introduce bias. Ultimately, REINFORCE was selected for its straightforward implementation, aligning with the simplicity requirements outlined in section 2.1.

4.3.1 Hub Determination

Drone agents within the network are repositioned to hub locations. To identify these hubs and their respective areas of coverage, we address a variant of *the p-median problem*, as elaborated by Daskin and Maass (2015). The p-median problem focuses on placing p facilities, or in our case hubs, to minimize the overall demand-weighted average distance between demand nodes and their closest selected facility. In our adaptation of the problem, we opt not to weigh distances by demand due to the dynamic nature of demand, which is instead incorporated into the problem through alternative means. The specifics of our approach to the p-median problem are outlined below.

Parameters

- I: set of locations requiring service
- J: potential facility locations, where $J \subseteq I$
- d_{ij}: distance between two locations
- p: intended number of facilities

Decision Variables

- x_{ij}: binary variable, states whether location i, is coupled to hub j

Objective Function

The objective is to minimize the sum of distances between the locations and their coupled hub location.

$$\min \sum_{i \in I} \sum_{j \in J} d_{ij} x_{ij} \tag{13}$$

Constraints

First, Equation 14 specifies that each location is served by a single hub. Following this, Equation 15 ensures the total count of hubs matches p. Lastly, Equation 16 mandates that each hub also serves its own location.

$$\sum_{j \in J} x_{ij} = 1 \quad \forall i \in I \tag{14}$$

$$\sum_{i \in I} x_{ii} = p \tag{15}$$

$$\sum_{j \in j} \sum_{i \in I} x_{jj} \leq x_{ij} \tag{16}$$

4.3.2 Reinforcement Learning Formulation

We present a detailed description of our repositioning problem through an RL framework. This framework encompasses the definition of the state space, action space, rewards, and the conditions for reaching the terminal state of an episode. In this setup, the drone agent requests a repositioning location from the command center, which employs a policy to choose the appropriate location. Therefore, the command center agent exclusively executes the RL policy, which oversees the entire network. Upon arriving at a terminal state within the simulation, rewards are distributed for all actions executed throughout the episode. This reward system provides feedback, enabling the agent to refine and improve its policy for future decisions.

State Space

The state space structure is outlined in Equation 17. Within this framework, D represents a vector of length p (the number of hubs), which indicates the anticipated demand for each region. Each region is defined by a hub and its associated locations, with the demand vector being dynamic, subject to change throughout the day, possibly reflecting demand forecasts for specific time segments. Furthermore, N is a vector of length p that denotes the count of drone agents available in each region. The variable r signifies the current region of the agent. Finally, C, another vector of length p, records the repositioning cost to each hub.

The demand and cost vectors are normalized to increase the clarity of the data and improve learning stability. This normalization is necessary as the values within these vectors can become significantly large and challenging to interpret directly. In contrast, the distribution of drone agents remains naturally bounded by the total number of agents, thereby eliminating the need for normalization in their case.

$$S = [D, N, r, C] \tag{17}$$

Action Space

As outlined in Equation 18, the action space represents a vector of hubs a drone agent can reposition to, thus equating the number of actions to the hub count. In this investigation, we constrain our attention to a subset of locations, comprising approximately 10 to 20% of all possible locations. This constraint is imposed to uphold computational feasibility and provide proof of concept. For instance, in a study encompassing a network of 65 customer locations, 9 locations are designated as repositioning hubs. The complexity of including all locations exceeds the current scope, necessitating a simplified approach. While this methodology demonstrates key ideas and strategies, a full-scale application would require more advanced methods and computational power.

$$A = [h_1, h_2, ..., h_p] \tag{18}$$

Rewards

The primary goal of the strategy outlined in this paper is to maximize cost efficiency. We have explored various reward mechanisms contributing to cost reduction to achieve this. Through our investigation, network availability emerged as the most effective reward metric. We define *availability*, used interchangeably with *delivery success rate*, as the ratio of orders completed by drone agents to the total orders placed within the network, as shown in Equation 19. Optimizing for higher order fulfillment with fewer drones leads to a leaner fleet, reducing maintenance and depreciation expenses. Additionally, this efficiency facilitates easier expansion of the network.

$$R = \frac{\text{completed orders by drones}}{\text{total orders}} \tag{19}$$

Terminal State

The terminal state is reached at end of a day, $t = 1440$. Therefore, regardless of the agents' actions the algorithm wil reach the terminal state.

4.3.3 Training Process

The training process for repositioning agents is depicted in Figure 3. At every timestep, the model executes operations as previously described, during which it identifies drone agents requiring repositioning. The primary criterion for flagging involves drones that have completed a flight and do not have any subsequent flights scheduled. Upon completing a timestep, the model temporarily exits the simulation environment, and a repositioning action is implemented for each flagged drone using a policy π, initiated randomly at the start of the training process. Each repositioning action, along with its corresponding state, is recorded.

This procedure is repeated continuously throughout the simulation. After concluding the simulation, the algorithm calculates the reward for the entire simulation and associates this reward with the recorded state-action pairs. The accumulation of state-action pairs and their corresponding rewards continues until the predetermined batch size is reached. At this point, the policy undergoes an update process, using the collected rewards and state-action pairs.

As mentioned, our approach utilizes a policy gradient algorithm augmented with a baseline. The incorporation of a baseline serves to mitigate variance and accelerate the learning phase. Our algorithm's update process, illustrated in Figure 3, adheres to the formula specified in Equation 20. Within this context, the vector $\boldsymbol{\theta}$ comprises the policy's parameters, α signifies the learning rate, γ the discount factor, and δ represents the discrepancy between the actual reward R and the estimated value $\hat{v}(S_t, \boldsymbol{w})$. Here, π denotes the policy, and $\boldsymbol{w}$ are the weights of the value function.

$$\boldsymbol{\theta_{t+1}} = \boldsymbol{\theta_t} + \alpha\ \gamma\ \delta\ \nabla \ln\ \pi(A_t|S_t, \boldsymbol{\theta}) \tag{20}$$

$$\delta = R - \hat{v}(S_t, \boldsymbol{w})$$

Table 2 outlines the hyper parameters for our RL setup. Firstly, the Adam optimizer is employed for its adaptive nature, dynamically adjusting learning rates for individual parameters based on past gradients and variances. This adaptability promotes faster convergence and enhanced performance compared to conventional optimizers like vanilla stochastic gradient descent (Ruder, 2016).

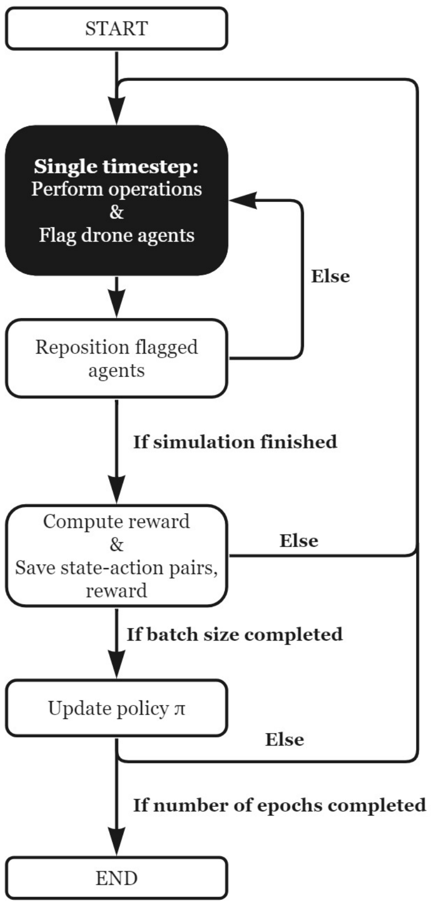


Figure 3: Flow diagram of repositioning training process.

Additionally, the learning rate policy (λ_π) and learning rate value function (λ_v) are set at 0.001, ensuring stable and gradual updates to the policy and value function parameters, thus preventing abrupt changes that could lead to instability or divergence in learning. Higher learning rates were experimented with but did not lead to policy convergence.

We deviate from conventional practice in the choice of the discount factor (γ), which is set to 1 instead of the more commonly used value closer to 0.99. This discrepancy influences the policy's behavior by assigning equal importance to immediate and future rewards, potentially overestimating the significance of long-term consequences. Adjusting the discount factor to align with standard practices in reinforcement learning may lead to a policy that better captures the dynamics of the problem.

Lastly, the simulation horizon (t_{end}) is set at 1440, reflecting the maximum number of time steps in a simulation episode and allowing the algorithm to capture long-term patterns and dependencies in the environment, possibly corresponding to the duration of a day in real-world applications.

The neural networks for both the policy and the value function leverage PyTorch's nn.Module, enabling us to construct multi-layer perceptrons (MLPs) that effectively map state inputs to either action probabilities or value estimates. We opted for MLP architectures due to their straightforward design and ease of implementation. In these architectures, input features extracted from the state are first processed by individual fully connected layers. Subsequently, these features are concatenated and passed through two more layers for further refinement. The final layer employs a softmax function for the policy network to normalize the outputs into action selection probabilities.

Table 2: Overview of all learning parameters

Hyper-Parameter	Value
Optimizer	Adam
Learning rate policy (λ_π)	0.001
Learning rate value function (λ_v)	0.001
Discount factor (γ)	1
Simulation horizon (t_{end})	1440

5 Case Studies

To illustrate the effectiveness and adaptivity of our proposed strategy, we conducted a case study. This case study comprises three distinct experiments, each designed to assess the strategy's performance in varying contexts: demand dynamics, order urgency levels, and the utilization of a hybrid fleet. These experiments show that the devised strategy is capable of handling various scenarios, making it highly suitable for growing networks. Detailed descriptions of these experiments are provided in the subsections that follow.

5.1 Experimental Set Up

Our experiments are set in the central-western region of the Netherlands, the operational country of MDS. Spanning from The Hague and Rotterdam in the West to Arnhem in the East, this area offers diverse testing conditions with densely populated zones in the West and sparser regions in the East, including bodies of water and natural reserves. This area was discretized into a grid containing squares of 500x500 meters, leading to a grid size of 273x187. Customers are placed in this grid based on the locations of actual Dutch hospitals.

For the optimization of agent schedules, Gurobi 9 was used to perform the optimization. For repositioning the policy was trained until the reward stabilized, of which further details are found in Appendix A.

All simulations are conducted on an 8-core Intel Core i7 chip with 16 GB RAM. For each parameter setting 30 simulations were run. This is the number for which the coefficient of variance ($\frac{\sigma}{\mu}$) became constant. More elaborate substantiation for this choice can be found in the appendix of the thesis report.

In addition to the variable factors tested in the experiments, several model inputs were held constant to provide a stable baseline for comparison. These static inputs, used consistently unless otherwise stated, are listed for reference in Table 3.

Table 3: Overview of static model inputs

Input	Value
Payload capacity	3 kg
Turnaround time	5 min
Order mass	0.5 kg
Speed	90 km/h
Range	100 km
Number of customer locations	65
Number of repositioning hubs	9
Variable flight cost	1 €/ min
Realloction threshold	1.4 [-]
Fixed flight cost	20 €/ flight
Fraction urgent orders	0.6
Fraction semi-urgent orders	0.3
Fraction same-day orders	0.1
Simulations per setting	20
Demand spread	Medium

5.2 Experiment A: Demand Spread

In our first experiment, we assess the adaptability of our strategy (allocation, reallocation, and repositioning) across three distinct demand patterns. Here, adaptability entails dynamic resource allocation adjustment to accommodate diverse demand spread patterns, ensuring efficient and effective delivery operations under unpredictable scenarios. We aim to maintain consistent performance across these varied patterns, showcasing the strategy's adept response to varying scenarios.

For this experiment, we maintained a constant total of 1200 orders for each simulation and explored three methods for distributing these orders among customers. The first approach implements a high demand spread, evenly dispersing orders across all network locations. Conversely, the second strategy concentrates the majority of orders within specific hubs. The third approach involves a medium demand spread with additional orders allocated to hospital locations near selected hubs. Visual representations of demand distributions are presented in Figure 4, Figure 5, and Figure 6.

Below, the reader can find the hypotheses for this experiment. The first three hypotheses assess the strategy's robustness to different demand spread inputs, aiming to demonstrate stable performance across scenarios. Subsequently, we focus on the contributions of the reallocation and repositioning modules to overall performance. Given its core objective of reducing costs and the observed side effect of a more evenly balanced workload, we anticipate the reallocation module will benefit all three KPIs. Furthermore, repositioning is expected to enhance the delivery success rate and fleet utilization as a result of more efficient resource distribution. However, this may come at the cost of higher operational expenses, attributed to the inclusion of empty flights in the drones' schedules. All hypotheses are tested with a significance level of $\alpha = 0.05$. Results for this experiment can be found in section 6.1.

- H_{A1}: *The difference in delivery success rate performance of the fleet management strategy among the three demand spread scenarios (high, medium, and low) is less than 10%.*
- H_{A2}: *The difference in costs per delivery performance of the fleet management strategy among the three demand spread scenarios (high, medium, and low) is less than 10%.*
- H_{A3}: *The difference in fleet utilization of the fleet management strategy among the three demand spread scenarios (high, medium, and low) is less than 10%.*
- H_{A4}: *Reallocation, as part of the fleet management strategy, reduces the costs per delivery by more than 5% for every demand spread scenario.*
- H_{A5}: *Reallocation, as part of the fleet management strategy, improves the delivery success rate by more than 5% for every demand spread scenario.*
- H_{A6}: *Reallocation increases fleet utilization by more than 5% for every demand spread scenario.*
- H_{A7}: *Repositioning, as part of the fleet management strategy, increases the delivery success rate by more than 5% for every demand spread scenario.*
- H_{A8}: *Repositioning, as part of the fleet management strategy, increases the costs per delivery by less than 5% for every demand spread scenario.*
- H_{A9}: *The effect of repositioning on the delivery success rate is the highest in scenarios with low demand spread.*
- H_{A10}: *Repositioning increases fleet utilization by more than 5% for every demand spread scenario.*

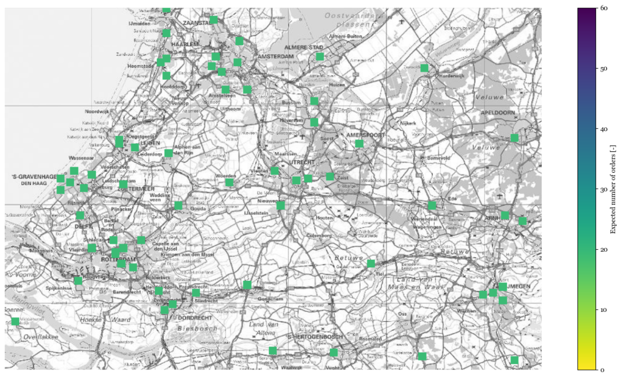


Figure 4: Map depicting high demand spread, where the intensity of demand at each location is indicated by color variation.

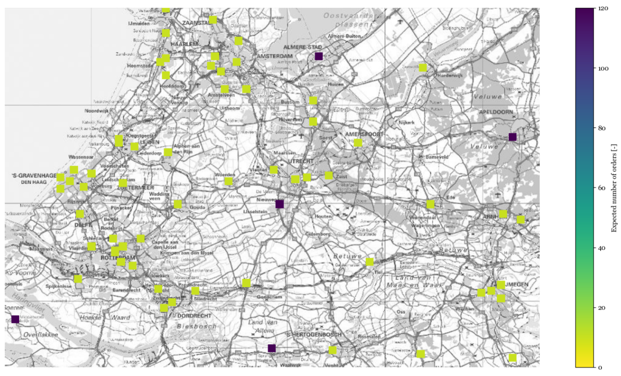


Figure 5: Map depicting a low demand spread, where the intensity of demand at each location is indicated by color variation.

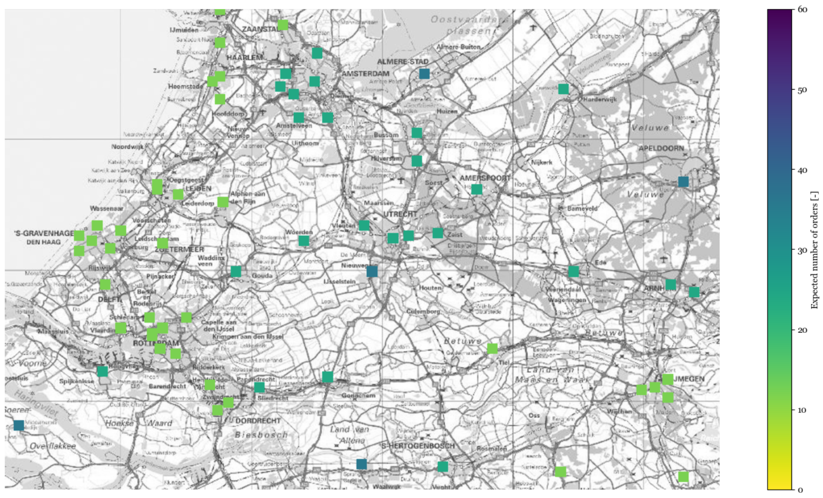


Figure 6: Map depicting a medium demand spread, where the intensity of demand at each location is indicated by color variation.

5.3 Experiment B: Urgency Levels

In our second experiment, we delve deeper into the fleet management strategy by focusing on the urgent orders present in the delivery network. These orders, characterized by their strict time constraints requiring completion within an hour of publication, introduce heightened stochastic challenges due to their unplannable nature. Our objective is to demonstrate the robustness of our strategy for increasing shares of urgent orders, showcasing its effectiveness in navigating these time-sensitive deliveries.

Each trial within this experiment maintains a consistent number of orders across medium-spread demand scenarios to ensure fair testing conditions. As we progress through the scenarios, we systematically increase the proportion of urgent orders from 40% to 60% and 80%, with the remaining orders evenly distributed between semi-urgent and same-day deliveries.

The hypotheses for this experiment are outlined below. The initial three hypotheses investigate the strategy's robustness to varying order compositions, aiming to showcase consistent performance despite these fluctuations. The last two hypotheses focus on evaluating the efficacy of the reallocation and repositioning modules. We anticipate these modules to exhibit the highest effectiveness in scenarios with 80% urgent orders, given their heightened demand for adaptability and flexibility. We aim to test every hypothesis with a significance of $\alpha = 0.05$. The results of this experiment are detailed in section 6.2.

- H_{B1}: *The difference in delivery success rate performance among the three distinct order compositions is less than 10%.*
- H_{B2}: *The difference in cost per delivery performance among the three distinct order compositions is less than 10%.*
- H_{B3}: *The difference in fleet utility performance among the three distinct order compositions is less than 10%.*
- H_{B4}: *Repositioning is the most effective for order compositions with 80% urgent orders.*
- H_{B5}: *Reallocation is the most effective for order compositions with 80% urgent orders.*

5.4 Experiment C: Hybrid Fleet

Our final experiment aims to demonstrate the advantage of employing a hybrid drone fleet with two drone types to enhance efficiency and flexibility. This approach illustrates how varying drone capabilities can be optimized to address diverse delivery demands.

The first drone type, characterized by its speed of 90 km/h, a payload capacity of 3 kg, and a range of 100 km, is suited for urgent deliveries that require swift transportation over longer distances. The second drone type, with its 10 kg payload capacity, 25 km range, and 75 km/h speed, is tailored for heavier deliveries within more localized areas. By integrating these two drone types, our experiment seeks to showcase a balanced approach to managing a variety of delivery scenarios, from time-sensitive orders to those necessitating larger payloads.

During the experiment, demand consists of a fixed number of orders with a medium spread, ensuring a fair and unbiased assessment. We test, several fleet compositions as displayed in Table 4.

Table 4: Overview of fleet compositions used for the experiment.

Fleet Composition	**Short Range High Payload**	**Long Range Low Payload**
1	0%	100%
2	10%	90%
3	20%	80%
4	30%	70%

The effectiveness of the fleets is evaluated based on three primary performance metrics: the total number of successful deliveries, the cost per delivery, and fleet utility. A diversified fleet is anticipated to outperform a uniform fleet in cost efficiency, utility, and delivery success rates. This outcome is attributed to the expectation that a larger payload capacity will facilitate the consolidation of orders, thereby enabling more efficient routing and delivery execution. Based on these expectations, we formulated the hypotheses below, which we aim to test with a significance level of $\alpha = 0.05$. This experiment aims to demonstrate the efficiency of hybrid fleets and leverage their associated benefits. Further insights can be found in section 6.3.

These expectations from the basis for the following hypotheses:

- *Hypothesis* H_{C1}: *Costs per delivery for hybrid fleets will be lower compared to a homogeneous fleet.*
- *Hypothesis* H_{C2}: *The delivery success rate for hybrid fleets will be higher compared to a homogeneous fleet.*
- *Hypothesis* H_{C3}: *The utility for hybrid fleets will be higher compared to a homogeneous fleet.*
- *Hypothesis* H_{C4}: *More than 50% of flights peformed using the short range drones, utilize the 10 kg payload capacity to at least 75% of its full potential.*

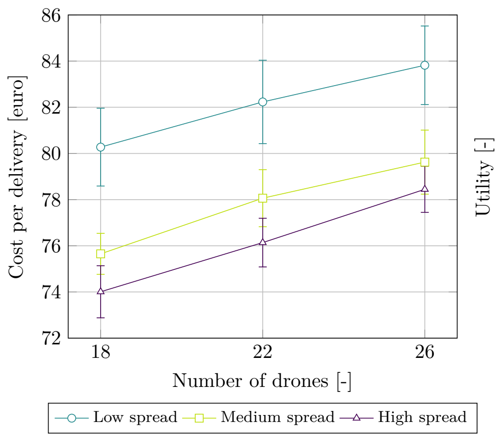
6 Results

In this section, we present the findings from a series of experiments designed to evaluate the performance of our strategy under varying operational conditions. The results are structured into three subsections, each dedicated to a specific experiment.

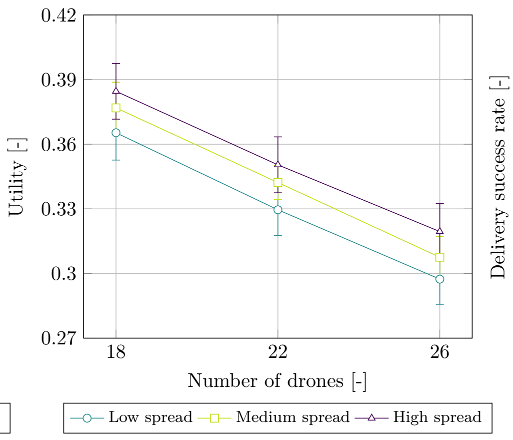
6.1 Experiment A: Demand Spread

Figure 7 provides insights into the performance of the complete fleet management strategy across three key performance indicators (KPIs): cost per delivery, fleet utility, and delivery success rate.

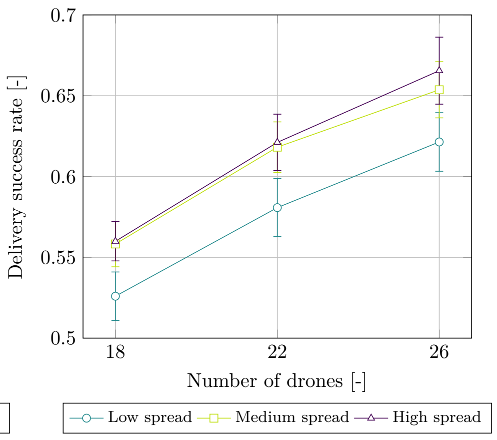
Cost per delivery is defined as the sum of fixed costs associated with each performed flight and the variable cost based on the duration of each flight. It increases with the growing number of drones due to the growing number of repositioning movements as shown in Figure 15 in Appendix C. *Fleet utility* is defined as the fraction of time the fleet spends flying with payload on board. It decreases as the fleet size expands, indicative of underutilization of resources. *Delivery success rate* is defined as the fraction of orders performed by drones. It rises proportionally with fleet size, indicating improved capacity to handle the workload effectively.



(a): Costs per delivery



(b): Utility



(c): Delivery success rate

Figure 7: Comparison of different performance metrics across varying demand spreads and fleet sizes.

We compare the worst and best-case data for each parameter setting to test the strategy's adaptability. In the graph above, these cases correspond to low-spread and high-spread scenarios. Notably, the high-spread scenario consistently demonstrates the best performance, while the low-spread scenario exhibits the worst performance across all parameters. These observations may be explained by the heightened consolidation of orders into trips when demand spread increases, as depicted in Figure 14 of Appendix C.

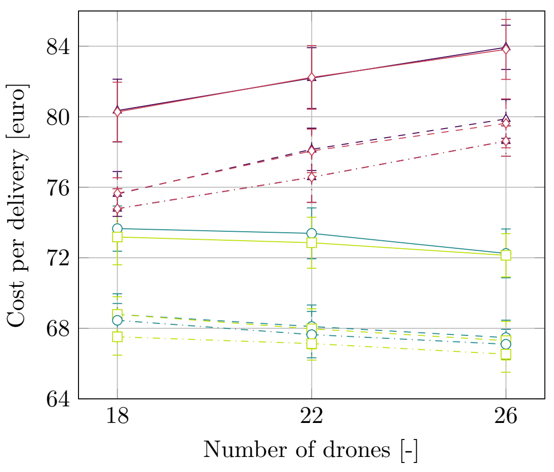
Only for the delivery success rate do we observe that the performance for the medium demand spread is very close to that of the high demand spread. Specifically, these datasets are statistically indifferent for drone numbers 18 and 22. However, data from 26 drones exhibit statistically significant differences, with the high-spread scenario performing best. Therefore, we compare the performance of high and low-spread scenarios in Table 5.

Table 5: Statistical data of comparison of low demand spread input scenario to high demand spread scenario Differences are stated with respect to the high demand-spread scenario.

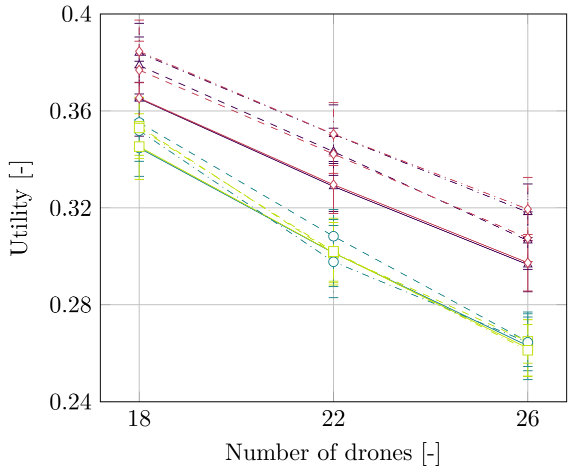
Numer of drones	**KPI**	**Shapiro-Wilk test p-value**	**Mean observed difference (95% CI)**	**One-sample one-tailed t-test p-value**
18	Cost per delivery	0.187	8.50% ± 1.17%	6.67e-3
	Utility	0.235	-4.88% ± 1.83%	1.68e-6
	Delivery succes rate	0.697	-6.02% pm 1.32%	5.38e-7
22	Cost per delivery	0.837	8.02% ± 0.993%	1.60e-4
	Utility	0.296	-5.80% ± 2.02%	1.01e-4
	Delivery succes rate	0.495	-6.42% ± 1.48%	1.50e5
26	Cost per delivery	0.315	6.86% ± 0.830%	7.86e-4
	Utility	0.967	-6.75% ± 2.04%	1.47e-3
	Delivery succes rate	0.550	-6.55% ± 1.34%	6.15e-6

The comparison is conducted by calculating the performance differences for each iteration. These iterations are paired, meaning that the same input is utilized for each comparison. We verify the normality of the computed differences using the Shapiro-Wilk test, which indicates non-normality when the p-value falls below 0.05. Subsequently, the observed differences are presented alongside a 95% confidence interval (CI), and a one-tailed one-sample t-test is performed to determine if these means are below the absolute value of 10%. Each p-value in Table 5 is found to be below 0.05, thereby allowing us to accept hypotheses H_{A1}, H_{A2}, and H_{A3} with a significance of $\alpha = 0.05$.

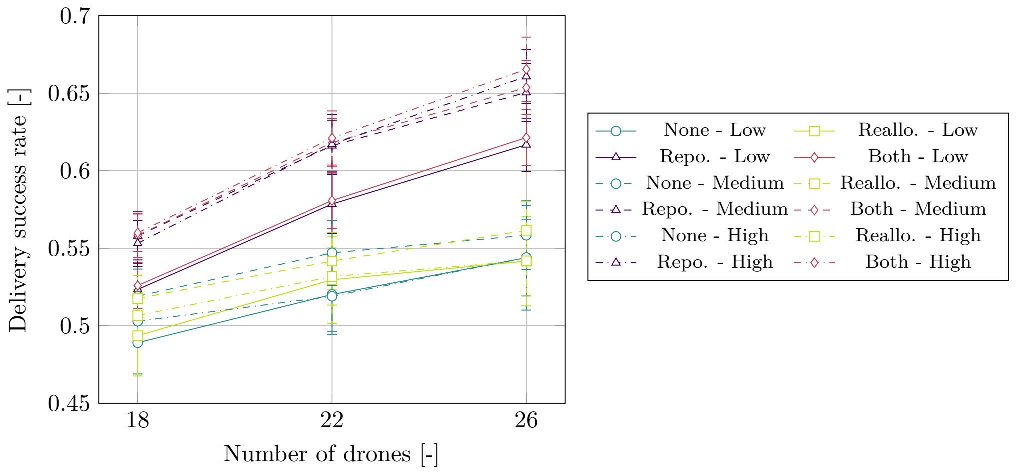
The graphs in Figure 8 illustrate the performance of the fleet management strategy under various configurations. Simulations were conducted with different modules of the strategy deactivated. Notably, repositioning significantly enhances utility and delivery success rates compared to the original strategy, albeit at an increased cost. Conversely, the effects of reallocation appear to be less pronounced. The plots show that the full fleet management strategy closely mimics the performance of the strategy with the repositioning module only.



(a): Cost per delivery



(b): Utility



(c): Delivery success rate

Figure 8: Performance of strategy including and excluding reallocation (realloc.) and repositioning (repo.) modules and for various demand spreads.

This observation is further supported by the statistical analyses presented in Table 10 of Appendix B. Regarding costs per delivery, all conducted statistical tests aimed at confirming a decrease of 5% have failed, leading us to reject H_{A4}. The observed decreases range from -1.34% to -0.14%, with no statistically significant evidence to suggest a consistent reduction. It is noteworthy that the reallocation module exhibits its largest contributions to mitigating costs under high-spread demand scenarios. The analysis presented in Figure 17 provides insights into the underlying factors contributing to these results. While a considerable number of orders are attempted for reallocation (10 to 20%), the figures reveal that only a small fraction achieves successful replanning. This outcome stems from the circumstance where alternative agents are not deemed more suitable for the given orders, resulting in the rejection of reallocation attempts. Despite this, the substantial cost savings achieved through reallocation underscore the potential benefits of effective order reallocation as shown in Figure 18.

Similarly inconclusive are the findings concerning the effects of reallocation on utility and delivery success rate, with no indication of a 5% increase. Consequently, we reject both H_{A5} and H_{A6}.

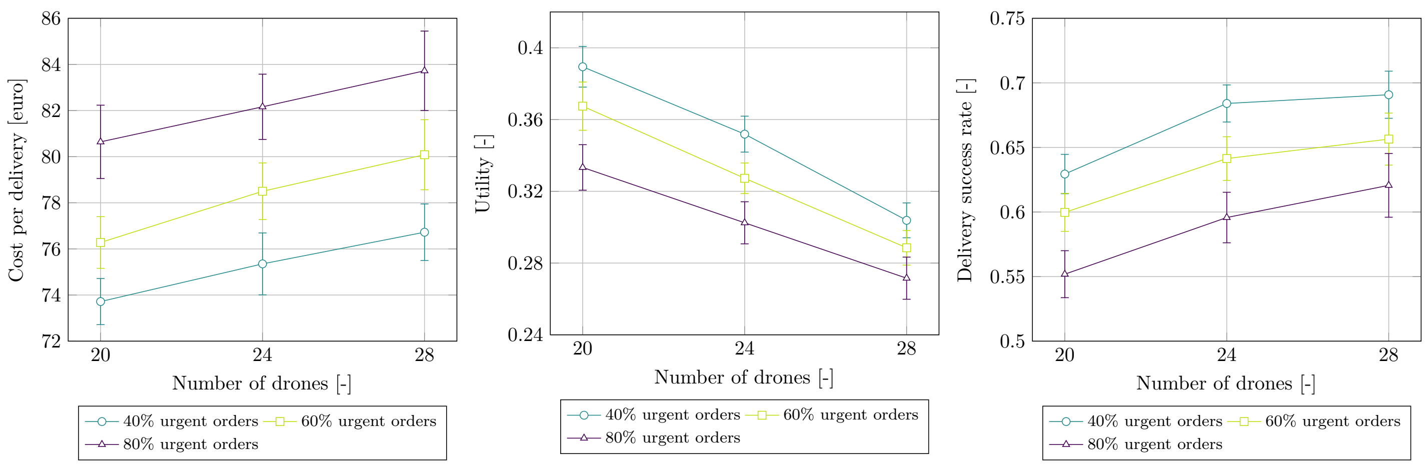
Nevertheless, the statistical analysis presented in Table 11 of Appendix B regarding the impact of repositioning reveals notable disparities. Firstly, implementing the repositioning module results in a considerable escalation in operational costs, substantiated by failed statistical tests displayed in Appendix B. Consequently, we reject H_{A8}, posing a marginal increase of 5%. The observed increments vary significantly, with the smallest being 9.1% and the largest reaching 18.4%. Additionally, this escalation demonstrates an amplifying trend as the fleet size expands.

However, despite the cost escalation, repositioning also improves fleet utility and delivery success rate, demonstrating values ranging from 5.94% to 21.71%. The statistical tests for these two KPIs yield predominantly favorable outcomes, with only one exception. Consequently, we accept H_{A7} and H_{A10}, affirming the effectiveness of repositioning in improving fleet utility and enhancing the delivery success rate.

To assess the validity of hypothesis H_{A9}, which posits that repositioning is most effective for low-demand spread scenarios, we conducted the Kruskal-Wallis test to examine whether the enhancements across different demand spreads exhibit statistical distinctions. For drone quantities of 18 and 22, the test yielded respective p-values of 0.61 and 0.58, leading to the rejection of the hypothesis. Conversely, the hypothesis is accepted for a drone quantity of 26 with a p-value of 0.022. However, it is noteworthy that the observed enhancements across two-thirds of the fleet sizes lacked statistical significance. Furthermore, the observed improvements do not align with the assumption of increased effectiveness for low-spread demand scenarios. Consequently, we reject hypothesis H_{A9}. Visualisations of repositioning movements throughout the simulation were also analysed in Figure 19, Figure 20 and Figure 21 of Appendix C. However, these did not show any conclusive patterns for the three different spreads.

6.2 Experiment B: Urgency Levels

In the second experiment, we explore the impact of urgent orders on fleet management effectiveness and the role of reallocation and repositioning in improving performance. Figure 9 illustrates fleet management strategy performance in terms of delivery costs, fleet utility, and success rate across different fleet sizes and urgent order proportions. Results indicate that performance is poorest when 80% of orders are urgent, while it peaks at 40%. This discrepancy arises due to two factors. First, as urgent orders decrease, trip efficiency improves, as seen in Figure 22 of (Appendix E). Second, fewer repositioning movements lead to cost reductions, evident in Figure 23 of (Appendix E). Similar trends to Figure 7 are observed with increasing drone numbers.



(a): Costs per delivery (b): Utility (c): Delivery success rate

Figure 9: Comparison of different performance metrics across varying order compositions and fleet sizes.

Similar to the comparison in section 6.1, we evaluate the discrepancy between the worst and best performances to establish whether performance remains within acceptable bounds. Presented in Table 6, the table compares the performance metrics under the 80% urgent orders scenario against those under the 40% urgent orders scenario. The table first verifies the normality of the data using the Shapiro-Wilk test and subsequently presents the observed changes in performance alongside their respective 95% confidence intervals. Subsequently, one-sample one-tailed t-tests are conducted to assess hypotheses H_{B1}, H_{B2}, and H_{B3}.

For the fleet utility and delivery success rate, statistical tests aimed at demonstrating a performance decrease of less than 10% failed to meet the significance criteria. Consequently, hypotheses H_{B1} and H_{B3} are rejected. However, the observed increase in cost per delivery is slightly less pronounced. While most tests support the hypothesis that costs remain within a 10% threshold, one test narrowly falls short of significance ($p = 0.059$), prompting the rejection of H_{B2}. Further analysis indicates that adjusting the bound to 10.1% results in all three tests yielding statistically significant results (p-values of 0.035, 0.014, and 0.024, for drone quantities 20, 24 and 28 respectively).

Table 6: Statistical analysis comparing the performance between order compositions with 80% urgent orders and 40% urgent orders. Percentual differences are expressed relative to the performance observed under the 40% urgent order composition.

Number of drones	**KPI**	**Shaprio-Wilk test p-value**	**Mean observed difference (95% CI)**	**One-sample one-tailed t-test p-value**
20	Cost per delivery	0.128	9.40% ± 0.76%	0.059
	Utility	0.991	-14.37% ± 1.24%	1.00
	Delivery success rate	0.492	-12.29% ± 1.11%	1.00
24	Cost per delivery	0.264	9.07% ± 0.92	0.023
	Utility	0.723	-14.02% ± 1.14%	1.00
	Delivery success rate	0.756	-12.88% ± 1.18%	1.00
28	Cost per delivery	0.305	9.14% ± 0.95%	0.037
	Utility	0.390	-10.56% ± 1.57%	0.763
	Delivery success rate	0.978	-10.11% ± 1.48%	0.56

To evaluate for which order composition repositioning is most effective, we compare samples of performance improvements in Table 7 for each order composition using the the One-way ANOVA test. The data in table shows all p-values below 0.05, indicating that performance improvements are significantly different across the three order compositions, leading to further investigation.

Table 7: Results of a One-Way ANOVA test comparing samples representing the differences in performance between the strategy with repositioning and the strategy without repositioning across different order compositions, treated as independent variables.

KPI	**20 Drones**	**24 Drones**	**28 Drones**
Cost per delivery	2.57e-45	1.53e-30	8.36e-29
Utility	1.41e-5	8.03e-6	1.75e-12
Delivery success rate	1.8e-3	1.74e-4	5.56e-8

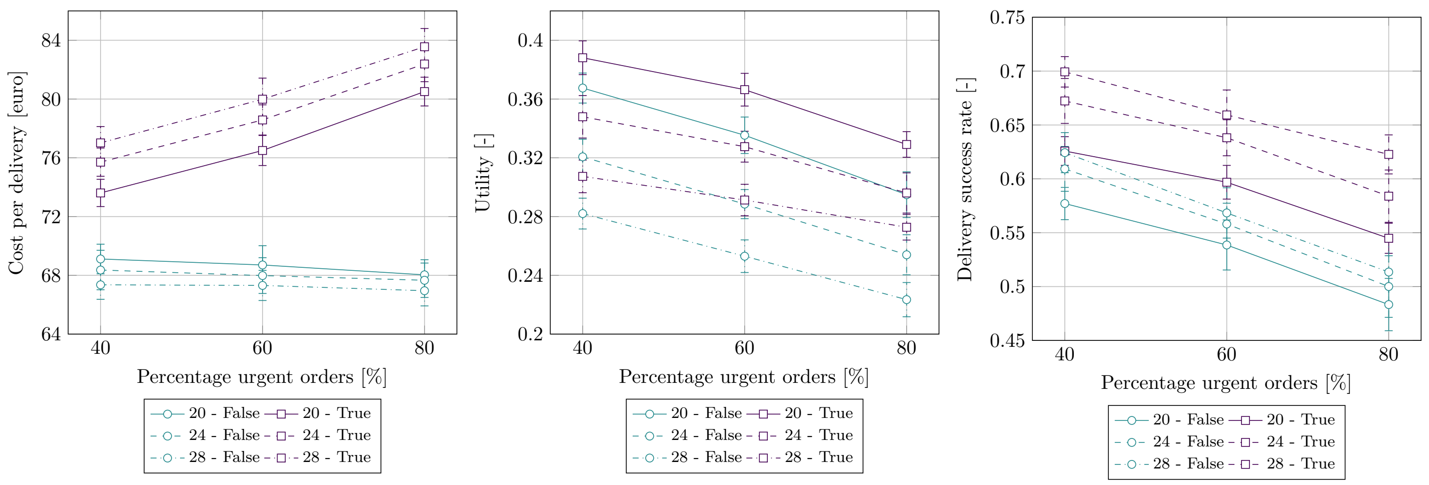
Consequently, the Tukey test is employed to determine the differences among samples of the effects of the repositioning module. The outcomes of the Tukey test are shown in Appendix D. Please note that the comparison revolves around the effects of repositioning rather than the actual performance, of which a visual is provided in Figure 10 for clarity.

The Tukey test on the cost per delivery reveals a trend: the influence of repositioning on costs increases as the fraction of urgent orders rises. Regrettably, this surge in costs is deemed unfavorable. Furthermore, the Tukey tests focusing on utility and delivery success rate indicate that repositioning is more effective in scenarios comprising 80% urgent orders than those comprising 40% urgent orders. However, comparisons between scenarios involving 60% and 80% urgent orders occasionally yield statistically insignificant outcomes. Notably, for the largest fleet size of 28 drones, statistically significant results emerge, confirming that the impact of repositioning is most pronounced in demand compositions containing 80% urgent orders. Consequently, we can accept H_{B4} for larger fleet sizes based on these findings.

For reallocation, we also start by performing the One-Way ANOVA test in Table 8. The data in table shows all p-values far above 0.05, indicating that performance improvements are not statistically different across the three order compositions. Therefore, we reject H_{B5}.

Table 8: Results of a One-Way ANOVA test comparing samples representing the differences in performance between the strategy with reallocation and the strategy without reallocation across different order compositions, treated as independent variables.

KPI	**20 Drones**	**24 Drones**	**28 Drones**
Cost per delivery	0.087	0.462	0.657
Utility	0.758	0.32	0.635
Delivery success rate	0.983	0.484	0.663



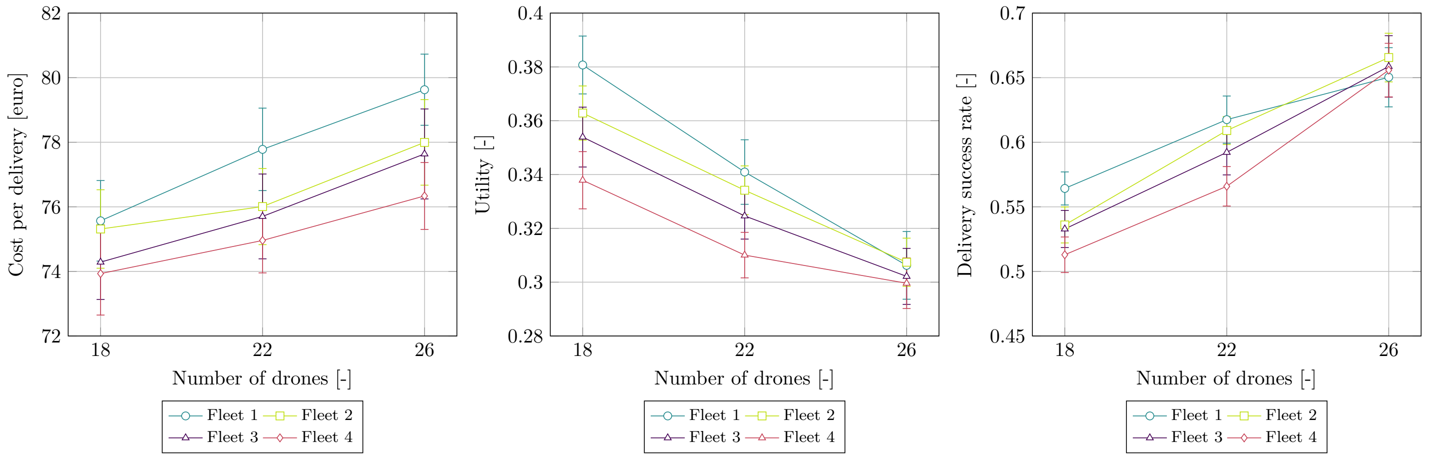
(a): Costs per delivery (b): Utility (c): Delivery success rate

Figure 10: Performance of strategy with repositioning (True) and without repositioning (False) for several drone quantities (20, 24, 28).

6.3 Experiment C: Hybrid Fleet

The concluding experiment assesses the performance of various hybrid fleets comprising long-range and short-range drones. The objective is to analyze the benefits of having such a fleet. Each fleet composition is denoted by a numerical identifier (the higher, the more short-range drones), as specified in Table 4 in the presentation of results.

Figure 11 illustrates the performance of fleets on key performance indicators (KPIs), including cost per delivery, fleet utility, and delivery success rate. Analysis of the graphs reveals that while a homogeneous fleet exhibits the highest cost per delivery, it also demonstrates a superior fleet utility and delivery success rate. The graphs show that with an increase in the proportion of short-range drones in the fleet, there is a corresponding decrease in cost per delivery, fleet utility, and delivery success rate. Additionally, as the fleet size grows, differences in performance metrics such as fleet utility and delivery success rate become less discernible. Nevertheless, outcomes from the One-way ANOVA statistical test indicate noticeable differences in fleet performances across most parameter settings.



(a): Costs per delivery (b): Utility (c): Delivery success rate

Figure 11: Comparison of different performance metrics across varying fleet compositions and fleet sizes.

Statistical findings presented in Table 15, Table 17, and Table 16 corroborate these observations, albeit with inconclusive data for certain fleet sizes. Specifically, we can support hypothesis H_{C1}, asserting the cost advantages of hybrid fleets for fleet sizes 22 and 26. However, statistical and graphical data analysis about delivery success rate suggests no discernible benefits associated with hybrid fleets. This conclusion holds primarily for a fleet size of 18, where we reject hypothesis H_{C2}. A similar trend is observed regarding the utility hypothesis H_{C3}, which we reject for fleet sizes 18 and 22. The reduction in performance for these KPIs is most likely attributed to the range constraints of short range drones.

Table 9: p-values of One-Way ANOVA test comparing performances of different fleet compositions for various KPIs.

KPI	**18 Drones**	**22 Drones**	**26 Drones**
Cost per delivery	4.34e-7	2.30e-14	8.64e-17
Utility	4.79e-29	6.18e-23	0.015
Delivery success rate	3.64e-26	3.23e-24	0.057

Figure 12 illustrates the average payload on board the two drone types across different fleets. Long-range drones demonstrate consistent payload-carrying capacity in all fleets, with a slight reduction observed as fleet size increases. Conversely, fleets comprising short-range drones exhibit less definitive patterns. Remarkably, none of the short-range drones carry an average payload exceeding 2 kilograms, suggesting an underutilization of their 10-kilogram capacity. Moreover, a noticeable decline in average payload onboard occurs with a fleet size of 22 drones. Delving into this decline in Appendix G, Figure 24 reveals significant variance in order allocation to each drone for hybrid fleets, indicating uneven distribution. This is corroborated by Figure 25, illustrating order distributions from sampled simulations, where the model favors long-range drones over short-range ones, potentially explaining the payload dip.

Each simulation sampled the percentage of short-range drone flights with a payload capacity exceeding 75% of the total capacity. Except for one simulation, where this percentage was 1.6%, all other instances resulted in a percentage of 0.0%. Consequently, it appears reasonable to dismiss H_{C4} without further statistical analysis.

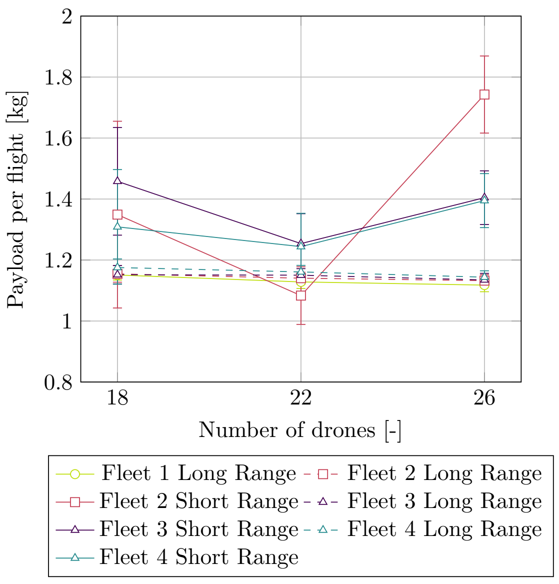


Figure 12: Payload for both drone types across four different fleet compositions for various fleet sizes.

7 Discussion

This section revisits and deliberates on the hypotheses set at the formulation of each experiment. By critically evaluating the outcomes against the initial predictions, we aim to draw insights and understand the implications of our findings within the broader context of the study.

Experiment A aimed to evaluate the adaptability of the fleet management strategy to different demand spreads and to assess the additional benefits of order reallocation and drone repositioning. Results from Experiment A demonstrated that the fleet management strategy maintained stable performance across various scenarios of demand spread, thereby supporting the acceptance of hypotheses H_{A1}, H_{A2}, and H_{A3}. However, findings indicated that the implementation of reallocation did not significantly enhance the effectiveness of the study, resulting in the rejection of hypotheses H_{A4}, H_{A5}, and H_{A6}.

Further analysis of the underlying data revealed that while the reallocation method attempted to redistribute a substantial proportion of orders, only a limited number of reallocation attempts proved successful. These findings suggest the possibility of either exceptionally optimal initial allocation or a subpar quality of the selection criteria used for reallocation. Consequently, exploring alternative selection criteria thoroughly and conducting a deeper analysis of the characteristics of orders being reallocated is recommended.

Given the potential cost-saving benefits associated with order reallocation, a nuanced understanding of task attributes such as urgency level, distance, and specific requirements could lead to more substantial cost reductions and enhance operational efficiency. Therefore, future research efforts should clarify the relationships between task attributes and cost savings to optimize reallocation.

Contrary to reallocation, the benefits of repositioning were evident. In the best scenario, the fleet utility and delivery success rate improved by approximately 20% due to the repositioning module, leading to the acceptance of hypotheses H_{A7} and H_{A10}. However, while increases in costs were anticipated, they exceeded expectations, resulting in the rejection of hypothesis H_{A8}.

To mitigate the cost increase, future research may explore adding more repositioning locations and consider incorporating the option to remain stationary within the action space used for the reinforcement learning algorithm. However, such enhancements must be applied with an alternative algorithm instead of the REINFORCE algorithm. A limitation of the REINFORCE algorithm is its slow learning process due to high variance, which restricts the expansion of the action space. Moreover, the current framework struggles to effectively attribute rewards to specific actions due to delayed rewards, a phenomenon known as the credit assignment problem. To clarify, the benefits of a repositioning action are not immediately evident and may not even directly impact the repositioning agent.

Foerster et al. (2018) and Nguyen et al. (2018) present two interesting studies on this subject. Although the project's scope precluded the adoption of these advanced methodologies, their potential for improving adaptability while mitigating cost increases justifies future exploration.

In Experiment B, we evaluated the effectiveness of the designed fleet management strategy under increasingly stochastic scenarios by augmenting the proportion of urgent orders, thereby demonstrating its adaptability to unpredictable demand. While the observed increase in cost per delivery remained only just within the predefined thresholds (H_{B2}), the magnitude of reduction in fleet utility and delivery success rate exceeded expectations, leading to the rejection of hypotheses H_{B1} and H_{B3}. Prioritizing the implementation of previously proposed alterations is expected to yield improved performance.

Furthermore, we examined whether the modules exhibited enhanced effectiveness in stochastic scenarios. Contrary to expectations, the reallocation module failed to demonstrate significant improvement (H_{B5}); however, it had already been established that this module had limited effectiveness. In contrast, the repositioning module yielded additional benefits (H_{B4}), emphasizing its usefulness.

The final experiment, Experiment C, aimed to quantify the benefits of a hybrid fleet for enhanced adaptability. While hybrid fleets reduced the cost per delivery (H_C1), this improvement came at the expense of fleet utility and delivery success rate (H_C2 and H_C3). The proposed short-range drone with increased payload capacity underutilized its potential (H_{C4}). Exploring alternative drone types is recommended to balance cost and service levels better. Moreover, refining the model's assumptions and inputs is essential to enhance the reliability of these findings for decision-making.

While we simulated demand using an origin-destination matrix and a Poisson process, this approach lacks the complexity of actual demand patterns. Developing a model based on customer demographics and historical ordering trends would offer deeper insights into demand behavior.

Furthermore, our current framework activates a backup network solely when drone capacity reaches its limits, neglecting other critical factors like adverse routing conditions and weather. Integrating a decision-making module to assess these factors would enhance the model's fidelity, mirroring real-world operational challenges more accurately. Additionally, our model considers nominal time for traveling. In reality, however, this is untrue. Lastly, assuming battery swapping as the primary means of replenishing drone energy presents logistical challenges, particularly in the scalability of operations; hence, a detailed battery model would offer a more accurate and practical representation of operational logistics. Such improvements would add realism to our simulations and provide a stronger foundation for conclusions to be drawn.

8 Conclusions

This paper presents and evaluates a fleet management strategy for dynamic and highly stochastic drone transportation networks handling pickup and delivery orders. The strategy comprises allocation via the Adapted Temporal Sequential Single-Item auction and two adaptive components: reallocation using a heuristic method and repositioning using a policy obtained via reinforcement learning with the REINFORCE algorithm. Our investigation centers on demonstrating the strategy's adaptability and the effectiveness of its adaptive components.

Our findings highlight the strategy's consistent efficiency across various demand scenarios, maintaining performance within predefined limits. Notably, the repositioning module significantly enhances the fleet utility and the fraction of served orders, albeit at the expense of increased cost per delivery. Conversely, the reallocation module causes minimal performance improvement. Under heightened stochasticity introduced by urgent orders, the strategy maintains stable costs per delivery while fleet utility and order fulfillment rates decline. Additionally, our investigation underscores the increasing benefits of repositioning in more stochastic scenarios. Moreover, exploring hybrid fleets reveals that while short-range high-payload drones can reduce cost per delivery, they compromise overall fleet utility and order fulfillment rates. Furthermore, we identify the underutilization of payload capacity in scenarios with orders weighing up to 2 kilograms for a drone with a payload of 10 kilograms.

For future research, refining the repositioning method to mitigate extra costs is essential, alongside enhancing the selection criteria for reallocation and delving deeper into the characteristics of successfully reallocated tasks. To strengthen the reliability of conclusions drawn from the model, refining demand modeling techniques and integrating assessments for adverse conditions are crucial for enhancing operational realism. Additionally, modeling drone battery life and charging will contribute to more accurate representations of operational dynamics. These advancements will contribute to more robust high-level decision-making.

References

Agatz, N., Erera, A., Savelsbergh, M., and Wang, X. (2012). Optimization for dynamic ride-sharing: A review. *European Journal of Operational Research*, 223:295–303.

ANWB Verkeersinformatie (2023). 17 procent meer files op de nederlandse wegen in 2023. https://www.anwb.nl/verkeer/nederland/verkeers-informatie/filezwaarte. Visited on 29-2-24.

Arslan, A., Agatz, N., Kroon, L., and Zuidwijk, R. (2018). Crowdsourced delivery: A dynamic pickup and delivery problem with ad-hoc drivers. *Transportation Science*, 53:222–235.

Beirigo, B. A., Schulte, F., and Negenborn, R. R. (2022). A learning-based optimization approach for autonomous ridesharing platforms with service-level contracts and on-demand hiring of idle vehicles. *Transportation Science*, 56(3):677–703.

Bélanger, V., Ruiz, A., and Soriano, P. (2019). Recent optimization models and trends in location, relocation, and dispatching of emergency medical vehicles. *European Journal of Operational Research*, 272:1–23.

Campuzano, G., Lalla-Ruiz, E., and Mes, M. (2022). *Computational Logistics*, volume 13557 of *Lecture Notes in Computer Science*, chapter The Dynamic Drone Scheduling Delivery Problem, pages 260–274. Springer.

Chen, S., Sharpanskykh, A., and Ermiş, G. (2022a). Multi-agent planning and coordination for automated aircraft ground handling. Master's thesis, Delft University of Technology.

Chen, X., Ulmer, M., and Thomas, B. (2022b). Deep q-learning for same-day delviery with vehicles and drones. *European Journal of Operational Research*, 298(3):939–952.

Daskin, M. and Maass, K. (2015). *The p-Median Problem*, pages 21–45. Springer International Publishing.

Demir, E., Syntetos, A., and van Woensel, T. (2022). Last mile logistics: Research trends and needs. *IMA Journal of Management Mathematics*, 33:549–561.

Dogterom, M. (2023). Medical drone service. Powerpoint.

Dorling, K., Heinrichs, J., Messier, G., and Magierowski, S. (2017). Vehicle routing problems for drone delivery. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 47(1):70–85.

Foerster, J., Farquhar, G., Afouras, T., Nardelli, N., and Whiteson, S. (2018). Counterfactual multi-agent policy gradients. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1).

Heap, B. and Pagnucco, M. (2013). Repeated sequential single-cluster auctions with dynamic tasks for multi-robot task allocation with pickup and delivery. In Klusch, M., Thimm, M., and Paprzycki, M., editors, *Multiagent System Technologies*, pages 87–100, Berlin, Heidelberg. Springer Berlin Heidelberg.

Hildebrandt, F., Thomas, B., and Ulmer, M. (2023). Opportunities for reinforcement learning in stochastic dynamic vehicle routing. *Computers & Operations Research*, 150.

Huang, H., Hu, C., Zhu, J., Wu, M., and Malekian, R. (2022). Stochastic task scheduling in uav-based intelligent on-demand meal delivery system. *IEEE Transactions on Intelligent Transportation Systems*, 23(8):13040–13054.

Khamis, A., Hussein, A., and Elmogy, A. (2015). *Multi-robot Task Allocation: A Review of the State-of-the-Art*, pages 31–51. Springer International Publishing, Cham.

Koenig, S., Keskinocak, P., and Tovey, C. (2010). Progress on agent coordiantion with coopeative auctions. In *Proceedings of the National Conference on Artificial Intelligence*, volume 3.

Korsah, G. A., Stentz, A., and Dias, M. B. (2013). A comprehensive taxonomy for multi-robot task allocation. *The International Journal of Robotics Research*, 32(12):1495–1512.

Kullman, N. D., Cousineau, M., Goodson, J. C., and Mendoza, J. E. (2021). Dynamic ride-hailing with electric vehicles. *Transportation Science*, 56(3):775–794.

Lagoudakis, M., Markakis, E., Kempe, D., Keskinocak, P., Kleywegt, A., Koenig, S., Tovey, C., Meyerson, A., and Jain, S. (2005). Auction-based multi-robot routing. In *Robotics: Science and Systems*, pages 343–350.

Liu, Y. (2019). An optimization-driven dynamic vehicle routing algorithm for on-demand meal delivery using drones. *Computers & Operations Research*, 111:1–20.

Macal, C. M. and North, M. J. (2009). Agent-based modeling and simulation. In *Proceedings of the 2009 Winter Simulation Conference (WSC)*, pages 86–98.

Macrina, G., Pugliese, L. D. P., Guerriero, F., and Laporte, G. (2020). Drone-aided routing: A literature review. *Transportation Research Part C: Emerging Technologies*, 120(102762).

Nasrollahzadeh, A., Khademi, A., and Mayorga, M. (2018). Real-time ambulance dispatching and relocation. *Manufacturing & Service Operations Management*, 20(3):467–480.

Nguyen, D. T., Kumar, A., and Lau, H. C. (2018). Credit assignment for collective multiagent rl with global rewards. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

Nunes, E. and Gini, M. (2015). Multi-robot auctions for allocation of tasks with temporal constraints. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 29, pages 2110–2116.

NZa-Magazines 04 (2022). Stand van de zorg 2022. https://magazines.nza.nl/nza-magazines/2022/04/3-trends-in-sectoren. Visited on 3-4-23.

Ruder, S. (2016). An overview of gradient descent optimization algorithms. *CoRR*, abs/1609.04747.

Sayarshad, H. and Chow, J. (2015). A scalable non-myopic dynamic dial-a-ride and pricing problem. *Transportation Research Part B*, 81:539–554.

Schmid, V. (2012). Solving the dynamic ambulance relocation and dispatching problem using approximate dynamic programming. *European Journal of Operational Research*, 2196(3):611–621.

Sutton, R. and Barto, A. (2018). *Reinforcement Learning: An Introduction.* Adaptive computation and machine learning series. The MIT Press, 2 edition.

Troudi, A., Addouche, S., Dellagi, S., and Mhamedi, A. E. (2018). Sizing of the drone delivery fleet considering energy autonomy. *Sustainability*, 10.

van Barneveld, T. (2016). The minimum expected penalty relocation problem for the computation of compliance tables for ambulance vehicles. *INFORMS Journal on Computing*, 28(2):370–384.

van Haasteren, J. (2022). Design and analysis of a uav assisted medical emergency delivery system. Master's thesis, Delft University of Technology.

Williams, R. J. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8:229–256.

Zhen, L., Wu, J., Laporte, G., and Tan, Z. (2023). Heterogeneous instant delivery orders scheduling and routing problem. *Computers & Operations Research*, 157:106–246.

Zipline (2023). Zipline press kit. https://www.flyzipline.com/press-kit. Visited on 27-3-23.

Appendices

A Repositioning Algorithm Training Process

The policy was trained using a training data set in which (combinations of) different regions in the model were subject to concentrated demand. A single simulation contained 1440 timesteps in which approximately 100 repositioning actions are conducted. One policy update used a batch of 10 simulations. This number is based on the diversity and complexity of the input scenarios, aiming to strike a balance between computational efficiency, convergence, and overfitting.

The training duration was extended until rewards exhibited stability. This stabilization phenomenon is graphically demonstrated in Figure 13, where an increase in the time window for the moving average corresponds to a convergence of rewards.

While pursuing further training could have been favorable, it is essential to acknowledge the practical constraints encountered. The training process had already consumed over four days. Given the resource-intensive nature of continued training within the existing framework, extending training further without reassessing computational resources or methodological adaptations became impractical.

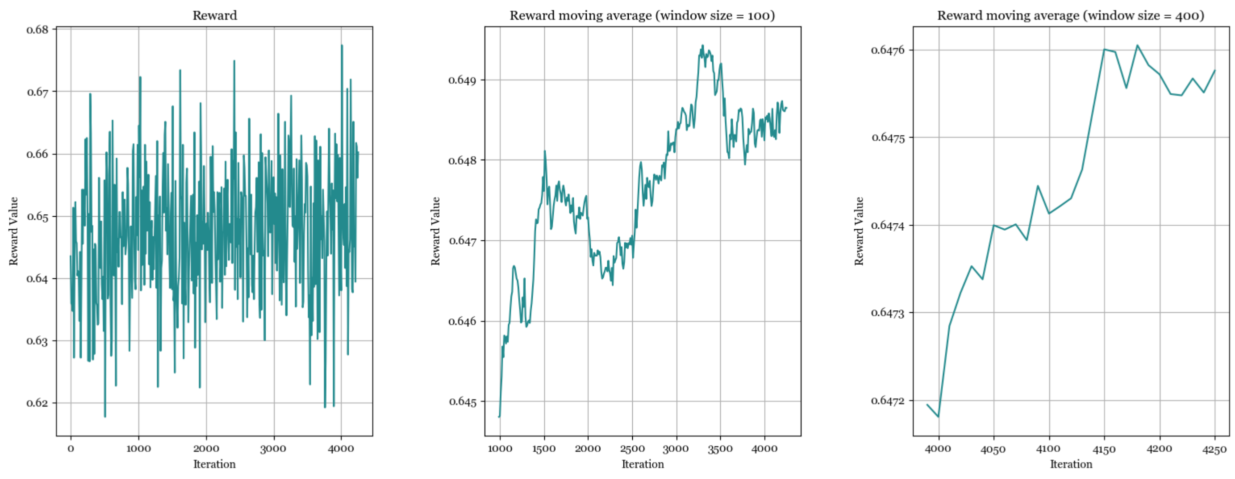


Figure 13: Convergence of rewards over training iterations with increasing time window for the moving average.

B Experiment A: Statistical Data

Table 10 compares the fleet management strategy with and without reallocation (repositioning is not included. Stated differences are respective to the strategy *without* the reallocation module. Depending on normality of the data the one-sample one-sided t-test or Wilcoxon test is performed to confirm the hypotheses. For cost per delivery we intend to confirm a decrease of 5% in cost and for utility and delivery success rate a 5% increase.

Table 10: Comparison of strategy with reallocation module to strategy without reallocation module.

Number of drones	**Demand scenario**	**KPI**	**Shapiro-Wilk test p-value**	**Mean observed difference**	**One-sample one-sided t-test/Wilcoxon p-value**
18	Low	Cost per delivery	0.268	-0.63% pm 0.70%	1.00
		Utility	0.053	0.22% pm 1.54%	1.00
		Delivery success rate	0.397	0.96% pm 1.71%	1.00
	Medium	Cost per delivery	0.751	0.02% pm 0.69%	1.00
		Utility	0.809	-0.41% pm 1.38%	1.00
		Delivery success rate	0.546	-0.18% pm 1.48%	1.00
	High	Cost per delivery	0.914	-1.34% pm 0.80%	1.00
		Utility	0.658	0.39% pm 1.57%	1.00
		Delivery success rate	0.387	0.80% pm 1.54%	1.00
22	Low	Cost per delivery	0.763	-0.72% pm 0.54%	1.00
		Utility	0.639	0.18% pm 1.83%	1.00
		Delivery success rate	0.459	1.94% pm 1.96%	1.00
	Medium	Cost per delivery	0.081	-0.17% pm 0.65%	1.00
		Utility	0.025	-2.20%	1.00
		Delivery success rate	0.075	-0.87% pm 1.51%	1.00
	High	Cost per delivery	0.161	-0.72% pm 0.75%	1.00
		Utility	0.549	1.55% pm 1.85%	1.00
		Delivery success rate	0.892	2.57% pm 1.53%	1.00
26	Low	Cost per delivery	0.752	-0.14% pm 0.40%	1.00
		Utility	0.792	-0.25% pm 1.58%	1.00
		Delivery success rate	0.564	-0.28% pm 1.47%	1.00
	Medium	Cost per delivery	0.977	-0.24% pm 0.61%	1.00
		Utility	0.690	0.17% pm 1.77%	1.00
		Delivery success rate	0.065	0.69% pm 1.73%	1.00
	High	Cost per delivery	0.527	-0.83% pm 0.67%	1.00
		Utility	0.546	-1.14% pm 1.58%	1.00
		Delivery success rate	0.511	-0.30% pm 1.41%	1.00

Table 11 compares the fleet management strategy with and without repositioning (reallocation is not included. Stated differences are respective to the strategy *without* the repositioning module. Depending on normality of the data the one-sample one-sided t-test or Wilcoxon test is performed to confirm the hypotheses. For cost per delivery we intend to confirm an increase less than 5% and for utility and the delivery success rate a 5% increase.

Table 11: Comparison of strategy with repositioning module to strategy without reallocation module.

Number of drones	**Demand scenario**	**KPI**	**Shapiro-Wilk test p-value**	**Mean observed difference**	**One-sampe one-sided t-test/Wilcoxon p-value**
18	Low	Cost per delivery	0.245	9.1% ± 0.83%	1.00
		Utility	0.439	5.94% ± 1.48%	0.101
		Delivery success rate	0.745	7.13% ± 1.48%	0.0031
	Medium	Cost per delivery	0.066	9.95% ± 0.81%	1.00
		Utility	0.904	6.71% ± 1.62%	0.020
		Delivery success rate	0.623	7.61% ± 1.56%	9.47e-4
	High	Cost per delivery	0.072	9.26% ± 0.65%	1.00
		Utility	0.933	9.20% ± 1.42%	7.29e-7
		Delivery success rate	0.190	10.05% ± 1.33%	6.69e-9
22	Low	Cost per delivery	0.834	12.02% ± 0.86%	1.00
		Utility	0.441	9.28% ± 1.90%	3.69e-5
		Delivery success rate	0.657	11.4% ± 1.86%	4.82e-8
	Medium	Cost per delivery	0.525	14.79% ± 0.92%	1.00
		Utility	0.349	11.44% ± 1.75%	1.30e-8
		Delivery success rate	0.784	12.8% ± 1.93%	2.09e-9
	High	Cost per delivery	0.907	13.23% ± 1.04%	1.00
		Utility	0.959	17.91% ± 2.46%	6.348e-12
		Delivery success rate	0.272	19.12% ± 2.39%	3.77e-13
26	Low	Cost per delivery	0.986	16.2% ± 0.71%	1.00
		Utility	0.558	12.98% ± 2.43%	1.13e-7
		Delivery success rate	0.800	13.75% ± 2.42%	1.82e-8
	Medium	Cost per delivery	0.210	18.4% ± 0.87%	1.00
		Utility	0.332	15.92% ± 2.35%	1.01e-10
		Delivery success rate	0.596	16.65% ± 1.95%	3.07e-13
	High	Cost per delivery	0.346	17.2% ± 0.90%	1.00
		Utility	0.115	20.48% ± 2.42%	5.486e-14
		Delivery success rate	0.063	21.71% ± 2.38%	5.096e-15

C Experiment A: Additional Graphs

C.1 Full Fleet Management Strategy

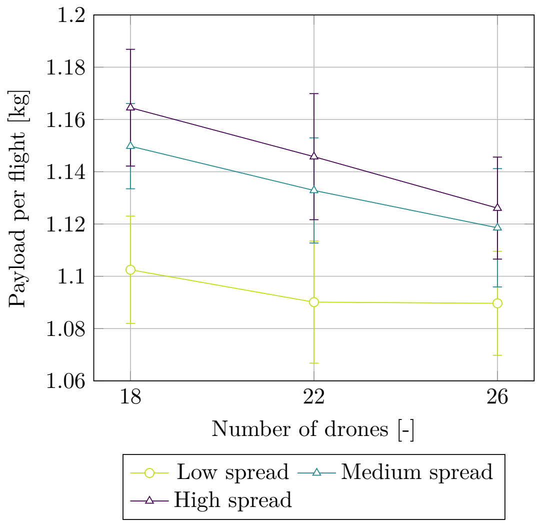


Figure 14: Average payload per trip for different demand spreads and fleet sizes.

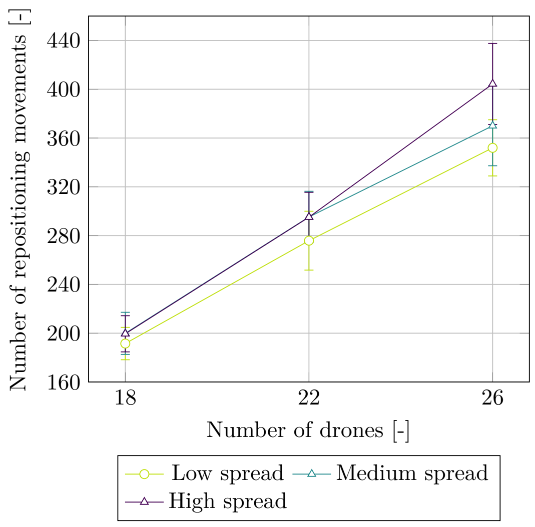


Figure 15: Average number of repositioning movements for different demand spreads and fleet sizes.

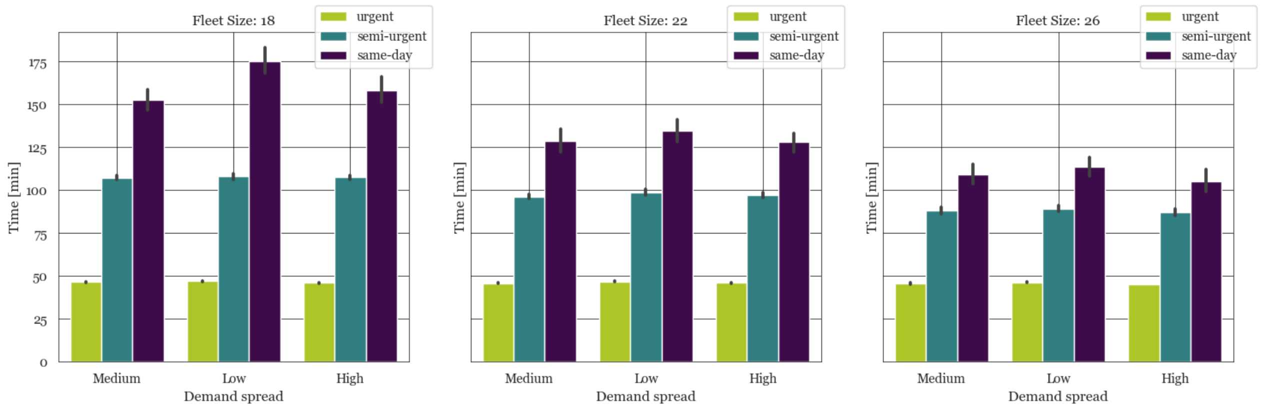


Figure 16: Time to order completion for different demand spreads and fleet sizes.

C.2 Reallocation Module

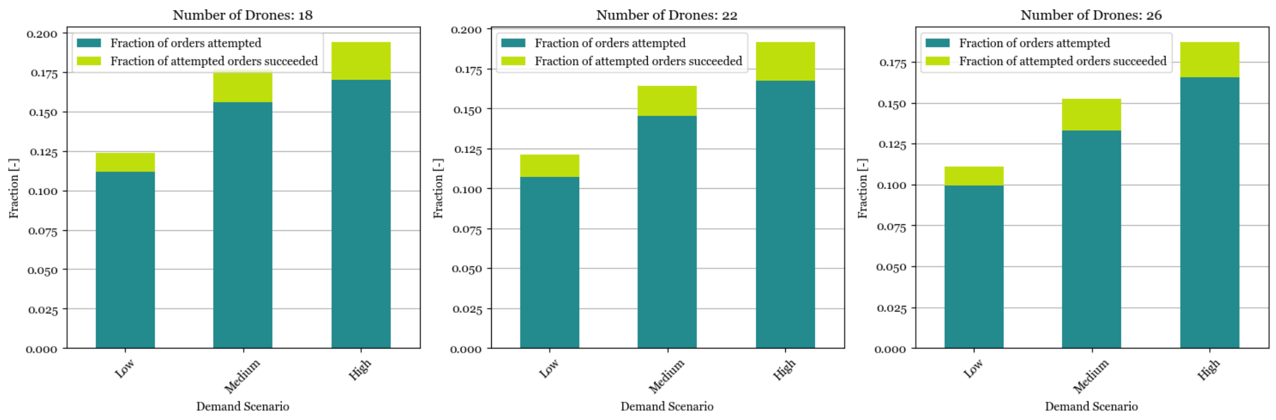


Figure 17: Fraction of attempted order replanning and their corresponding success rates.

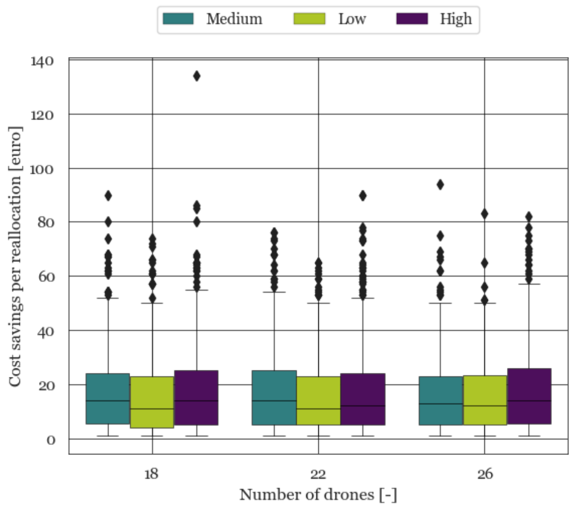


Figure 18: Spread of saved of costs due to reallocation of tasks for different fleet sizes and demand spreads.

C.3 Repositioning Module

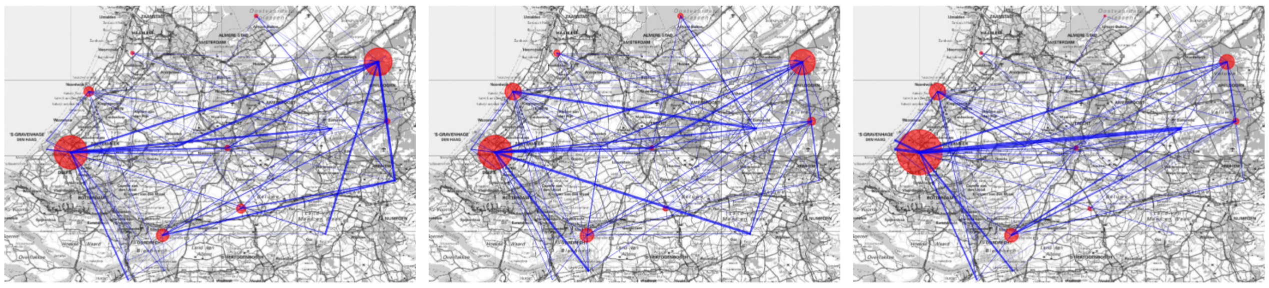


Figure 19: Repositioning movements for high demand spread scenarios (fleet size 22).

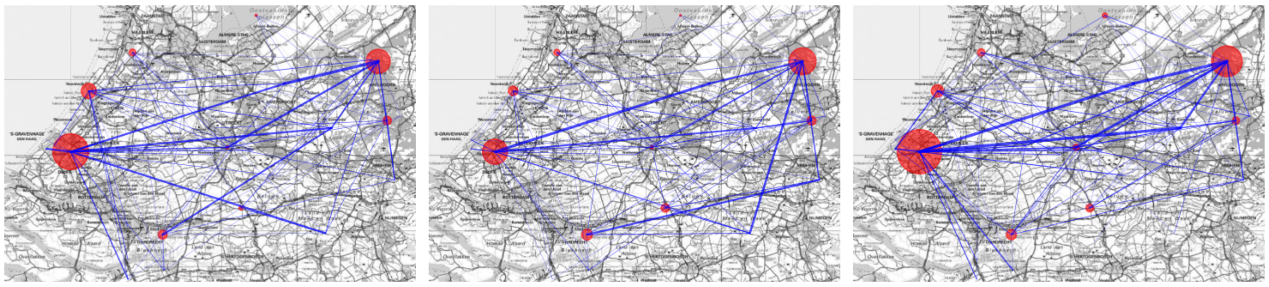


Figure 20: Repositioning movements for medium demand spread scenarios (fleet size 22).

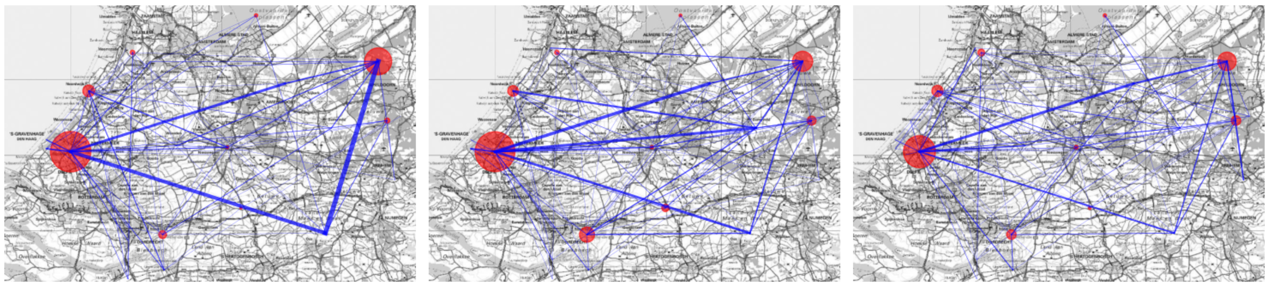


Figure 21: Repositioning movements for low demand spread scenarios (fleet size 22).

D Experiment B: Statistical Data

Table 12: Results of Tukey Test for performance differences in cost through repositioning across three orders compositions (40%, 60%, and 80% urgent orders).

	20 Drones				**24 Drones**				**28 Drones**			
Comparison	Statistic	p-value	Lower CI	Upper CI	Statistic	p-value	Lower CI	Upper CI	Statistic	p-value	Lower CI	Upper CI
40 - 60	-4.846	0.000	-5.827	-3.864	-4.838	0.000	-6.276	-3.400	-4.489	0.000	-5.941	-3.037
40 - 80	-11.831	0.000	-12.813	-10.850	-11.006	0.000	-12.444	-9.568	-10.468	0.000	-11.919	-9.016
60- 40	4.846	0.000	-12.813	-10.850	4.838	0.000	3.400	6.276	4.489	0.000	3.037	5.941
60 - 80	-6.986	0.000	-7.938	-6.004	-6.167	0.000	-7.606	-4.729	-5.979	0.000	-7.431	-4.527
80 - 40	11.831	0.000	10.850	12.813	11.006	0.000	9.568	12.444	10.468	0.000	9.016	11.919
80 - 60	6..986	0.000	6.004	7.968	6.176	0.000	4.729	7.606	5.979	0.000	4.527	7.431

Table 13: Results of Tukey Test for performance differences in utility through repositioning across three orders compositions (40%, 60%, and 80% urgent orders).

	20 Drones				**24 Drones**				**28 Drones**			
Comparison	Statistic	p-value	Lower CI	Upper CI	Statistic	p-value	Lower CI	Upper CI	Statistic	p-value	Lower CI	Upper CI
40 - 60	-3.724	0.009	-6.663	-0.786	-5.081	0.006	-8.917	-1.245	-6.283	0.000	-9.928	-2.637
40 - 80	-6.175	0.000	-9.114	-3.236	-8.278	0.000	-12.114	-4.442	-13.242	0.000	-16.887	-9.597
60 - 40	3.724	0.009	0.786	6.663	5.081	0.006	1.245	8.917	6.283	0.000	2.637	9.928
60 - 80	-2.451	0.121	-5.389	0.488	-3.196	0.121	-7.032	0.640	-6.960	0.000	-10.605	-3.314
80 - 40	6.175	0.000	3.236	9.114	8.278	0.000	4.442	12.114	13.242	0.000	9.597	16.887
80 - 60	2.451	0.121	-0.488	5.389	3.196	0.121	-0.640	7.032	6.960	0.000	3.314	10.605

Table 14: Results of Tukey Test for performance differences in delivery success rate through repositioning across three orders compositions (40%, 60%, and 80% urgent orders).

	20 Drones				**24 Drones**				**28 Drones**			
Comparison	Statistic	p-value	Lower CI	Upper CI	Statistic	p-value	Lower CI	Upper CI	Statistic	p-value	Lower CI	Upper CI
40 - 60	-2.478	0.107	-5.361	0.405	-3.985	0.028	-7.611	-0.358	-4.100	0.019	-7.628	-0.571
40 - 80	-4.449	0.001	-7.331	-1.566	-6.611	0.000	-10.238	-2.984	-9.419	0.000	-12.948	-5.890
60 - 40	2.478	0.107	-0.405	5.361	3.985	0.028	0.358	7.611	4.100	0.019	0.571	7.628
60 - 80	-1.971	0.239	-4.854	0.912	-2.627	0.201	-6.253	1.000	-5.319	0.002	-8.848	-1.790
80 - 40	4.449	0.001	1.566	7.332	6.611	0.000	2.984	10.238	9.419	0.000	5.890	12.948
80 - 60	1.971	0.239	-0.912	4.854	2.627	0.201	-1.000	6.253	5.319	0.002	1.790	8.848

E Experiment B: Additional Graphs

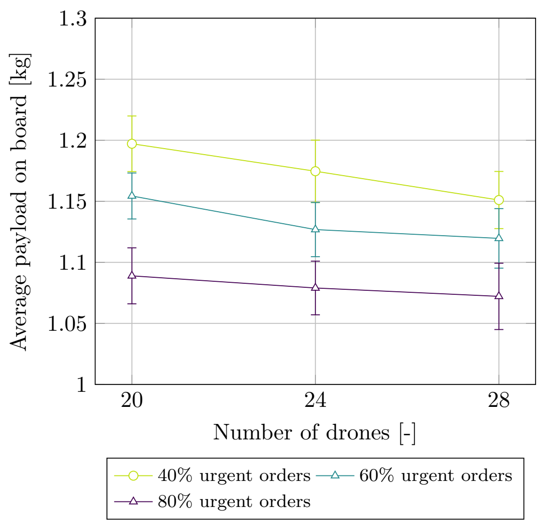


Figure 22: Average payload on board for different order compositions and fleet sizes.

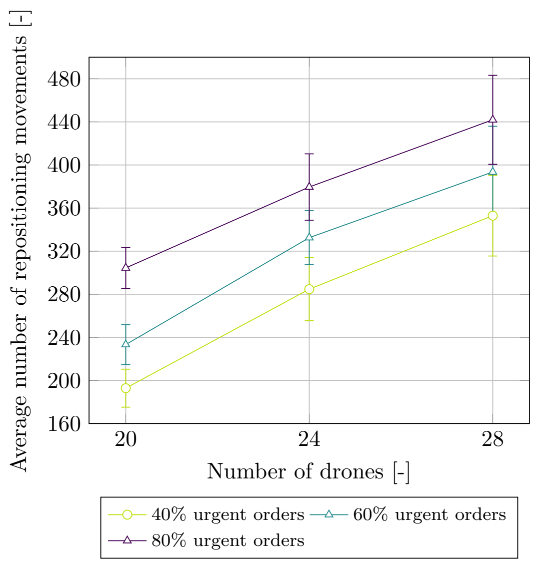


Figure 23: Average payload on board for different order compositions and fleet sizes.

F Experiment C: Statistical Data

Table 15: Results of Tukey Test for performances in cost per delivery of different fleet compositions (fleets 1, 2, 3, 4).

	18 Drones				**22 Drones**				**26 Drones**			
Comparison	Statistic	p-value	Lower CI	Upper CI	Statistic	p-value	Lower CI	Upper CI	Statistic	p-value	Lower CI	Upper CI
1 - 2	0.254	0.854	-0.572	1.079	1.772	0.000	0.965	2.579	1.632	0.000	0.809	2.455
1 - 3	1.280	0.001	0.454	2.105	2.076	0.000	1.269	2.884	1.989	0.000	1.165	2.812
1 - 4	1.635	0.000	0.809	2.460	2.821	0.000	2.014	3.629	3.291	0.000	2.468	4.114
2 - 1	-0.254	0.854	-1.079	0.572	-1.772	0.000	2.014	3.629	-1.632	0.000	-2.455	-0.809
2 - 3	1.026	0.008	0.200	1.852	0.305	0.759	-0.503	1.112	0.357	0.672	-0.466	1.180
2 - 4	1.381	0.000	0.555	2.207	1.049	0.005	0.242	1.857	1.659	0.000	0.836	2.482
3 - 1	-1.280	0.001	-2.105	-0.454	-2.076	0.000	-2.884	-1.269	-1.989	0.000	-2.812	-1.165
3 - 2	-1.026	0.008	-1.852	-0.200	-0.305	0.759	-1.112	0.503	-0.357	0.672	-1.180	0.466
3 - 4	0.355	0.677	-0.471	1.181	0.745	0.082	-0.062	1.552	1.302	0.000	0.479	2.125
4 - 1	-1.635	0.000	-2.460	-0.809	-2.821	0.000	-3.629	-2.014	-3.291	0.000	-4.114	-2.468
4 - 2	-1.381	0.000	-2.207	-0.555	-1.049	0.005	-1.857	-0.242	-1.659	0.000	-2.482	-0.836
4 - 4	-0.355	0.677	-1.181	0.471	-0.745	0.082	-1.552	0.062	-1.302	0.000	-2.125	-0.479

Table 16: Results of Tukey Test for performances in fleet utility of different fleet compositions (fleets 1, 2, 3, 4).

	18 Drones				**22 Drones**				**26 Drones**			
Comparison	Statistic	p-value	Lower CI	Upper CI	Statistic	p-value	Lower CI	Upper CI	Statistic	p-value	Lower CI	Upper CI
1 - 2	0.018	0.000	0.011	0.025	0.007	0.037	0.000	0.013	-0.001	0.973	-0.008	0.006
1 - 3	0.027	0.000	0.020	0.034	0.016	0.000	0.010	0.023	0.004	0.429	-0.003	0.011
1 - 4	0.043	0.000	0.036	0.050	0.031	0.000	0.024	0.037	0.007	0.071	-0.000	0.014
2 - 1	-0.018	0.000	-0.025	-0.011	-0.007	0.037	-0.013	-0.000	0.001	0.973	-0.006	0.008
2 - 3	0.009	0.008	0.002	0.016	0.010	0.001	0.003	0.016	0.005	0.213	-0.002	0.012
2 - 4	0.025	0.000	0.018	0.032	0.024	0.000	0.018	0.031	0.008	0.023	0.001	0.015
3 - 1	-0.027	0.000	-0.034	-0.020	-0.016	0.000	-0.023	-0.010	-0.004	0.429	-0.011	0.003
3 - 2	-0.009	0.008	-0.016	-0.002	-0.010	0.001	-0.016	-0.003	-0.005	0.213	-0.012	0.002
3 - 4	0.016	0.000	0.009	0.023	0.015	0.000	0.008	0.021	0.003	0.780	-0.004	0.010
4 - 1	-0.043	0.000	-0.050	-0.036	-0.031	0.000	-0.037	-0.024	-0.007	0.071	-0.014	0.000
4 - 2	-0.025	0.000	-0.032	-0.018	-0.024	0.000	-0.031	-0.018	-0.008	0.023	-0.015	-0.001
4 - 4	-0.016	0.000	-0.023	-0.009	-0.015	0.000	-0.021	-0.008	-0.003	0.780	-0.010	0.004

Table 17: Results of Tukey Test for performances in delivery success rate of different fleet compositions (fleets 1, 2, 3, 4).

	18 Drones				**22 Drones**			
Comparison	Statistic	p-value	Lower CI	Upper CI	Statistic	p-value	Lower CI	Upper CI
1 - 2	0.028	0.000	0.019	0.038	0.008	0.166	-0.002	0.019
1 - 3	0.031	0.000	0.022	0.041	0.025	0.000	0.015	0.035
1 - 4	0.051	0.000	0.042	0.061	0.052	0.000	0.041	0.062
2 - 1	-0.028	0.000	-0.038	-0.019	-0.008	0.166	-0.019	0.002
2 - 3	0.003	0.813	-0.006	0.012	0.017	0.000	0.006	0.027
2 - 4	0.023	0.000	0.014	0.032	0.043	0.000	0.033	0.054
3 - 1	-0.031	0.000	-0.041	-0.022	-0.025	0.000	-0.036	-0.015
3 - 2	-0.003	0.813	-0.012	0.006	-0.017	0.000	-0.027	-0.006
3 - 4	0.020	0.000	0.011	0.029	0.026	0.000	0.016	0.037
4 - 1	-0.051	0.000	-0.061	-0.042	-0.052	0.000	-0.062	-0.041
4 - 2	-0.023	0.000	-0.032	-0.014	-0.043	0.000	-0.054	-0.033
4 - 4	-0.020	0.000	-0.029	-0.011	-0.026	0.000	-0.037	-0.016

G Experiment C: Additional Graphs

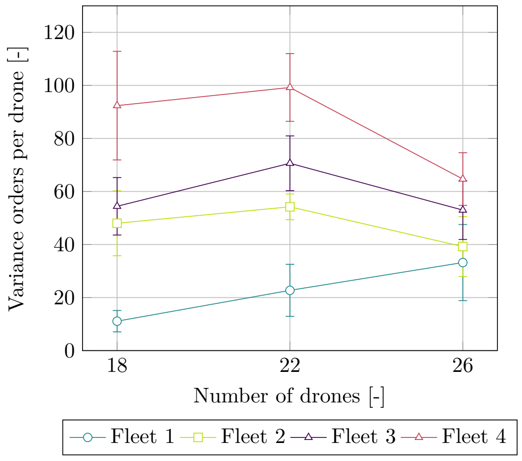


Figure 24: Variance of orders per drone across four different fleet compositions for various fleet sizes.

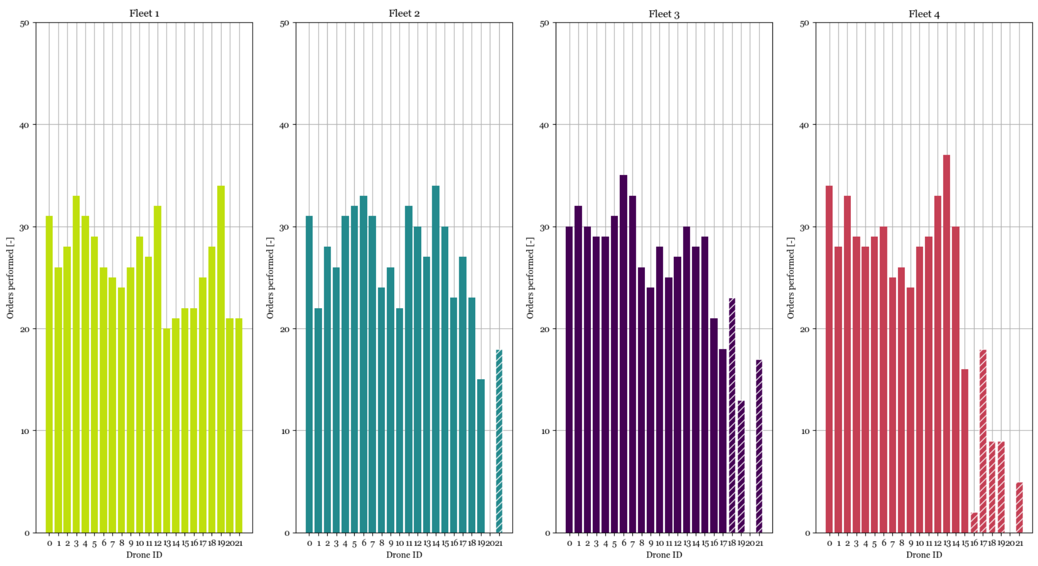


Figure 25: Spread of orders for a sample of simulations with varying fleet compositions (fleet size 22)

II

Literature Study

previously graded under AE4020

1 Introduction

In today's fast-paced and interconnected world, the need for on-demand services and fast delivery is becoming increasingly large. The emergence of this phenomenon places significant pressure on the domain of logistics and transportation. For a while now, delivery using unmanned aerial vehicles (UAVs) has been suggested as a means to cope with this increased pressure. The benefits of employing UAVs are compelling; they can avoid congestion, have the ability reach areas that are hard to access, are cost-efficient, and have a wide range of potential applications. Such applications include transport and logistics, agriculture, but also military operations. In the medical domain too, the use of UAVs is gaining interest. For example, they are of large value when disaster strikes, by being able to deliver medical supplies to inaccessible areas. In rural regions too, where healthcare is scarce and distances are hard to cover due to bad infrastructure, the use of drones is extremely beneficial. A company that is already active within this field is Zipline. Zipline started out on the continent of Africa in Rwanda and is now the national drone service provider in this country with the aim to perform over 2 million drone by 2029. Furthermore, the company has expanded to, amongst others, Ghana, Côte d'Ivoire, the United States, and Japan, showing the enormous potential of delivery using UAVs.

In the Netherlands, not the absence of good infrastructure, but the increased levels of congestion and need for cost efficiency are instigating the need for medical delivery using UAVs. This year alone, the number of traffic jams has already increased by 10% compared to pre COVID-19 times. As a result, urgent deliveries of blood are at risk of being delayed too much. Furthermore, resource shortages in terms of personnel and budget call for increased centralisation of healthcare. Therefore, a solution that would enable such centralisation and is cost-efficient is necessary. Medical Drone Service (MDS) is a company that is initiated to fulfill this need. The company's ambition is to contribute to the accessibility and availability of healthcare in the Netherlands through innovative mobility solutions. To achieve this, MDS is setting up a nation-wide network of UAVs to ship medical material between various types of partners. Various questions arise when designing such a network. Because how should the network cope with such stochastic demand and strict temporal requirements? How should resources and delivery tasks be allocated to achieve maximal availability at minimum cost?

This literature study aims to provide an answer to the previously posed questions, as well as a description of what the MDS network looks like. The first two chapters aim to provide some context on Medical Drone Service. First, chapter 2 elaborately introduces Medical Drone Service. Next, chapter 3 further introduces the main carrier of the network, the UAV. This chapter also provides other companies that utilise UAVs for their services. Then, to understand the research gap, chapter 4 dives into research on UAV delivery problems and general dynamic pickup and delivery problems. The next two chapters look into techniques that could be implemented in a model for the MDS network. chapter 5 looks into allocation and scheduling methods and provides a trade-off. Subsequently, chapter 6 documents and chooses suitable repositioning methods. Finally, the proposed research is outlined in chapter 7

2 Medical Drone Service

In this chapter, Medical Drone Service, the company for which a UAV network will be developed, is introduced. First, section 2.1 introduces the ambition of the company. Then, section 2.2 shortly describes the project history. Next, section 2.3 sheds light on why such a service is needed in the Netherlands. The concept of operations is described in detail in section 2.4. Lastly, resulting from the previous section a list of requirements is constructed in section 2.5.

2.1. Ambition of Medical Drone Service

Medical Drone Service is an initiative from ANWB Medical Air Assistance (MAA) and PostNL Health (for both logos are displayed in Figure 2.1 and Figure 2.2). ANWB MAA is the Dutch medical air transportation company that operates trauma helicopters to transport Mobile Medical Teams. Besides, ANWB MAA operates an ambulance helicopter to transport patients from the West Frisian Islands. PostNL is a logistic service provider in the Netherlands and is also active in healthcare. For instance, the company delivers medical posts for diagnostic centres and supplies medication at hospitals, pharmacies and to patients at home [73].

Together with several other medical partners, MDS researches how drones could contribute to delivering healthcare at the right place and time. MDS's ambition is to contribute to the accessibility and availability of healthcare in the Netherlands through innovative mobility solutions. The employment of drones should bring healthcare faster and closer to the patient while strengthening the connection between medical institutions in the Netherlands. The ultimate goal is to set up a nationwide network of medical drones to increase medical transport's sustainability and delivery rates in the Netherlands [74, 75].

Figure 2.1: PostNL logo.

Figure 2.2: ANWB logo.

2.2. Project History

In 2019, MDS, a consortium of ANWB MAA, PostNL, Erasmus MC, Sanquin, and technology partners KPN and Avy, first reached the news for their initiative. This news came at the start of a three-year pilot project. In preparation for this project, ANWB MAA already bought the first UAV in 2018. The pilot project was split up into different phases. The first step of the intended pilot project was to fly the drone above a protected test location within a maximum radius of 4 kilometres. These test flights included Beyond Visual Line of Sight (BVLOS) operations. The next step was to test flights from point to point above sparsely populated areas. The last step was to fly from medical location to medical location, ultimately with a payload of medical nature [27].

At the end of 2020, ANWB announced the first test flights between the hospital locations of Isala, Meppel and Zwolle. These locations are approximately 15 kilometres apart, and most of the route contains flying over rural areas. However, instead of directly landing at Isala Zwolle, the medical drone landed on a meadow north of Zwolle to avoid flying over densely populated areas. During the test flights, no medical payload was carried along. Figure 2.3 shows a photo which was taken during one of the test flights. The test flights were carried out until mid-2021 [3].

Test flights also started between Rhoon and Oud-Beijerland in May 2021. Erasmus MC and Sanquin, a dutch hospital group and blood bank, commissioned these flights. These test flights investigated which technical adjustments are necessary to fly the intended routes [67].

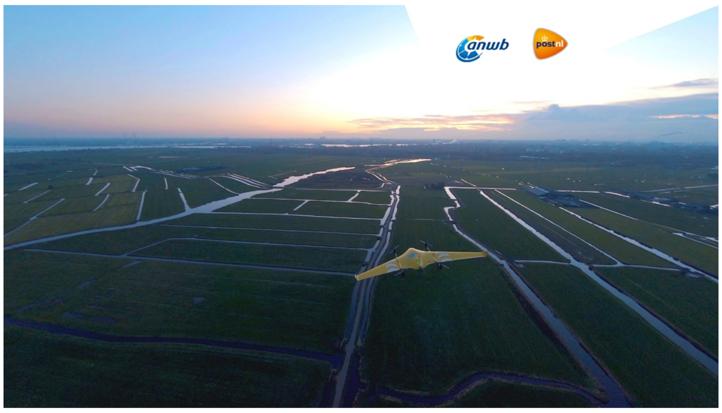


Figure 2.3: A photo that was taken during a test flight [71].

In 2022, it was announced that the first medical drone flights would start in 2023. Medical Drone Service could build a medical airlift between Meppel and Zwolle with all the information gathered from the test flights. Although the flight is autonomous, an operator located in the Hague will monitor the flight [14].

Moreover, MDS collaborated with the European project AMU-LED that same year to perform test flights within the Controlled Tower Region (CTR) of Rotterdam The Hague Airport. This project is an essential step towards improving regulations for unmanned flight in Europe. The test flight shows how manned and unmanned flights can simultaneously operate in the same airspace. From the test flights it was concluded that current communication procedures result in a high workload for the drone pilot [72].

Now, the goal is to start medical flights between the two Isala locations. Furthermore, the aim to also start carrying out medical flights at other locations, for instance, near Rotterdam, which provides another operational scenario [76]. For the future, there will be technological developments to make flying above densely populated areas possible and MDS's network will expand to a nation-wide network.

2.3. Service Need

In the first part of this section, the situation and challenges of the Dutch healthcare sector are sketched. Afterwards, these challenges are concretised into issues explicitly relating to MDS, and the service's added benefit to the sector is shown.

At the end of 2022, the Dutch Healthcare Authority (NZa) published an article highlighting trends in Dutch healthcare. The authority concluded that the demand for care is increasing in every healthcare sector due to the effects of an ageing population. However, the labour market for healthcare is decreasing, resulting in personnel shortages. Furthermore, the increased demand puts pressure on financial resources. The risk is that long waiting lists will develop when too few resources are available, leading to longer waiting times. This phenomenon could further increase social segregation due to social and geographical inequalities. Management should deploy resources more strategically to cope with this scarcity of financial resources and personnel. This strategic deployment requires revising current operational procedures and intensive cooperation to make centralisation possible [1].

As stated earlier, centralisation could partially solve personnel and resource shortages. The minister of the Dutch Ministry of Health has mentioned areas of improvement regarding this subject. For instance, not all hospitals should offer fully equipped emergency departments. These fully equipped emergency departments could be centralised to regional hospitals with the appropriate resources and expertise. Local hospitals can still execute basic acute care; however, complex acute care will be referred to regional hospitals. This example of centralising a specialisation, such as a fully equipped emergency department, can happen for all specialisations, each in a different local hospital. Therefore, intensive collaboration on a regional level is necessary. Consequently, transport between hospitals will increase due to an increased need for exchanging medical materials. Furthermore, resulting from this increased centralisation, there is a greater need for digitisation and central databases to improve communication between medical institutions [42].

The National Institute for Public Health and Environment (RIVM) also focuses on the theme of 'sustainable care and prevention' to make healthcare accessible and affordable considering the current hazards. The institute focuses on innovation, organisation and funding, and prevention. Regarding the first topic, it is vital for the RIVM to make a trade-off between the costs and benefits of new technologies. These analyses take not only the final results but also operational characteristics into consideration. Moreover, is there a need for these technologies, and do they replace current procedures or add to them? The RIVM commits to developing more cost-and-benefit analyses on this matter. Concerning the second topic, it comes down to which party does what and when? Making an effective division can decrease costs and workload, as elaborated in the previous paragraph [86].

As mentioned earlier, a vital issue in reducing personnel shortages is ensuring the correct entity takes the right action at the right time and place. Logistics make up a significant part of medical operations, even more, when centralisation is increasingly implemented. Centralisation also has the downside that the physical accessibility of medical specialisations decreases, which again increases the need for transport. To visualise the need for transport in the medical sector, on a yearly basis, 3.5 million medical deliveries are made to patients at home and the same number is delivered to hospitals [30]. An example is Erasmus MC, a Dutch hospital that delivers medication to approximately 10.000 patients at home [27]. Moreover, traffic congestion hinders most of these deliveries, which could be vital in the case of emergency deliveries. ANWB Traffic Information even concluded that road congestion has increased by 10 % for the first quarter of 2023 compared to 2019 (pre-COVID era) [4].

This is where MDS comes in. The service provides a solution that takes the workload off of medical personnel and is cost-effective and time-efficient. By outsourcing medical deliveries, hospital personnel can focus more on their core tasks. Furthermore, the service can be cost-effective because it can collaborate with multiple medical institutions, centralising resources for medical deliveries. This centralisation leads to further cost reductions. Until now, hospitals have been used as the only example for medical deliveries. Plenty of other medical institutions, such as pharmacies, blood banks, and laboratories, also perform medical deliveries, creating an enormous market for MDS. The deployment of drones also avoids a big issue in transportation: traffic congestion. When deliveries cost less time, available resources increase and costs decrease. Therefore MDS could fill a gap in the current Dutch healthcare environment. Besides, the service exploits innovation and has the potential to reduce environmental impact through the use of UAVs making it, which adds to its attractiveness.

2.4. Concept of Operations

This section describes the entire concept of operations of MDS. First, subsection 2.4.1 describes the characteristics of the UAV used for operations. Then, subsection 2.4.2 what type of orders the network will have to process. Next, subsection 2.4.3 describes the operational process when an order is received. Finally, subsection 2.4.4 elaborates on the backup network in place.

2.4.1. UAV Characteristics

MDS uses the Avy Aera 3 for their operations, depicted in Figure 2.4. The aircraft has a wingspan of 2.4 meters and is a hybrid UAV that can perform VTOL with a fixed-wing design. The UAV can autonomously fly up to 100 kilometres at 90 kilometres per hour. There will be an operator located on the ground that monitors the flight. The UAV can hold a payload of up to 3 kilograms, and Avy provides the Avy Medkit for medical cargo [9, 30]. Avy developed this kit together with Sanquin, a Dutch blood bank. The kit can maintain stable temperatures up to 40 degrees Celsius for 100 minutes. Moreover, the kit contains sensors that allow the user to monitor the cargo's state during flight. The operational window of the UAV is large; it can fly with winds up to 25 knots and rain up to 3 millimetres per hour [7]. The latter is comparable to regular shower that is common in the Netherlands.

Figure 2.4: Render of Avy Aera 3 [6].

Figure 2.5: Render of Avy docking station [8].

Every time the UAV arrives at a new destination, personnel that receives the UAV will swap the battery. Hence, each flight starts with full battery capacity. Furthermore, the drones can stay on the landing pad for approximately two hours after arrival. After that point, drones should return to their docking stations to maximise their lifespan and prevent unnecessary maintenance. This docking station is also provided by Avy [8]. It has the advantage of being weatherproof and remotely controlled. The station is shown in Figure 2.5. These docking stations will not be located at every location that MDS serves but at clients whose location frequently acts as a pickup destination.

2.4.2. Order Specifications

The orders that MDS receives are always pickup and delivery orders. These can have a certain level of urgency attached to them. MDS differentiates between three levels of urgency. The first level is regular transport; this has to happen on a given day but is not bound to a specific time. The second level is urgent transport. This type of transport usually has a deadline within 2 to 4 hours from order arrival. The third level is very urgent transport. These orders must arrive at their final destination within approximately 1 hour of order arrival.

The MDS network is going to ship three categories of products: blood, medication, and diagnostic materials. Each of these products has specific temperature requirements. Therefore it is rarely possible to combine orders. Orders can originate from clients such as hospitals, laboratories, blood banks, large retailers, and general practitioners.

2.4.3. Operational Process

The process surrounding an order looks like the following. A client generates an order, and the MDS operations centre receives the order. This centre is active round-the-clock and has a complete overview of all operations. A drone is assigned and scheduled to perform the order. Before the UAV departs, the operator inspects the weather conditions to ensure safe operations. The order is prepared just before the UAV lands at the pickup location. As soon as the UAV lands at the pickup location, the order is loaded onto the UAV and the UAV's battery is swapped. It is assumed that these actions do not cause much delay. When the UAV departs for its delivery destination, weather conditions are inspected again. Once the UAV arrives, it is unloaded, and its battery is swapped again. If no docking station is present at the delivery location, the UAV can either stay put for a maximum of two hours or return to its docking station.

2.4.4. Backup Network

As mentioned earlier, the UAV has a specific operational window. When winds or rain are too heavy, the drone cannot fly. Moreover, it is not possible to fly everywhere due to airspace restrictions. For instance, a delivery location could be located in a region where drone flight is (temporarily) prohibited. Therefore, it is necessary to have a backup network in place.

This backup network will consist of ground vehicles (either bikes, motorcycles or cars). Whenever it is impossible to carry out a delivery by air, the ground network carries out the order in a similar fashion as that described in subsection 2.4.3. Furthermore, the ground network could be deployed to prevent the fleet size from becoming unnecessarily large due to, for instance, overlapping orders.

2.5. Model Requirements

In order to obtain a cost-efficient network and to be able to make managerial decisions, a model must be created. This model has to allocate and schedule UAVs to tasks and simulate the execution of those tasks. The previously described setup and interviews with MDS and industry partners have led to an elaborate set of requirements. These requirements serve as the basis for decisions that are made in this document. Each of the requirements and an accompanying explanation are listed below.

- **Modular:** MDS is still in the piloting phase. Hence there is much room for growth and adjustments to operational processes. The chosen techniques must allow for the expansion, adjustment, and removal of variables.
- **Explainable:** The model is meant to provide information about future scenarios to decision-makers. Therefore, the model cannot act as a black box, and understanding the model's reasoning is essential.
- **Scalable and computationally efficient:** The ultimate goal of MDS is to create a nationwide network covering all sorts of medical partners. The model must be capable of handling large instances with over 100 clients and UAVs.
- **Centralised nature:** MDS centralises healthcare transport logistics. Hence, a model should reflect this characteristic.
- **Cooperative:** All agents are in service of MDS and, therefore, cooperative.
- **Suitable for a highly dynamic environment:** The arrival of part of the orders is a stochastic process. Therefore, implemented methods should be able to handle a dynamic environment.
- **Anticipating:** A vital part of task assignment in the described environment is considering orders that may arrive in the future. Implemented techniques should therefore possess anticipating features.
- **Handle tasks with heterogeneous deadlines:** As previously described, the orders that MDS receives have a certain urgency. Therefore, the model must be able to cope with this urgency.
- **Multiple tasks:** Although assigning pickup and delivery is the model's core. Tasks such as repositioning and scheduling are also included. Therefore, implemented techniques must be able to handle a diverse set of tasks.
- **Heterogeneous fleet:** As described earlier, next to UAVs, ground vehicles also ship orders. Hence the network must be able to handle multiple types of vehicles.

3 On UAV Deployment

This chapter introduces the primary vehicle of the Medical Drone Service (MDS) network, the drone. The research field of drones, or unmanned aerial vehicles (UAVs), is a very active field. This is proven by the abundance of recent literature on the topic that is available.

First, section 3.1 provides a description of UAVs. Then, section 3.2 highlights the strengths and weaknesses of UAVs. Next, the potential applications of UAVs are discussed in section 3.3. Lastly, section 3.4 zooms in on UAV delivery applications in practice.

3.1. Introduction to Unmanned Aerial Vehicles

The term 'UAV' denotes an aircraft without an on-board human pilot that is operated remotely. Other terms that are used to describe this type of aircraft are drone or remotely piloted aircraft (RPA). According to the International Civil Aviation Organization (ICAO), the latter reflects best that although there is no human on-board, it is still piloted remotely by human operators. To also include all of the associated elements with the unmanned aircraft, the terms unmanned aerial system (UAS) or remotely piloted aircraft system (RPAS) are used [57].

According to Hassanian and Abdelkefi [37], the aerial vehicle discussed in chapter 2 are categorized as UAVs. These vehicles fall within a weight range of 5 to 15,000 kilograms and can have a wingspan spanning from 2 to 61 meters. Specifically, the Avy Aera 3 belongs to this UAV class and is designed with a fixed-wing configuration. This configuration offers the advantage of reduced complexity, making repairs and maintenance easier. However, it also means that the vehicle is unable to hover, unlike other configurations like rotary-wing.

The current market for drones is exponentially growing. In the United States only, McKinsey & Company reports a rise in the worth of drone enterprises from 40 million dollars to 1 billion dollars from 2012 to 2017. The company estimates this value to grow to 31 to 46 billion dollars by 2026. Startups drive most UAS activity. Over 300 companies have entered the market since 2000, typically focusing on hardware, operations, or support devices such as navigation [24].

Furthermore, the European Union adopted the European Drone Strategy 2.0, stating that the drone services market could grow to 14.5 billion euros in 2030 with the proper framework. The European Union has already played a role in establishing an extensive drone regulatory framework that facilitated the sector's development. The strategy focuses on two main objectives: expanding the drone services market and enhancing the Union's civil and military industry capabilities and synergies [25].

3.2. Advantages and Limitations of UAV Deployment

The utilization of UAVs in various market applications has gained significant attention due to their potential for tailored solutions to modern-day challenges. This section aims to provide an extensive overview of the strengths and weaknesses associated with UAV deployment.

3.2.1. Strengths

One notable strength of drones is their high speed, which is attributed to both their configuration and the ability to travel directly [22, 29, 50]. By avoiding physical infrastructure, drones can take more direct routes, bypassing traffic congestion. This not only reduces travel time but also leads to more reliable arrival time estimations and lower energy consumption. Additionally, the absence of infrastructure dependencies enhances accessibility to areas that are difficult to reach due to geographical challenges or lack of established infrastructure. Collectively, these factors result in time and cost savings, leading to improved operational efficiency and increased resource availability for companies and clients [5, 10, 45, 64].

Furthermore, the environmental impact of UAVs is considered positive by several authors. Most drones are battery-powered, ensuring zero emissions during operation. Moreover, their implementation in logistics can reduce the number of delivery vans in urban environments, resulting in decreased emissions, noise pollution, and traffic congestion [15, 31, 33, 45].

In addition, the modular nature of UAVs contributes to their high flexibility in application. Drones can be customized with varying characteristics such as range, payload capacity, and configuration, enabling tailored solutions for diverse challenges [45, 64]. For instance, during the COVID-19 pandemic, UAVs facilitated contactless delivery, mitigating the spread of the virus [5, 10]. Banik et al. [11] developed a decision support model for selecting the optimal drone for different applications. Further exploration of the wide array of UAV applications can be found in section 3.3.

3.2.2. Weaknesses

However, the use of drones also has a few weaknesses. Many authors highlight the existing regulatory framework (or lack thereof) as a hindrance to the commercial application of UAVs [10, 11, 31, 62]. Additionally, liability issues pose significant challenges. The current guidelines lack specificity regarding coverage policies for personal and commercial liability, personal injury, and privacy infringement.

Moreover, public perception and acceptance play crucial roles. Privacy invasion and personal injury are commonly perceived as major concerns by the public [62]. Privacy-related issues include concerns about spying and data collection [10, 22, 64]. There is also a risk of valuable information being compromised by malicious attackers. Regarding personal injury, technological failures leading to drone crashes, hacking through jamming or spoofing, collisions with objects or other aircraft, and unauthorized intrusion into restricted airspace (e.g., airports) can all potentially cause harm. Technological advancements have room for further development in addressing these concerns.

Another weakness of drone deployment is the initial cost. While many authors argue that drone-based solutions can be cost-efficient, reaching a certain level of utilization is necessary. In the case of delivery operations, McKinsey & Company suggests that an operator would need to control as many as 20 drones for them to become cost-competitive [26]. In the medical domain, it has become apparent that the frequency of system utilization must outweigh the capital costs involved [10, 55].

Lastly, UAVs still have limited capabilities compared to traditional trucks. They are constrained by payload capacity, have restricted range and battery life, and face limitations imposed by natural conditions such as weather and geography [10, 22, 45, 58].

3.3. Possible UAV Applications

As mentioned in section 3.2, the deployment of UAVs offers broad applicability across multiple sectors. Macrina et al. [47] categorize the main application areas as civil, environmental, and defence. Within the civil sector, Otto et al. [58] have identified several promising applications, including physical infrastructure, agriculture, and transport.

In the realm of physical infrastructure, operators utilize drones for terrain examination, monitoring construction site progress, and inspecting facilities for maintenance purposes. These activities can be carried out at a relatively low cost while minimizing risks for employees.

In agriculture, UAVs play a crucial role in tasks such as fertilizer spraying, crop health assessment, and soil property mapping. The agricultural sector, especially in the United States, anticipates significant growth in the adoption of drone technology.

Transportation applications of UAVs primarily focus on delivery services, with emphasis on first- and last-mile delivery, as these segments are typically labor-intensive. Developing countries also show considerable interest in utilizing drones for the delivery of medical supplies, addressing infrastructure limitations and enhancing reliability. Other notable applications include surveillance, disaster management, entertainment and media, as well as telecommunications.

Furthermore, UAVs find utility in environmental endeavors. They are employed for tasks such as monitoring air quality, managing national parks, and surveying ecosystems. These applications contribute to environmental conservation efforts and enable more effective ecosystem management.

Lastly, the field of defense has been the traditional domain for drone technology. UAVs serve various purposes in this sector, including combat operations, surveillance, intelligence gathering, and supply logistics.

The versatility of UAVs allows for their integration into a wide range of sectors, offering novel solutions and capabilities across diverse industries.

3.4. UAV Deployment Examples in Medical Applications

In recent years, the use of UAVs, in the medical domain has gained significant attention due to its potential to revolutionise healthcare delivery. Aside from MDS, there are several other notable projects and partnerships in this domain that showcase the diverse applications and benefits of drone technology.

3.4.1. Matternet

One prominent example is the collaboration between Swiss Post and Matternet, which produces an end-to-end drone-based solution, to transport lab samples between hospitals and laboratories. They conducted trials in Zürich, Bern, and Lugano from 2017 to 2022. These trials demonstrated tangible benefits, including a clear reduction in delivery time. However, it became apparent that the operational costs of drone-based deliveries were high, making profitability in the medium term challenging. As a result, Swiss Post decided to hand over the project to Matternet, who will continue operations in Lugano this year [60].

Figure 3.1: Photo of operations of Swiss Post above Lugano, retrieved from ...

Apart from the Swiss Post partnership, Matternet has partnered with many other organisations in the medical domain. For example, Matternet worked with UPS to deliver COVID-19 vaccines to a North Carolina hospital group in 2021 [79]. This collaboration demonstrated the potential for drones to contribute to emergency healthcare responses, especially during critical situations such as a pandemic. Furthermore, the Department of Health of Abu Dhabi has partnered with Matternet and SkyGo to transfer medical supplies within the healthcare sector. This system will work around the clock to distribute medical supplies, medicine and blood units, vaccines and samples between laboratories, pharmacies and blood banks [56]. Lastly, Matternet also started a collaboration in Germany with Labor Berlin. This organisation is responsible for diagnostic services for a large part of patient beds in Berlin. Trials have started by connecting three out of thirteen locations. During trials, Matternet wants to explore how drone delivery can be integrated best into the existing workflow [48].

3.4.2. Delft University of Technology

Another example of using UAVs in the medical domain is the Ambulance Drone designed by Alec Momont from Delft University of Technology. This drone can carry medical equipment such as an Automated External Defibrillator, medication, and Cardiopulmonary Resuscitation aids. What sets this drone apart is its capability to fly indoors, making it invaluable in emergency situations where quick response times are crucial. The Ambulance Drone has the potential to provide critical support in the immediate aftermath of accidents or medical emergencies, enhancing the effectiveness of first responders and potentially saving lives. TU Delft, in collaboration with relevant stakeholders, is currently considering further steps for the development and implementation of this groundbreaking technology. [49].

3.4.3. Zipline

Zipline, a pioneering company in drone-based medical deliveries, has also made significant strides in this field. Zipline started delivering blood and vaccines in Rwanda in 2016. Rwanda is known for its many hills making medical facilities hard to reach. Therefore a drone-based solution could have a significant impact. Several studies have now shown that this is indeed true. A study published by the Lancet shows that the average delivery time was reduced by 79 minutes, reducing blood wastage by 67 per cent [53]. Moreover, the Bill & Melinda Gates Foundation found that Zipline contributes to increased health access and equity.

Encouraged by its success in Rwanda, Zipline expanded its operations to several other countries, including the United States, Nigeria, Ghana, Kenya, Côte D'Ivoire, and Japan. With over 540,000 deliveries completed, Zipline has emerged as a frontrunner in the field, contributing to increased health access and equity. Building on its accomplishments in the medical domain, Zipline has diversified its services and ventured into other sectors such as E-commerce and agriculture. This expansion demonstrates the versatility of drone technology and its potential to revolutionize various industries beyond healthcare [88].

3.4.4. Conclusion

In conclusion, the medical domain has witnessed notable advancements in the application of UAVs, or drones, for a range of purposes. These projects, partnerships, and initiatives, such as those involving Matternet and Zipline, exemplify the potential of drone technology in healthcare. From transporting lab samples and medical supplies to facilitating emergency responses and improving access to essential medical resources, drones are proving to be a promising tool for enhancing efficiency, reducing delivery times, and ultimately transforming the way healthcare services are delivered.

However, less is known about the scalability and cost-effectiveness of drone-based medical delivery systems. Although the mentioned projects have demonstrated the feasibility and benefits of using drones for transporting medical materials, there is still a need for further research on how to scale up these operations and make them financially viable in the long term. Research could explore strategies to optimize operational costs, overcome regulatory challenges, and design efficient logistical networks to support widespread implementation of drone-based medical delivery systems.

4 Related Research

This chapter presents an overview of the UAV-related problems that have been studied, aiming to identify any gaps in the current research. Specifically, the first section (section 4.1) focuses on routing problems related to UAVs. Subsequently, the following section (section 4.2) provides further insights into dynamic pickup and delivery problems.

4.1. UAV-based Literature

This section provides a detailed examination of the literature related to UAV-based research. Firstly, subsection 4.1.1 categorizes the literature into different problem categories. Subsequently, subsection 4.1.2 explores the drone delivery problem, while subsection 4.1.3 focuses on the parallel drone scheduling TSP and VRP.

4.1.1. Classification of UAV-based Delivery Models

Macrina et al. [47] proposed a classification scheme for literature on drones in delivery, shown in Figure 4.1. They identified four main categories. Additionally, they introduced two overarching macro classes that encompass these categories.

The first macro class includes scenarios where both trucks and drones are involved in deliveries. Within this class, two distinct categories exist: TSP with drones and VRP with drones. The differentiation between these categories lies in the number of trucks considered in the problem. In TSP with drones, there is a single truck accompanied by one or more drones, and their objective is to find an optimal tour. On the other hand, VRP with drones involves multiple trucks, each aiming to find a tour that covers all locations with the assistance of several drones. Furthermore, both categories can be further classified based on the nature of truck-drone operations, specifically as parallel or synchronous operations. In the parallel operations category, the truck and drone operate independently of each other, while in the synchronous operations category, their actions are interdependent.

The second macro class is characterized by scenarios where only drones are responsible for deliveries. This class encompasses two categories: the drone delivery problem (DDP) and the carrier-vehicle with drones problem. The drone delivery problem represents a drone-only variant of the VRP, where all customer requests must be fulfilled exclusively by drones. The carrier-vehicle with drones problem explores the research area where drones can hitch a ride on larger vehicles to conserve battery power.

Although all of these categories describe relevant problems in the context of drone deliveries, the MDS network is primarily concerned with studies related to the drone delivery problem, as well as the TSP and VRP problems where drone operations are parallel.

4.1.2. The Drone Delivery Problem

As stated, the drone delivery problem is a drones-only variant of the VRP. Table 4.1 shows a summary of the reviewed papers on the DDP. In order to relate the studies in this table to the MDS, it is important to remember the MDS problem characteristics. The most notable characteristics are pickup and delivery tasks; deadlines; multiple-region coverage; absence of a central depot; and highly stochastic requests.

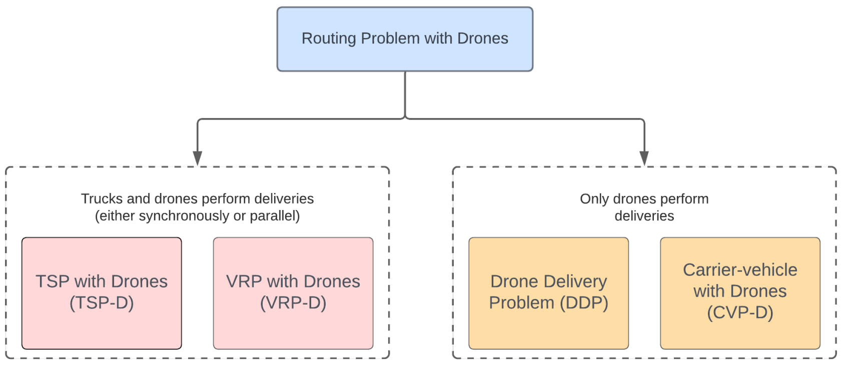

Routing Problem with Drones
Trucks and drones perform deliveries (either synchronously or parallel)
TSP with Drones (TSP-D)
VRP with Drones (VRP-D)
Only drones perform deliveries
Drone Delivery Problem (DDP)
Carrier-vehicle with Drones (CVP-D)

Figure 4.1: Classification of UAV delivery models as classifed in [47].

In Table 4.1, there is no paper present that relates closely to the previously described concept. The table shows that only two out of eight papers discuss pickup and delivery operations. Furthermore, three out of eight papers discuss a dynamic network. For these dynamic networks task allocation and scheduling is performed using optimisation alogrithms. Only Sawadsitang et al. [68] discuss a network that handles deliveries with time windows without a central depot present. However, their approach only allows UAVs to carry one package at a time, and their formulation does not consider the dynamic nature of the environment.

Most of the presented papers utilise heuristics, simulated annealing or a commercial solver to solve their formulation of the problem. Additionally, a large part of the objectives is cost related in some manner. Only the papers specifically addressing meal delivery focus on minimizing delivery time. Furthermore, all papers discussed are on a regional scale and some of them mention the expansion of networks to utilise inter-region synergies. The following few paragraphs briefly summarize the papers presented, organising them based on their respective concept of operations.

Liu [46] and Huang et al. [40] both develop a UAV network for meal delivery. Their primary objective is to minimize delivery time, which they refer to as "tardiness" or "lateness". Liu additionally incorporates a hierarchical objective that prioritizes safety, followed by minimizing lateness, and maximizing efficiency. To address the dynamic nature of the environment, Liu employs a rolling horizon approach and solves an MIP formulation. Huang et al., on the other hand, propose a Stochastic Event Scheduling framework. They cluster tasks and solve the scheduling problem for each cluster, allocating the resulting schedules to the appropriate UAVs. The framework also incorporates rescheduling of unexecuted tasks to adapt to the dynamic environment. Both papers address the challenge of a dynamic, infinite-horizon Vehicle Routing Problem (VRP) with en-route vehicle diversion, which closely aligns with the challenge posed by MDS.

Campuzano et al. [17] and Sawadsitang et al. [68] explore the concept of a UAV network with the possibility of outsourcing deliveries to ground vehicles at a higher cost. Campuzano et al. discuss a dynamic network with a central depot where UAVs can make round-trip flights to customers. At each time step, the coordinator must decide which drones need to be charged and which ones should perform deliveries. If no UAV is available for a particular delivery, the delivery is outsourced to a ground carrier, incurring a high cost. The objective is to minimize the number of occurrences of outsourcing events. Campuzano et al. employ approximate value iteration, a reinforcement learning method. Sawadsitang et al. adopt a similar setup, but assign individual depots to each drone. In this case, outsourcing is necessary due to uncertainties like adverse weather conditions. They solve an MILP formulation using a commercial solver. Both papers consider time windows for deliveries. An interesting aspect of these studies is their approach to incorporating backup vehicles, which is also a key element in the MDS network.

Dorling et al. [32], Troudi et al. [80] and Gómez-Lagos et al. [35] individually address solving UAV routing problems in the context of delivery applications. Dorling et al. solve a VRP, where a UAV can visit a depot multiple times. Their energy consumption model highlights the optimisation of battery weight. Gómez-Lagos et al. incorporate pickup and delivery operations by sending a UAV to pick up an order at a facility before delivering it to the customer. They explore various mathematical formulations to achieve this. Troudi et al. tackle a capacitated VRP with time windows, determining fleet size based on delivery forecasts. Although each of the concepts has distinctive features, they are not closely related to the MDS project.

Furthermore, Coelho et al. [23] propose a microgrid system specifically designed for UAV delivery. This

Table 4.1: Summary of characteristics of papers on the drone delivery problems.

	Problem Formulation Solved	**Time Windows**	**Central Depot**	**Method**	**Objectives**	**Other Characteristics**	**Application**
Dorling et al. [32]	Multi Trip VRP	✗	✓	Simulated annealing	Cost Delivery time	· Energy model · Multiple parcels · Multi-trip	Package delivery
Coelho et al. [23]	Green Dynamic VRP	✗	✗	Matheuristic	Multi-objective using 7 indicators	· Realistic operational constraints · Package exchange · Microgrid system · Heterogeneous fleet	Package delivery
Troudi et al. [80]	Capacitated VRP with Time Windows	✓	✓	Commercial Solver	Distance n UAVs n batteries	· Energy model · Multiple parcels · Multi-trip	Fleet dimensioning in delivery
Liu [46]	Dynamic PDP	✗	✗	MIP rolling horizon based heuristic	Lateness or hierarchical (safety, lateness, efficiency)	· Multiple orders · Restriction on combinations of orders · Battery swap · Multi-trip · Arbitrary pickup and delivery locations · Multiple couriers per order	Meal delivery
Huang et al. [40]	Dynamic Capacitated VRP	✗	✗	Hybrid of K-Means++ and simulated annealing	Total tardiness	· Task scheduling · Multiple orders · Battery swap · Multi-trip	Meal delivery
Campuzano et al. [17]	Dynamic Drone Scheduling Delivery Problem	✓	✓	Reinforcement Learning	Minimize number of used backup vehicles	· One unit load · Ground vehicle backup · Charge or deliver	Package delivery
Sawadsitang et al. [68]	Multi-objective Drone Delivery Problem	✓	✗	MILP with commercial solver	Multi-objective using 3 indicators	· One unit load · Ground vehicle backup · Realistic operational constraints	Package delivery
Gómez-Lagos et al. [35]	Pickup to Delivery Drone Routing Problem	✗	✓	Greedy Randomized Adaptive Search Procedure	Makespan	· Multiple orders · No mixing of orders · Battery swap · Facility inventory	Package delivery

system consists of multiple layers, with each layer accommodating a different type of drone. UAVs are allowed to transfer packages between different layers in order to consolidate deliveries. While this concept is very innovative, it is not applicable within the context of the MDS network.

4.1.3. Parallel drone scheduling TSP and VRP

In the parallel drone scheduling TSP and VRP, both UAVs and trucks are involved in carrying out deliveries simultaneously. These problems offer interesting opportunities for the MDS project because of the backup network that is in place.

An overview of studies on the parallel drone scheduling TSP and VRP is presented in Table 4.2. Most of these studies do not allow for more than one delivery per trip, considering capacity constraints. However, Hamid et al. [36] deviate from this approach by incorporating a detailed energy model of the UAV, enabling multiple deliveries per trip. Ulmer, Thomas, and Chen [20, 82] address the problem by considering stochastic requests and time windows. They employ solution methods based on reinforcement learning, which effectively leverage the advantages of using either UAVs or trucks in specific areas. These findings can be valuable for determining threshold travel times for transitioning from trucks to UAVs and dividing service regions into areas served by either UAVs or trucks. The authors also propose exploring re-assignment strategies to better handle stochastic requests. Finally, Hamid et al. [36] extend the problem by incorporating various other types of agents, such as motorcycles and crowd-sourced couriers, each with their own distinct set of constraints.

Table 4.2: Summary of characteristics of papers on parallel drone scheduling TSP and VRP

	TSP or VRP	**Stochastic requests**	**Time Windows**	**More than one delivery per UAV trip**	**Solution Method**	**Objective Function**
Saleu et al. [66]	VRP	✗	✗	✗	Hybrid metaheuristic	Min. completion time
Nguyen et al. [52]	VRP	✗	✗	✗	Heuristic	Min. cost
Ulmer and Thomas [82]	VRP	✓	✓	✗	Approximate Dynamic Programming	Max. nr. of served customers
Chen et al. [20]	VRP	✓	✓	✗	Deep Q-Learning	Max. nr. of served customers
Hamid et al. [36]	VRP	✗	✓	✓	Self-adaptive hyper-heuristic	Min. cost, Max. freshness, Max. due-date satisfaction
Saleu et al. [65]	TSP	✗	✗	✗	MILP-based heuristic	Min. makespan
Dell'Amico et al. [28]	TSP	✗	✗	✗	Matheuristic methods	Min. maximum work time

4.2. Dynamic Pickup and Delivery Problems

In pickup and delivery problems (PDPs), goods must be carried from an origin to a destination. These problems are vehicle routing problems and can be categorised into three groups: many-to-many, one-to-many-to-one and one-to-one. In the first group, any node can serve as an origin or destination for a commodity. To illustrate, a famous problem in this category is the *swapping problem*. Each node possesses a specific commodity and desires one in this problem. The goal is to create the shortest possible route so that each customer possesses their desired commodity. In the second group, one-to-many-to-one, goods are stored in a depot and delivered to customer nodes. Additionally, customers store goods that can be taken back to the depot. Lastly, each request has a given origin and destination in the third group. Applications of this group of problems include courier operations. The MDS problem falls in the one-to-one category.

In a dynamic PDP, user requests are revealed over time, and the planning horizon is continuous, contrary to the static version of the problem. Hence, the solution to a dynamic problem should be a solution strategy that specifies the correct action under certain circumstances instead of being a static output. In this category, again, different subproblems exist: the dynamic vehicle routing problem with pickups and deliveries, the dynamic stacker crane problem and the dynamic dial-a-ride problem. The first group allows for the consolidation of packages, while the second does not. The dynamic dial-a-ride problem could be categorised together with the first group. However, the problem differentiates itself due to tight time windows and maximum ride time constraints. The MDS model is a dynamic vehicle routing problem with pickups and deliveries.

Berbeglia et al. [12] describe the general framework of a PDP. A *request* is an order to ship a load from an origin node to a destination node. PDPs are defined on a directed graph $G = (V, A)$, where V is the set of origin and destination nodes, including the depot, and A represents the set of arcs between these nodes. A *route* is a tour along a subset of nodes that starts and ends at the depot. At each time point, all vehicles in the problem either serve a customer node, wait at a customer node or move towards a customer node. The problem contains two types of decisions. When an agent is finished serving a customer node, it can decide to wait at that node or go towards the customer node of the subsequent request. Furthermore, incoming requests can either be accepted or rejected. Of course, many extensions, such as time windows and capacity constraints, are possible. The quality of algorithms that solve PDPs can be assessed by inspecting the total cost or distance travelled, the number of accepted requests, and the computational time.

A frequently used solution strategy for the dynamic PDP is to adapt an algorithm that applies to the static PDP. For instance, a static PDP could be solved each time a new request arrives. In this case, continuity constraints would have to be implemented. A disadvantage of this method is that it is computationally costly. Another method is to solve the static version of the problem once at the start of the planning horizon and then insert a new request using an insertion heuristic. These heuristics are computationally efficient and can be used in real-time computations.

Stochastic PDPs refer to the type of PDPs where there is some prior knowledge of future requests in the form of a probability distribution, which is often approximated using historical data due to the complexity of the problem. This approximation is commonly used in many dynamic PDPs and dynamic vehicle routing problems. Anticipation of future requests can then be implemented using, for instance, scenario sampling. Future requests may also be accounted for using *waiting strategies*. A waiting strategy determines how long a vehicle should wait at a node before continuing its route. Such a strategy could be beneficial in anticipation of future requests. The strategy can also be reversed. Then requests are held for a certain amount of time before being assigned to a vehicle. This is called a buffering strategy. Repositioning of idle vehicles is also considered [12].

Stochastic Dynamic Vehicle Routing Problems (SDVRP) are a class of problems that extend traditional static VRPs by incorporating time-dependent changes and uncertainty represented through probability distributions. This class is closely related to stochastic PDPs. These problems are relevant in on-demand services such as meal delivery, parcel delivery, and ride-hailing, as well as emergency healthcare. SDVRPs are typically modeled using Markov Decision Processes (MDP) that consist of states, actions, rewards, and transition functions. Early research focused on optimizing resource usage based on the current state, while another area concentrated on identifying actions that would be advantageous for future dynamics. Cost function approximation methods integrated policy function approximation into the optimization process, combining comprehensive search of the action space with evaluation of future value. An alternative approach involves data-driven methodologies like reinforcement learning and scenario sampling, where predictions are made about future developments. The multiple scenario approach considers all possible scenarios simultaneously, while the post-decision rollout algorithm narrows down the action space and evaluates potential actions through simulations. These methods thoroughly evaluate actions but require computation effort. Partitioning an SDVRP into assignment and routing has also been considered. By implementing such a partition, the solution space is effectively narrowed down, enabling the utilization of advanced techniques [39].

In recent literature about PDPs the use of electric ground vehicles has also gained attention in pickup and delivery problems [2, 19]. These studies introduce battery constraints into the classical PDP formulation. Agrali and Lee [2] suggest introducing drones as a new vehicle type, while noting that UAVs have much stricter battery and capacity constraints, but can obtain higher speeds. Furthermore, through literature study, it becomes clear that many studies still use static environments to solve their respective problem formulation and propose an extension to the dynamic realm. Moreover, most of the performed experiments are on a small scale and simplified. Hence, a proven strategy that can cope with the complexity of the operations of the MDS project and its intended scale is still missing. Finally, a very limited body of literature exploits repositioning of vehicles in the context of PDPs, creating an opportunity for further investigation in this area.

5 Task Allocation and Scheduling

This chapter describes how the choice for a method that allocates and schedules tasks was made. First, section 5.1 provides some background on task allocation and scheduling. Then, section 5.2 elaborates on the classes of solution methods. Next, section 5.3 dives deeper into the different types of auction mechanisms. Finally, section 5.4 presents a trade-off between the different mechanisms.

5.1. Background of Task Allocation and Scheduling

A first taxonomy of task allocation and scheduling problems presented by Gerkey and Matari defined multi-robot task allocation problems along three axes [34]. The first axis differentiates between problems where robots can only execute one task at a time and problems where robots can perform multiple tasks simultaneously. The agents in these types of problems are called single-task (ST) robots and multi-task (MT) robots, respectively. Along the second axis, the taxonomy distinguishes between single-robot (SR) and multi-robot tasks (MR), referring to the number of robots necessary to complete a task. The final axis makes a distinction between instantaneous assignment (IA) and time-extended assignment (TA). The previous allocates tasks without planning future tasks, while the latter is concerned with current and impending allocations that must be executed according to a schedule.

The problem described in chapter 2 belongs to the MT-SR-TA category. It belongs to the MT category because UAVs can ship multiple orders at a time if all environmental requirements are satisfied. Furthermore, it belongs to the SR category because UAVs cannot cooperate in executing delivery orders. Finally, it is also possible to schedule tasks in advance due to some partners' decreased urgency levels. Hence, a schedule for a UAV can be created.

However, the two authors of the earlier mentioned taxonomy observe that this taxonomy still needs to capture problems with interrelated utilities and task constraints. The authors in [44] propose iTax, a taxonomy which adds to the previously described one. They propose a two-level taxonomy in which the first level describes the interdependencies and the second level provides extra information using the taxonomy provided by Gerkey and Matari. iTax defines four levels of problems. In the first level, no dependencies are present (ND). In these types of problems, the presence of other agents or tasks is independent of the task's utility. The second level contains in-schedule dependencies (ID). In ID problems, the utility of an agent performing a task depends on other tasks that the agent performs. The third level has cross-schedule dependencies (XD). In these types of problems, the utility of an agent performing a task depends not only on its own schedule but also on other agents' schedules. The last problem category contains complex dependencies (CD). In this case, the utility of an agent performing a task depends on other agents' schedules, the agent's schedule, and the decomposition of the complex task from which it originates. Figure 5.1 provides an overview of the previously described dependencies.

According to this taxonomy, the problem described in chapter 2 belongs to the ID category. Each order will bring a UAV to a new location that will determine its proximity to the pickup location of its next order; a larger proximity to the next destination results in fewer costs. Therefore, the utility of the UAV's tasks depends on its schedule. However, there are no relations between different orders, and they cannot be decomposed into simpler tasks. Hence, this problem does not belong to the XD or CD category. The authors of the iTax taxonomy mention that auction-based or market-based approaches are frequently used to solve problems in this category.

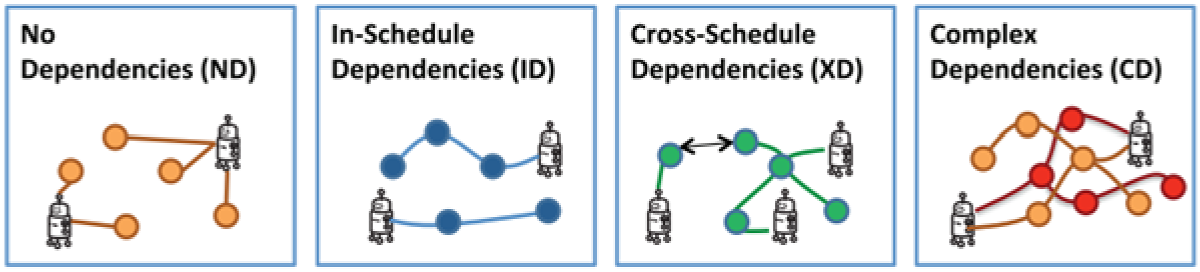

No Dependencies (ND)
In-Schedule Dependencies (ID)
Cross-Schedule Dependencies (XD)
Complex Dependencies (CD)

Figure 5.1: Overview of level 1 dependencies from iTax taxonomy [44].

5.2. Classes of Solution Methods

Chen [18] describes four common classes of solution methods for task allocation, namely optimization approaches, evolutionary and swarm algorithms, auction mechanisms, and game-theoretic approaches. In the following paragraphs each of these methods will be shortly explained.

Optimization approaches formulate the task allocation problem as an optimization model with specific constraints and objectives. These models can be solved using algorithms to find the optimal assignment and scheduling of tasks. Common optimization problems are the vehicle routing problem or the pickup and delivery problem. There exist exact methods to solve these problems such as the Hungarian algorithm or the brand and bound method. However, for large problems heuristic solvers are also applied. Optimization approaches can provide optimal and complete solutions, but also generally require long computation times. Hence, these methods are poorly scalable. Furthermore, because the task allocation happens all at once it is impossible to adjust costs based on the ongoing allocation progress [77].

Evolutionary and swarm algorithms are inspired by nature and can also be used to find solutions to the earlier mentioned mathematical formulations. Evolutionary algorithms use a set of possible solutions that is slowly improved by applying the principles of natural evolution, such as selection, recombination, and mutation. Swarm algorithms replicate swarm movement. Each particle in the swarm is attracted to the best solution found so far, furthermore each particle is also attracted to the best solution that it has experienced itself. Another algorithm that replicates nature is the ant colony optimization algorithm. Here, pheromones are left to indicate the quality of the solutions found. Ants will follow paths based on the pheromone concentration. Evolutionary and swarm algorithms have been widely applied to successfully solve optimization problems [87].

Auctions act as some kind of market mechanism. For task allocation each vehicle would determine its cost and the task would be rewarded to the lowest bidder. Here, it is critical to establish the right bidding and scheduling rules to achieve the desired objective. Contrary to real auctions, auction mechanisms in multi-agent task allocation are cooperative. These methods pose a solution to issues concerning computational time and communication limits for centralized systems. Auction mechanisms were successfully applied for dynamic pickup and delivery problems in [18] and [63].

Game-theoretic approaches to task allocation involve modeling the strategic interactions between self-interested agents. Agents are viewed as rational entities seeking to maximize their own utility. Cooperative game-theoretic approaches focus on forming coalitions and coordinating actions to achieve mutual benefits, while non-cooperative game-theoretic approaches analyze individual decision-making and seek stable outcomes such as Nash equilibria. These approaches aim to allocate tasks efficiently by considering factors such as fairness, stability, and efficiency.

The author of [63] classifies auction mechanisms as the most appropriate approach to pick-up and delivery problems. They are *robust* since they can include uncertainties in bids and are adaptable to a changing environment, contrary to centralized approaches, such as optimization approaches or evolutionary/swarm algorithms. The reason for this is that these methods have to resolve the whole problem when changes occur. Although these methods do not guarantee optimal solutions, they still provide decent, and sometimes suboptimal, *solution quality*. Furthermore, auction mechanisms are *scalable and efficient*. They outperform optimization techniques, which need long computational time for large scale instances. Although evolutionary/swarm algorithms are fast, they also require many iterations to converge. Finally, auction mechanism are very *flexible*, allowing the system designer to tailor the mechanism to their own preference. Therefore, auctions mechanisms will be considered for task allocation and scheduling of UAVs in the MDS project. The next section will elaborate on the different types of auctions.

5.3. Auction Mechanisms

In the realm of auction theory, there exist multiple classes of auction mechanisms. These are categorized based on their distinct rules and mechanisms. It is possible to make a first division by separating mechanisms based on the presence of a central auctioneer.

Auctions in which a central auctioneer is present are, for instance, *parallel* or *combinatorial* auctions. In parallel auctions tasks are allocated via independent and simultaneous single-round auctions. As a result, synergies between tasks are not taken into account, hence the resulting total cost is high. Combinatorial auctions allocate all tasks in one single-round auction. Here each agent bids on all bundles of targets. This method takes synergies between tasks into account. However, a significant challenge arises as the generation, transmission, and processing of bids (exponential in the number of tasks) requires much more computational power than parallel auctions. *Sequential single-item* (SSI) auctions combine the best of both of the previously mentioned auctions. In this type of auction all tasks are allocated in one auction consisting of multiple rounds. Every agent bids on all tasks, and the winner is the one provided the lowest bid to a task that is then allocated to it. The bid is made up of the smallest cost of adding the task to the agent's task set (consisting of previously allocated tasks). There exist a number of extensions to this type of auction, such as roll-outs, bundle bids, and regret clearing [43]. *Sequential single-cluster* (SSC) auctions are another step towards combinatorial auctions. In a similar fashion to SSI auctions, clusters of tasks are auctioned to agents. This type of task allocation is able to capture synergies between tasks, while also being able to reduce computational time [38].

Auctions without a central auctioneer include the consensus-based bundle algorithm (CBBA). This algorithm is first presented by Choi et al. as a generalization of the consensus-based auction algorithm (CBAA). The CBBA algorithm consists of two phases. In the first phase, all agents create their own bundle of tasks. In the second phase, conflicts between the constructed bundles are resolved. The authors prove that CBBA guarantees 50 % optimality [21]. Many extensions to CBBA exist. For instance, there are algorithms capable of handling time windows, heterogeneous agents, re-planning of tasks, and mandatory tasks [13, 59]. Furthermore, the Asynchronous CBBA was formulated to improve the consensus phase of CBBA, which is sensitive to communication overflow for larger problem instances [41]. A special extension to CBBA is the performance impact algorithm. This algorithm introduces two metrics used in the task inclusion and consensus phase: inclusion performance impact and removal performance impact. Through these metrics, the performance impact algorithm is able to solve time-critical problems that CBBA cannot. An extension to this algorithm known as PI-MaxAss, maximizes the number of assigned tasks through a redefinition of the earlier mentioned metrics. The authors of this extension state that it could also be used in combination with any other scheduling method [81].

Table 5.1 shows an overview of the discussed allocation techniques. In the context of the MDS network, as stated in chapter 2, it is important that an allocation and scheduling method is centralised and cooperative, as this reflects the true concept of operations. Therefore, algorithms such CBBA and performance impact are not considered. Furthermore, it is important that synergies for regular transport requests are captured to increase profitability. For this reason, parallel auctions are also disregarded. Finally, combinatorial auctions can be challenging computationally. While, they could work for small instances, they are not suitable for large problems. Because scalability is a requirement for the MDS network, these type of auctions are also not considered. This leaves SSI and SSC auctions. The next subsections further elaborate on these type of auctions. However, emphasis will be put on extensions that can handle temporal constraints. The reason for this is that the auction method should solve a pickup and delivery problem with deadlines, hence considering methods that do not include temporal constraints is of no use.

5.3.1. Temporal Sequential Single-Item Auction

The first method that will be discussed is the Temporal Sequential Single-Item auction (TeSSI) algorithm. This algorithm allocates tasks using an extension to the SSI algorithm. The algorithm uses an auctioneer that communicates tasks to agents, receives bids, determines the allocation and communicates this to agents. The advantage of an auctioneer is that they ease communication among agents and enables a comprehensive overview of individual allocations. However, contrary to centralised approaches the auctions offer the benefit

Table 5.1: Overview of auction mechanism characteristics.

	Centralised	**Number of rounds**	**Account for inter-task synergies**	**Computational Power**
Parallel auction	✓	Multiple	Low	Low
Combinatorial auction	✓	Single	High	High
SSI	✓	Single	Medium	Medium
SSC	✓	Single	Medium-High	Medium
CBBA	✗	Two	Medium	High
ACBBA	✗	Two	Medium	Medium
PI-algorithm	✗	Two	Medium	High
PI-MaxAss	✗	Two	Medium	High

of distributing bid calculations among agents. The agents function as bidders, with each agent independently determining the cost associated with executing each task based on its own private schedule.

The process of TeSSI auctions, works the following way. It starts when available tasks are announced for bidding by the auctioneer. Each agent sends a bid vector and the auctioneer selects the task with the lowest bid out of all the sent bids. This task is then allocated to the agent that sent the bid and each agent is notified about the allocation. Tasks that no robot is able to perform are added to the set of unallocated tasks. The auction then restarts with the remaining set of unallocated tasks and continues until this is empty.

Each agent computes its bid by considering its current schedule. This schedule is kept as a spatial temporal network (STN). An example of an STN is displayed in Figure 5.2. In an STN, the time points S_t and F_t represent each individual task. Furthermore, it contains an origin time point that references the starting point (omitted in Figure 5.2). Self-loop arrows are used to represent the fact that time windows are present. Hence, a task must be executed between ES_t and LF_t. Two types of constraints are present between each pair of time points: travel time and duration constraints. Travel time constraints are enforced when two consecutive time points belong to different tasks. These constraints also enforce that a task is finished before the agents heads on to the next task. When two time points belong to the same tasks, duration constraints are enforced. These constraints ensure that a task cannot finish before it is started. The agents calculate a bid for a task by attempting to place the task in each possible time segment in the agent's schedule and choosing the position that minimizes the makespan while avoiding any temporal conflicts. Tasks are added by adding the start and finish time points to the already existing STN, together with the accompanying constraints. They STN is reset after each try and is only updated when the agent actually wins the auction.

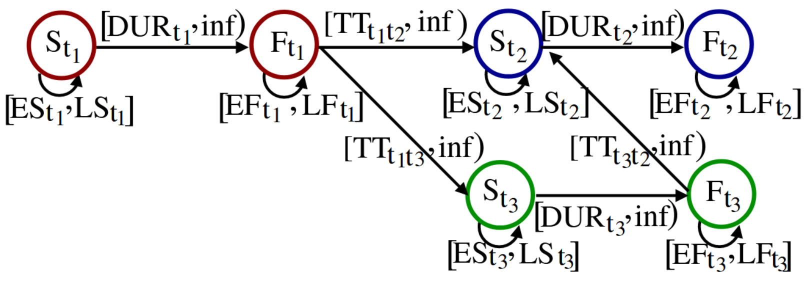


Figure 5.2: Example of a spatial temporal network as given by [54].

Nunes and Gini, the authors of the TeSSI algorithm [54], consider two types of objectives. Agents can either bid using the makespan of their schedules or combine it with the distance traveled. The latter means that the makespan and total travelled distance are linearly combined and is referred to as TeSSIduo. Note that minimizing the makespan, maximizes the agents' availability.

The TeSSI auction produces in total $n \times m$ bids, where n represents the number of agents and m the number of tasks, for m iterations (every tasks is auctioned separately). Hence, the resulting complexity is $\mathcal{O}(nm^2)$. Furthermore, scheduling tasks for each agent requires up to $\mathcal{O}(|T_r|^2)$, where $|T_r|$ represents the size of the set of tasks allocated to an agent. Additionally, the Floyd-Warshall algorithm that propagates an STN after adding a task is $\mathcal{O}(m^3)$ for worst-case scenarios.

Results of experiments done using the TeSSI algorithm show that TeSSI outperforms CBBA and a greedy algorithm in terms of number of necessary agents and number of allocated tasks. The use of the makespan to conduct bids allows TeSSI to allocate more tasks than CBBA, because tasks can be moved around. For dynamically arriving tasks, it becomes apparent TeSSI performs just as well as a greedy algorithm, when not many tasks are available, because it is not able to incorporate the synergies between tasks. However, when many tasks are available, the performance of TeSSI quickly increases. Experiments also show that clustered tasks based on, for instance, their relative distance allow TeSSI to allocate more tasks. Furthermore, it shows that TeSSIduo is able to generate paths that are consistently lower in total distance, without significantly increasing the makespan. Hence, when the total travelled distance is of importance, TeSSIduo is preferred. Next to that, the computation time of the TeSSI algorithm is two orders of magnitude less than the computation time of the CBBA algorithm [54].

In conclusion, concerning scalability and efficiency the TeSSI algorithm seems suitable. However, taking advantage of synergies is important and the TeSSI algorithm does not fully satisfy this requirement. Furhtermore, the STN presented does not allow for mixed pickup and delivery tasks because a task cannot start before the other task is finished. Finally, when TeSSI is not able to allocate a task it puts it in the set of unallocated tasks. As a high number of allocations is preferred, this is a feature that should be improved upon.

5.3.2. Probabilistic Temporal Sequential Single-Item Auction

Rizzo and Sharpanskykh continued the work on the TeSSI algorithm in the context of pickup and delivery problems with time windows. They state that time windows substantially complicate the allocation problem, because agents have to reflect on the temporal consistency of their schedules, rather than only spatial synergies. Two shortcomings of TeSSI algorithm are given:

- The STN representation lacks flexibility in enabling agents to express and analyze potential sources of uncertainty in task durations.
- Re-auctioning of tasks is impossible. Once an allocation is made, it cannot be revisited, hence limiting the number of synergies that can be exploited.

A probabilistic version of TeSSI (pTeSSI) is proposed that includes re-auctioning of tasks. This algorithm is proposed in the setting of meal delivery, where each delivery includes a pickup point and a delivery point, and a time window in which the order must be completed. Contrary to TeSSI, durations of tasks are regarded as random variables with known distributions. The principle of the auction method is the same as that of SSI and TeSSI auctions.

Two team objectives to be minimized are presented, shown in Equation 5.1 and Equation 5.2. Here, S represents set of all agent schedules. Furthermore, $\mathcal{M}_i(S_i)$ represents the minimum time an agent i needs to finish its schedule S_i. Next to that, $\mathcal{D}_i(S_i)$ refers to the total distance agent i needs to travel to finish its schedule S_i. Note that each of the two objectives contains the penalty $\rho \cdot \mathcal{R}_i(S_i)$. This penalty is meant to capture the risk of unsuccessful dispatch of a schedule. Here, ρ indicates how heavily unsuccessful dispatch is penalized, while $\mathcal{R}_i(S_i)$ indicates the probability of an agent not being able to finish its schedule. For each task auction an agent bids the minimum incurred cost of being allocated a task, using mathematical reformulation it is deducted that the bids that are conducted follow Equation 5.3 and Equation 5.4 for the two objectives respectively. In this formulation P_i refers to the partial allocation before the auctioned task is allocated and P_i' to the partial allocation after allocation of the task. It is concluded that to determine $\mathcal{M}_i$ and $\mathcal{D}_i$ a TSP with time windows with time windows has to be solved, which is an NP-hard optimization problem. The formulation of this problem using an STN allows for determining these variables in a highly efficient way.

$$\texttt{MAX-T}: \mathcal{F}(S) = \max_i [\mathcal{M}_i(S_i) + \rho \cdot \mathcal{R}_i(S_i)] \tag{5.1}$$

$$\texttt{SUM-DIST}: \mathcal{F}(S) = \sum_i [\mathcal{D}_i(S_i) + \rho \cdot \mathcal{R}_i(S_i)] \tag{5.2}$$

$$\mathcal{M}_i(P_i') + \rho \cdot \mathcal{R}_i(P'_i) \tag{5.3}$$

$$\mathcal{D}_i(P_i') + \rho \cdot \mathcal{R}_i(P_i') - (\mathcal{D}_i(P_i) + \rho \cdot \mathcal{R}_i(P_i)) \tag{5.4}$$

The authors introduce the spatial temporal network with uncertainty (STNU) as displayed in Figure 5.3. Four actions are used to represent a single pickup and delivery task: travel to pickup location, perform pickup action, travel to delivery location, perform delivery action. For each of these variables the duration is respectively denoted as TT_{pi}, p_i, t_{di}, and d_i for a task T_i. In the STNU curved arrows, also called *contingent* constraints, are used to represent when these variables are uncertain. They are then interpreted as a random variable with upper and lower bounds. The straight arrows represent *requirement* constraints to enforce the sequence in which the different tasks must be carried out. Finally, the dashed arrows indicate the time window in which the task must be carried out with respect to the origin node. This node is continuously updated based on the agent's current position and state. Minimizing the number of nodes utilized for representing a schedule and applying compacting operations whenever possible is always preferable, as the runtime of algorithms executed on the STNU escalates with the network's size.

Just like in the TeSSI auction, an agent positions a task up for auction at each possible position in its STNU. Eligible positions include the one right after origin node or after each last node of another task. The validity of each insertion is checked by inspecting if for two consecutive tasks the earliest delivery time of the first task is no later than the latest delivery time of the second task. For a valid insertion point, the agent first the determines the risk associated with adding this task to its task set. If this risk is below a certain threshold the agent will compute its bid. The agent submits the best computed bid from insertion point $i*$ to the auctioneer. After winning the auction the agent will directly insert the task at position $i*$.

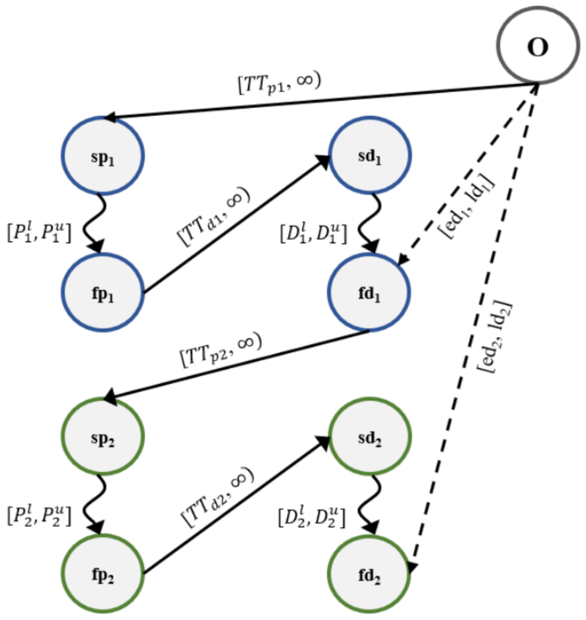


Figure 5.3: Example of a spatial temporal network with uncertainty as found in [63].

To compute risk of adding a task to an existing schedule the degree of dynamic controllability is used. This is also known as the probability that the realization of contingent edges is such that network is dynamically controllable. First, the network is inspected for conflicts. These are sets of constraints that cannot be satisfied simultaneously by any execution strategy. When a network contains no conflicts, it is said to be dynamically controllable. Otherwise, the operator investigates how the intervals on contingent edges should be shrunk to resolve conflicts and ensure dynamic controllability. Then the probability is computed that contingent edges will actually fall within the shrunk intervals.

The complexity of the bidding procedure in pTeSSI is described to be $\mathcal{O}(mn + m^2)$. This is because after the first round, where each agent bids on each task, in the next rounds only the winning agent has to reevaluate its bids for the remaining tasks, since its schedule has changed. Furthermore, the bidding and scheduling algorithm has an overall complexity of $\mathcal{O}(\frac{m^6}{n^4})$, which is similar to the original TeSSI algorithm although the operations are more elaborate. Finally, the algorithm always terminates, however optimality of the final allocation is not guaranteed due to the nature of the sequential auction.

The main disadvantage of pTeSSI (and TeSSI too) is that the number inter-task synergies to can be considered is limited. The current auction strategy only considers the synergies between the task set on auction and each of the agent's individual schedules, but no combinations. To solve this problem, agents are allowed to temporarily decommit from the tasks in their schedules an re-auction them. To decrease the duration of the auction tasks are bundled together. The version that makes bundles of tasks is called the probabilistic temporal sequential single bundle (pTeSSB) auction. Tasks selected to remove from an agent's schedule by computing the improvement of the objective by removing the task. If this improvement exceeds a certain threshold, the task is removed from the agent's task list. Bundles are constructed using an hierarchical agglomerative mechanism. For two tasks, first their sequence is inspected for *validity*. This term points to whether it is possible to carry out the two tasks consecutively with acceptable risk. Afterwards, the distance between two tasks is measured through computing the alignment of the time windows and the travel time. Bundles are only combined if the distance is below a certain threshold. The bidding phase is similar to pTeSSI. However, a bid is now constructed by inserting every task of a bundle B into the agent's schedule. Only when the agent can successfully insert all task, it will bid a valid number. The authors state that the number of generated bid is now less than before but that this offset by the extra operations required in the bidding and scheduling procedure. To mitigate this effect, the bidding procedure has been modified, practically making pTeSSB faster than pTeSSI. It is noted that pTeSSI (and other TeSSI auction algorithms) makes wiser decisions when the set of already allocated tasks is larger as it able to better capture synergies with the agent's schedules. Therefore, the threshold for decommiting from tasks should not be too low [63].

While the re-auctioning mechanism is interesting for dynamic allocation problems, the STNU is an addition to the MDS network that does not add to the current objective. The bundling of tasks is interesting, however, the authors mention little on how bundling tasks affects the objective.

5.3.3. Adapted TeSSI Auction

To allocate and schedule tasks for ground support equipment vehicles, the author in [18] used an adapted version of the TeSSI algorithm. They state that the strength of the TeSSI algorithm is its ability to handle temporal constraints. However, they modify the algorithm because the STN is substituted by a pickup and delivery optimization model. The bidding mechanism remains the same, just like the two objectives of minimizing the makespan and the total distance (or travel time).

When tasks are supposed to be inserted into the STN to compute the new makespan, an optimization algorithm is carried out to generate a schedule. There is limit on the computation time, to ensure that the algorithm is suitable for real-time implementation. In the case that an optimal solution is not reached before the time limit, a feasible solution is returned. When it is also not possible to find a feasible solution, the agent will bid an infinite number. The algorithm solves the single pickup and delivery problem tailored to the concept aircraft ground handling operations. An example of a schedule is provided in Figure 5.4. The advantage of such an algorithm is that it allows for mixed pickup and delivery operations, something that is not possible when using an STN.

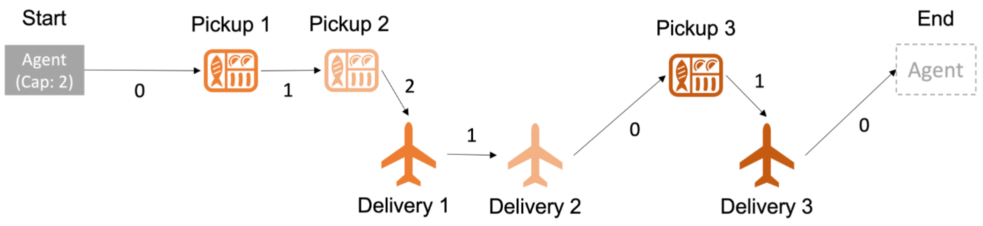


Figure 5.4: Example of schedule generated by single vehicle pickup and delivery problem algorithm [18].

To reduce the allocation time, tasks are bundled. The idea of non-overlapping bundles taken from the bundle generation problem (BuGP) is used to construct bundles. However, in this case a heuristic is used. Whenever it is impossible to allocate a bundle, the bundle is split into individual tasks and these tasks are allocated. It is stated that bundling is not ideal as if affects the overall optimality of the auction results. Furthermore, the objective of reducing the overall computation time is not achieved through bundling tasks. Two possible reasons are given for this. The first reason is that it could be possible that the integration between the bundles and the optimizer has to be improved. The second reason is that through the generation of bundles, the time windows might have become tighter, therefore increasing the problem complexity.

To cope with uncertainties, two measures are taken. The first measure introduces buffer coefficients into computations to deal with uncertainties. The other allows for re-allocation because of disruptions in the operational process. Only tasks that the disrupted agent has not yet started to process are reallocated. However, although re-planning in this model provides higher task allocation rates it did not yet manage to improve the makespan.

5.4. Trade-Off

This section aims to provide a concise summary of the decision-making process and present the final selected method.

Firstly, Table 5.2 displays the decision regarding the category of methods that deserves greater consideration. The criteria used in the table are defined as follows:

- **Robust:** Ability to handle uncertainties and adapt to a changing environment.
- **Scalable and efficient:** Presence of computational requirements that are deemed acceptable.
- **Solution quality:** Acceptable performance level concerning solution optimality.
- **Flexibility:** Capability to customize the algorithm according to the preferences of the designer.

Based on the analysis presented in the table, it can be inferred that auction mechanisms are the most suitable for the MDS network. Subsequently, decentralized auction mechanisms were excluded from consideration as they do not accurately reflect the operational setting of the MDS network.

Table 5.2: Overview of whether different allocation and scheduling criteria satisfy the minimum requirement for model criteria.

	Robust	**Scalable and Efficient**	**Solution Quality**	**Flexibility**
Optimisation	✓		✓	
Evolutionary/Swarm Algorithms		✓	✓	
Auction Mechanisms	✓	✓	✓	✓
Game-Theory	✓			

The decision-making process narrows down the options to parallel, combinatorial, and sequential auctions. Among these, sequential auctions emerge as the primary candidate due to their satisfactory solution quality and efficiency. Further filtering for methods that accommodate temporal constraints leads to the consideration of extensions of the TeSSI algorithm.

Consequently, the adapted TeSSI auction is identified as the most suitable choice. This decision is based on the fact that the optimizer used in this algorithm enables the incorporation of mixed pickup and delivery operations, which is not possible with the STN approach. Moreover, allowing agents to decommit from tasks, as described in the pTeSSI algorithm, addresses the dynamic nature of the problem. While the STNU approach described in pTeSSI is appealing for handling uncertainty, incorporating margins and allowing for reallocation in extreme scenarios will effectively address the uncertainty inherent in the MDS network.

6
Vehicle Repositioning

This chapter discusses the method selection for repositioning UAVs in the network. First, section 6.1 discusses the background of repositioning and proposes suitable methods. Then, section 6.2, section 6.3, and section 6.4 each treat one of the proposed methods. Finally, section 6.5 makes a trade-off between the different methods.

6.1. Background of repositioning

Vehicle repositioning is crucial in balancing supply and demand across various industries, including emergency healthcare, the taxi industry, and bike-sharing systems. The efficient location of emergency vehicles, such as ambulances, can significantly affect the patient's survival. Therefore, it is essential to strategically position these vehicles throughout an area to ensure prompt responses to emergencies. Similarly, in the taxi industry, it is crucial to have sufficient vehicles available to cater to the customers' demands at any given time. Repositioning vehicles in high-demand areas can reduce customer waiting times and cruising times for taxi drivers, ultimately increasing customer satisfaction. In the case of bike-sharing systems, moving bikes from less popular stations to high-demand areas can ensure that enough bikes are available for users to rent. This strategy can help bike-sharing companies increase usage rates and profitability.

Sayarshad and Chow [69] state that several studies have indicated that dynamic models capable of looking ahead, also known as "non-myopic" models, perform better than myopic dynamic models. These studies have covered the areas of pickup and delivery problems, fleet management, adaptive network design problems, and facility relocation problems. Non-myopic policies lead to better-informed decisions and are therefore beneficial. Therefore, primarily non-myopic methods are considered for the repositioning of UAVs. Bélanger et al. present three different categories of methods to treat dynamic vehicle relocation problems in emergency medical services [16]. These categories are online methods, compliance table/offline methods, and learning based methods and each of these categories will be considered for method selection.

6.2. Real-time Relocation Models

Real-time approaches strive to determine optimal relocation plans by considering the system's current state when making decisions. These approaches involve solving or approximating relevant models each time a decision is required. Because UAV relocation in the MDS network resembles ambulance relocation, such a real-time model would be based on static ambulance location models, such as the double standard model (DSM). This model tries to maximize the demand covered twice within a certain timespan S. Furthermore, it ensures that a fraction of the demand is covered within S, while all demand is covered within S', where $S' > S$. The proposed ambulance relocation problem (RP^t) is based mainly on the DSM but also seeks to minimize relocation costs simultaneously. Since this model must be solved each time a vehicle is sent to perform a pickup and delivery request, it can become computationally expensive. Therefore, the developers propose to compute the solution for each possible allocation decision beforehand.

Other static ambulance location models could serve as a basis for real-time ambulance relocation. The earliest optimization models are the location set covering problem (LSCP) and the maximal covering location problem (MCLP). While the former seeks to minimize the number of required vehicles to satisfy coverage constraints, the latter tries to maximize coverage for a given number of vehicles. Hence the former is helpful for strategic fleet decisions, while the latter can be employed to make fleet management more efficient. Many variants of these two formulations exist, such as the earlier presented DSM, to overcome the initial assumptions made. Furthermore, probabilistic and stochastic versions of these problems exist. One such model is the maximum expected covering location problem (MEXCLP) and extension of the MCLP, which incorporates vehicle availability through a busy fraction.

Furthermore, techniques have emerged to avoid requiring recomputation each time a vehicle is dispatched. This method defines a *preparedness measure*, which denotes the system's ability to meet future demands. Equation 6.1 shows how this measure is computed. Here, i represents a demand zone. Furthermore, a_i represents the weight associated with this demand zone based on the fraction of requests it generates. K_i denotes a predetermined number of vehicles (typically the closest) that will be used in computing the preparedness. t_i^k represents the travel time for each vehicle k to zone i and γ^k the contribution factor associated with each vehicle. When the preparedness for a zone drops below a certain threshold, the computation for the relocation of vehicles is launched.

$$\rho_i = \frac{1}{a_i} \sum_{k=1}^{K_i} \frac{\gamma^k}{t_i^k} \tag{6.1}$$

Finally van Barneveld et al. [84] developed a method for ambulance management in rural regions, where the number of ambulances is limited. As a consequence, dispatching a vehicle to a request affects coverage more. Furthermore, the demand variance in different zones is much higher, affecting en-route coverage and thus making ambulance relocation more risky. To solve their problem, authors developed a one-step look-ahead heuristic that can act according to different performance measures. The heuristic allows for a relocation decision at each time step instead of only when an event occurs. The dynamic ambulance relocation process is formulated as an MDP. The action space consists of dispatching an ambulance to its neighbouring nodes or letting it hold its current position. Given an incoming request, the heuristic chooses an action that minimizes the weighted penalties for the minimum achieved response time in each next possible state. A limitation of this heuristic is that it only allows for gradual changes in the ambulance configuration since ambulances can only move to their neighbouring nodes. Furthermore, many extensions, such as incorporating request priority levels and multiple ambulance types, are still possible. Lastly, the authors state that this heuristic outperforms methods based on compliance tables.

6.3. Compliance Tables

Compliance tables are offline methods to compute vehicle repositioning strategies. For larger instances, more computational power is required for online methods making real-time decision-making harder. These methods solve the ambulance relocation problem for each possible system state a priori. However, in this case the system state only depends on the number of available or idle vehicles. Hence, these types of methods contain little detail about the system's environment. A weakness of compliance tables is that there is no cohesion between different states, resulting in unwanted relocations of vehicles. This challenge can be solved by putting bounds on the number of relocations that occur when transitioning from one state to another. The strength of the method is that it is easy to complain and that the tables can be computed in advance.

One example of the application of compliance tables is presented by van Barneveld [83]. He introduced the *minimum expected penalty relocation problem* to compute compliance tables. The objective of this problem can be based on several performance measures such as maximizing total coverage, which why the authors introduced the penalty function. Furthermore, the authors extend their model in such a way that it overcomes the binary definition of coverage and is more realistic concerning the availability of ambulance waiting sites. Additionally, the authors investigated the effects of a relocation threshold to prevent unnecessary relocations.

Aside from computing compliance tables, vehicles also have to be assigned to a waiting site when the number of available vehicles changes. According to van Barneveld [83] this can either be done by solving a minimum weighted bipartite matching problem to minimize the total travel time, or by solving a linear bottleneck assignment problem to minimize the maximum experienced travel time. For these two problems fast solving methods exist, therefore they could be solved online.

6.4. Reinforcement Learning

As mentioned in section 6.1, non-myopic policies, policies capable of looking ahead have been shown to perform better than myopic policies. Reinforcement learning, therefore, presents an interesting opportunity. Reinforcement learning tries to find the optimal way of interacting with the environment to maximize the reward signal. The method is especially useful for large state spaces and is one of the first methods to address computational issues that may arise in this type of problem setting.

Most reinforcement learning problems share the following characteristics. First, there is interaction between a decision-making agent and its environment. Furthermore, the agent aims to achieve a certain goal within its environment, which most of time contains some kind of uncertainty. As a result, the agent's actions can affect future states, but never fully determine the next state. Hence, the agent must display foresight to achieve its goal. Most of reinforcement learning theory is based on environments that satisfy the *Markov property*. This property states that all future states can only depend on the current state. If the environment satisfies the Markov property the environment is called a *Markov decision process* (MDP). The objective in reinforcement learning is to find the optimal policy. This policy maps actions to states such that the reward signal is maximized.

Two reinforcement learning methods are of interest for repositioning. The first method is approximate dynamic , as it has been successfully applied in previous studies by Nasrollahzadeh et al. and Schmid [51, 70]. The second is Monte Carlo methods, since this method is model-free and can be applied offline. subsection 6.4.1 elaborates on the theory of dynamic programming and approximate dynamic programming, while subsection 6.4.2 elaborates on Monte Carlo methods.

6.4.1. Dynamic Programming

This subsection discusses dynamic programming. First, the theoretical concept is discussed. However, approximate dynamic programming is also highlighted since dynamic programming is only sometimes applicable due to high computational requirements. These heuristics are based on the core idea of dynamic programming.

Exact Method

Dynamic programming can be used used to solve finite MDPs. In a finite MDP, the number of possible states and actions is finite. The method assumes that the environment's dynamics are fully known. These dynamics are represented by $p(s', r|s, a)$. This expression refers to the probability of transitioning to state s' and obtaining reward r, while choosing action a in state s. The main essence of dynamic programming is to use state value functions to ease the search for adequate policies.

The state value function can be approximated using the Bellman equation. Equation 6.2 shows the update for each iteration of the state value function used by dynamic programming.

$$\begin{aligned} v_{k+1}(s) &= \mathbb{E}_\pi[R_{t+1} + \gamma v_k(S_{t+1})|S_t = s] \\ &= \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a)[r + \gamma v_k(s')] \end{aligned} \tag{6.2}$$

Where:

- $v(s)$ is the state value function
- k is the iteration number
- t is the time step in the model
- R_{t+1} is the reward signal obtained at time t+1
- r is a realisation of R_{t+1}
- γ is the discount factor
- S represents the set of all possible states
- s is realisation of S
- π is the policy that is adhered to

- a is a possible action

This iterative process is usually repeated when the absolute difference between v_{k+1} and v_k is sufficiently small. Once the value function is determined, this function is used to improve the current policy by making it greedy towards the newly determined value function as shown in Equation 6.3.

$$\pi'(s) = \arg\max_a \sum_{s',r} p(s', r|s, a)[r + \gamma v_\pi(s')] \tag{6.3}$$

The process of approximating the value function and improving the policy based on this approximation is repeated as shown in Equation 6.3 until both are value function and policy are optimal. Dynamic programming is quite efficient because the time to find an optimal policy is polynomial in the number of states and actions. Hence, it is faster than any exhaustive policy search that provides the same optimality guarantee. Linear programming methods provide better convergence guarantees but become impractical at a much smaller number of states.

$$\pi_0 \rightarrow v_0 \rightarrow \pi_1 \rightarrow v_1 \rightarrow ... \rightarrow \pi* \rightarrow v* \tag{6.4}$$

Although dynamic programming is efficient, its disadvantage is that it involves operations over the entire state set. For large state spaces, this can be very expensive computationally. *Asynchronous* dynamic programming poses a solution to this issue. Here, instead of the method updating the value function for all states, some subset of states is improved. As a result, the policy can be improved more quickly [78].

Approximate Dynamic Programming

Approximate dynamic programming is a branch of dynamic programming that addresses the challenges posed by high-dimensional problems that suffer from the 'curses of dimensionality' [61]. It aims to find solutions that are close to the optimal policy by evaluating the post-decision state based on an estimated value function.

To achieve this, approximate dynamic programming employs various techniques. One important aspect is simplifying the stochastic dynamic system described earlier. This can be done by aggregating states, which involves grouping similar states together and computing a single value for each group. This reduces the complexity of the problem. Additionally, certain areas of the state space that are unlikely to be relevant can be excluded.

The action space can also be streamlined using a technique called decomposition. This approach divides the problem into a top-level and base-level subproblem, such as task assignment and routing. The top-level problem is thoroughly analyzed, while the sub-level problem may be addressed using heuristics or simplified models. Another option is to omit details and aggregate decisions, which further reduces complexity. Finally, the number of decision points can be minimised.

In approximate dynamic programming, there are several methods available, such as the roll-out algorithm, which approximates the value function, and approximate value iteration.

6.4.2. Monte Carlo Methods

Monte Carlo methods differ from dynamic programming because they only require experience and no knowledge of the model. This experience could either be a real-time or simulated experience. The latter still requires some model knowledge but no complete knowledge of all the transition probabilities, which can be a significant advantage. The method is based on averaging sample returns for episodic tasks. An *episodic task* is a task for which there exists a terminal state that ends the task. This property is valid for, for instance, playing a chess game.

In Monte Carlo methods, an episode is generated using a policy π. Returns are stored each time that state s is visited and averaged at the end to determine $v_\pi(s)$. The method is proven to converge to the actual value function as the number of visits to state s goes to infinity. As a result, contrary to dynamic programming, each state is assessed individually, and its value estimate is not based on the value estimates of other states. This feature is attractive when one only wants to examine a subset of all possible states. Monte Carlo methods can estimate state action values $q_\pi(s, a)$ in the same manner by averaging returns after a visit to state action pair (s, a). This characteristic is beneficial for improving the policy as it only has to be made greedy towards $q_\pi(s, a)$. The process of improving the state value function and the policy is the same as the one described in subsection 6.4.1. Monte Carlo methods have to overcome two significant problems. The first problem is obtaining a satisfactory result without an infinite number of simulations. The second problem is how to ensure that all states are visited.

To summarize, Monte Carlo methods are beneficial because they allow the agent to learn optimal behaviour from interaction with the environment without full knowledge of its dynamics. This feature is advantageous because, in many applications, it is hard to construct an explicit model such as the one required for dynamic programming. Furthermore, the method allows the user to focus on smaller subsets of states. Finally, because the method does not bootstrap, it is less sensitive to violations of the Markov property [78].

6.5. Method Trade-off

Having thoroughly examined and discussed all the candidate methods suitable for vehicle repositioning, a comparative analysis can be made to evaluate and trade-off these approaches. To achieve this the the following criteria are used to assess the different methods:

- **Modularity:** Refers to the ease of integration of the method with the rest of the model.
- **Flexibility:** Refers to the possibility to customize algorithm according to the preferences of the designer.
- **Explainability:** Refers to the ability of a method to provide clear and understandable explanations of its learning and decision-making processes.
- **Anticipative:** Refers to the ability of a method to estimate the value of an action by considering future consequences or looking ahead in the decision-making process.
- **Scalability and efficiency:** Assesses the computational requirements of the proposed methods.

Because each of the previously presented methods are unmistakably centralised and cooperative, these two criteria are omitted from the trade-off performance measures. They will be more relevant to the issue of task allocation, where decentralised methods are more prominent.

Furthermore, the earlier presented methods are categorised into four categories: real-time methods, compliance tables, Monte Carlo methods, and approximate dynamic programming. Real-time methods point to the methods that solve optimization problems in real-time or heuristics that are applied in the same manner. Compliance tables are methods that precompute the solutions to optimization problems for different system states. Monte Carlo methods and approximate dynamic programming are both reinforcement learning methods, but they have been separate because of the large differences in their nature.

Figure 6.1 shows the results of the trade-off of the method categories. Each of the methods was given a score from 1 to 4 (4 being the best score). As can be read from the diagram, Monte Carlo methods are the best performing category for repositioning. An explanation on the scores for different performance criteria is given in the following paragraphs.

For the *modularity* performance criterion, Monte Carlo methods score the best. These methods sample and estimate the action values through repeated interaction with the environment, in which each episode is standalone. This model-free approach allows for incremental improvements, while also allowing system adjustments. Approximate dynamic programming scores the worst because it allows for bootstrapping, thus modifications to a specific part of the system affect the entire process. Furthermore, ADP models require tuning of parameters, hence adjusting the environment will affect the system configuration. Real-time optimization methods and compliance tables score higher than ADP, but below Monte Carlo methods. The reason for this is that the optimization algorithms used, the systems dynamics, and the decision-making process are tightly coupled. Hence, altering a part of the model could require significant alterations to the whole system.

Regarding the criterion of *explainability*, it is observed that Monte Carlo methods and compliance tables exhibit the best performance among the various method categories. Monte Carlo methods provide a clear and intuitive approach to learning and decision-making, as they directly simulate and evaluate episodes. The estimation process in Monte Carlo methods relies on empirical averages, enabling relatively straightforward interpretation and explanation of the outcomes. In contrast, approximate dynamic programming and real-time optimization techniques often involve intricate mathematical formulations and complex non-linear function approximators, optimization algorithms, or control policies. Consequently, these methods pose challenges when it comes to providing clear explanations, particularly in understanding the rationale behind specific decisions or the impact of parameter choices. In terms of explainability, compliance tables outperform real-time techniques as they explicitly specify the preferred vehicle configurations in each situation, although potentially lacking clarity regarding the underlying reasons behind such specifications.

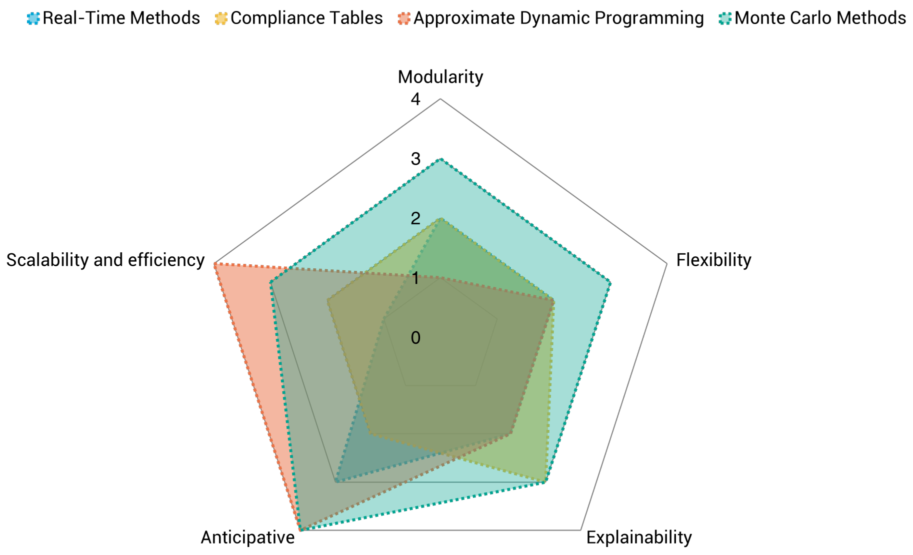


Figure 6.1: Diagram of trade-off for vehicle repositioning methods.

The next criterion is *scalability and efficiency*. Here, ADP methods score the highest because reinforcement learning methods are meant to address computational problems in large solution spaces. Monte Carlo methods score slightly worse because they have the potential to be computationally more expensive. Real-time methods, although providing near-optimal solutions, can become infeasible for large instances. While compliance tables aim to solve this by computing solutions beforehand, these methods contain little detail about the environment. Therefore, these methods will provide worse results than the others.

Concerning the methods' *anticipative* skills, for this criterion both Monte Carlo and ADP methods score the best. Both of these methods are anticipative and able to estimate the value of an action by looking ahead. To a lesser extent, real-time methods are also able to do this by, for example, looking one step ahead or via stochastic programming. Real-time methods are generally more suitable for highly dynamic environments compared to compliance tables. On the other hand, compliance tables are static tables that specify predetermined configurations or actions for specific conditions. These tables are designed based on preexisting knowledge or predefined rules and are not updated in real-time.

The last criterion is the *flexibility* criterion. Here again, Monte Carlo methods score the highest since these methods are model-free. ADP methods often rely on specific model knowledge, optimisation methods require well-defined mathematical models and constraints, and compliance tables lack adaptability. These factors limit their flexibility compared to Monte Carlo methods, which can learn and make decisions solely based on interaction with the environment, adapt to unknown systems, estimate various value functions, focus on specific subsets of states, and improve policies without a complete model.

Summarized Monte Carlo methods score the highest on four out of five criteria, although they require more computational power, than, for instance, ADP. However, the maximum size of the possible problem instances for MDS is limited. The modularity and suitability for dynamic environments make these methods attractive to use.

7 Research Proposal

This chapter discusses the research proposal. First, section 7.1 discusses the research gap that became apparent during the literature study. Then, section 7.2 and section 7.3 define the research objective and research questions. Next, section 7.4 defines the project further by providing assumptions, requirements, the scope and constraints. Finally, section 7.5 elaborates on the approach and planning of the project.

7.1. Research Gap

Developing efficient algorithms that can be applied to large-scale pickup and delivery networks is becoming more and more relevant in today's fast-paced economy. A strategy that is able to handle the envisioned scale and complexity of the MDS network does not exist yet. Such a strategy would partially fulfill the need for scalable algorithms that can optimise allocation while considering various factors, such as traffic conditions, vehicle capacities, time windows, and dynamic changes in the network in a reasonable timeframe.

Next to that incorporating real-time adaptability into such strategies is an important research direction. Most algorithms for dynamic pickup and delivery problems are formulated and tested in a small static environment and then applied to dynamic environments afterwards. As a result, these algorithms do not take advantage of methods that improve real-time adaptability such as repositioning or request buffering.

While research has already progressed a lot in the area of ground vehicles, strategies for complex networks incorporating UAVs are still lacking. While van Haasteren [85] built a UAV model based on the MDS project. Much of his efforts focused on model development itself and his analysis was on a much smaller scale. Hence, there is an opportunity to focus on resource management through novel allocation and repositioning strategies for a large-scale network. Doing so should add to understanding the impact, potential benefits and challenges, of incorporating UAVs in such networks.

7.2. Research Objective

The research objective is formulated as the following:

'To design and evaluate a strategy for allocating and repositioning a fleet of UAVs with the goal of minimising operational costs in the context of large-scale dynamic pickup and delivery tasks with deadlines for medical material'

7.3. Research Questions

In order to achieve the previously stated objective, several research questions are identified. These are listed below:

1. How can the allocation and scheduling of UAVs be designed to minimise operational costs of a UAV fleet in large-scale dynamic pickup and delivery networks?
   - What do requests to be allocated look like?
   - What does the environment that the UAVs operate in look like?
   - What assumptions are present?

- How should uncertainty surrounding different variables be dealt with?
- What constraints are present during allocation?
- What technique should be implemented? *From literature study it becomes clear that this technique is adapted TeSSI.*
- What bidding rule should be used?
- How should the cost of bid be computed?
- What is the mathematical optimisation problem that has to be solved for scheduling?
- What method should be used to solve the optimisation problem?
- What steps can be undertaken to ensure the speed of the allocation procedure?
- How to implement reallocation of tasks as proposed in pTeSSI?
- What is the relation between cost minimisation and other key performance indicators such as timeliness or emissions?
- How does replanning tasks affect operational cost?
- What can be said about the scalability of the implemented strategy?

2. How does repositioning UAVs affect the operational costs of an allocation strategy in a large-scale dynamic pickup and delivery network?
    - What technique should be implemented? *From literature study it becomes clear that this should Monte Carlo methods in the context of reinforcement learning?*
    - How should the states, actions and rewards of the problem be defined?
    - What techniques within Monte Carlo methods is suitable for the problem at hand?
    - What algorithm using Monte Carlo methods is suitable for implementation?
    - Which Python libraries are out there to support the implementation of Monte Carlo methods?
    - How should the algorithm be trained and tuned?
    - How does repositioning affect other key performance indicators such as timeliness?
    - What is the influence of repositioning on the scalability of the strategy?
    - What UAV moving patterns become apparent?
    - Is the information from moving patterns useful to making strategic decisions such as placing UAV hubs?

7.4. Project Definition

7.4.1. Assumptions

1. Each location is able to swap the battery of a drone.
2. The battery of the drone is swapped each time it arrives at a new destination.
3. The control center has full knowledge of the schedules and positions of each of the UAVs in the network.
4. The quality of communication between the control center and the UAV agents does not hinder operations.
5. Maintenance is considered out of scope, since it is only necessary every 200 flight hours.
6. A binary variable is used to reflect whether or not a UAV is allowed perform an order. This variable will reflect the effects of extreme weather conditions and commands by air traffic control.
7. The generation of routes will be based on the work performed by van Haasteren [85], these routes are assumed to be conflict-free.
8. Whenever it is impossible perform a delivery using UAVs, a ground vehicle will carry out the delivery. The safety net that accommodates these type of deliveries is assumed to always be available, but is subject to current traffic conditions.
9. Docking stations are neglected.

7.4.2. Requirements

1. The strategy must allocate each request to either a UAV or a ground vehicle.
2. The strategy must be able to generate a schedule from a set of allocated tasks to UAV.
3. The strategy must be able to reposition UAVs.
4. The strategy must allow for replanning of allocated tasks.
5. The strategy must be able to cope with instances in which UAV flight is prohibited.
6. The strategy must be able to cope with UAV failure.
7. Each order must be completed by the MDS network within its deadline.
8. The range of a UAV for a single leg from location A to location B cannot be exceeded.
9. The combined mass and volume of orders cannot exceed the mass and volumetric capacity of the UAV.
10. The strategy must be capable of efficiently managing a client set with a size on the order of thousands.
11. The model architecture must be modular.

7.4.3. Scope

Figure 7.1 illustrates the proposed process that will be used to address the research questions. This process will be implemented in Python through agent-based simulation, following the approach implemented by van Haasteren [85]. Utilising this modeling technique offers several advantages, such as the ability to accurately capture system complexity, facilitate modularity, and enable analysis of both system-wide and local behaviors.

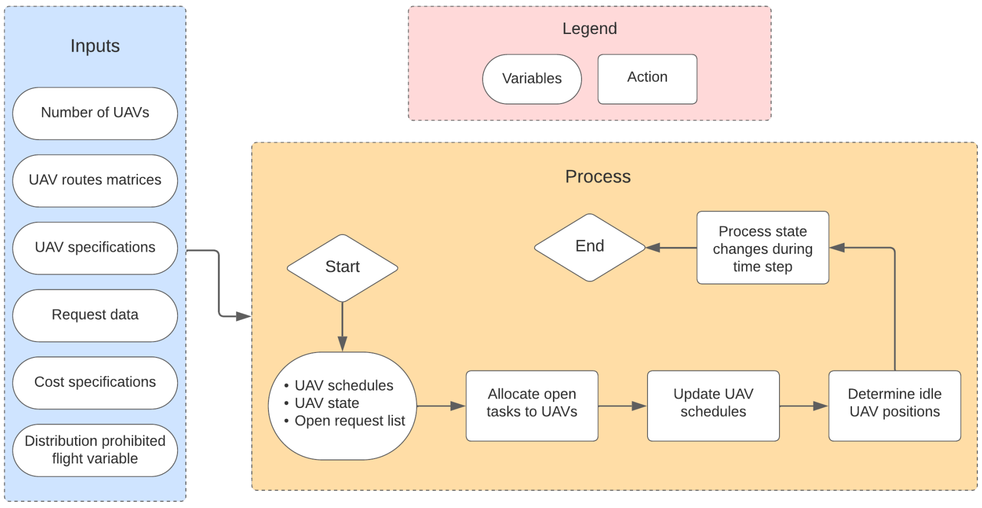


Figure 7.1: Structure of envisioned process.

The model requires the incorporation of various inputs, consisting of both simple and more complex elements. Simple inputs include UAV specifications, fleet size, and operational cost details. On the other hand, more complex inputs encompass UAV routes matrices, request data, and information regarding prohibited flights.

To generate UAV routes matrices, van Haasteren's work [85], which leverages geographical characteristics, will be used to determine these routes. Moreover, requests for a full day will be pre-generated, modeling potential customers for MDS based on their size. The scope of clients will primarily be limited to hospitals and blood banks due to the availability of publicly accessible information on these institutions.

Additionally, careful consideration will be given to situations in which UAV flights are not possible and how they should be represented in the model. This will be accomplished by introducing a binary variable that indicates that flight is prohibited. The goal is to determine the distribution of this variable in advance.

The process itself requires the implementation of an allocation technique and a repositioning technique. The technique that will be used for allocation is Adapted TeSSI and the technique that is used for repositioning is Monte Carlo methods in the context of reinforcement learning.

Regarding the backup network, tasks are only allocated to ground vehicles in case no UAV is available or when flight is prohibited. There will be a module in place that can determine the effects of such an allocation on cost and on-time performance. However, the ground vehicles themselves are not modelled.

7.4.4. Key Performance Indicators

The quality of the strategy is analysed using several key performance indicators. The list below denotes each of the indicators. For each of these indicators it is possible to analyse their sum or to scale them based on the request input.

- Variable costs (€)
- Fixed costs (€)
- Emissions (kilogram CO^2)
- Time left to task deadline (minutes)
- Share of orders that are on time (%)
- Share of orders allocated to backup network (%)
- Computational time (seconds)

7.5. Approach and Planning

Table 7.1 shows an overview of the identified work packages and their estimated durations. In the following subsections each of these packages is discussed in more detail and deliverables are defined.

It can be noted that work package 1 and 2 have the same deadline. This is due to the fact that both of these work packages require a lot information provided by business partners. Therefore, the work packages are executed in parallel to prevent any unnecessary delays.

Furthermore, the time planned for some tasks may seem excessive. However, this is due to external activities that take approximately 20 to 25 hours a week.

Finally, the mid-term and green light meetings are planned after work package 5 and 6 respectively.

Table 7.1: Overview of identified work packages and their estimated workload.

Work Package Number	**Work Package Name**	**Estimated Duration**	**Approximate Deadline**
1	Demand Generation	2 weeks	23 August ’23
2	Agent-based Simulation	6 weeks	23 August ’23
3	Allocation Algorithm	3.5 weeks	22 September ’23
4	Replanning	2 weeks	6 October ’23
5	Repositioning	5 weeks	10 November ’23
6	Verification, Valiation and Analysis	6 weeks	22 December ’23
7	Finalisation	4 weeks	17 January ’24

7.5.1. Work Package 1: Demand Generation

Description

The purpose of this work package is to be able to create a module that can generate a request schedule for a given timeframe and client base. The first step is to set up a database of potential clients and to determine relations between these clients. Then, a module can be created that can generate a schedule based on this information. The final step is to implement customer agents and requests in the agent-based simulation.

Deliverables

- Module that pre-generates demand
- Python files containing client agents and request objects
- Report on work package

7.5.2. Work Package 2: UAV Operating Environment

Description

This package is concerned with setting up the remainder of the agent-based simulation. The goal is to set up the simulation and implementing the described KPIs so that the model can run using a simple greedy allocation strategy.

Furthermore, UAV routes must be generated for the full set of locations and a decision must be made on how to implement no-fly conditions.

While this package accounts for a lot of work, it is possible to built on the work of van Haasteren [85], therefore decreasing the workload.

Deliverables

- Agent-based simulation of MDS
- File containing all possible UAV routes based on client set
- Report on work package

7.5.3. Work Package 3: Allocation Algorithm

Description

The goal of this package is to implement Adapted TeSSI in the working agent-based simulation. Before this can be done, some theoretical work must be performed. This includes determining the bidding rule, determining constraints and formulating the scheduling problem. The work by Chen [18] can be used as a foundation.

Deliverables

- Allocation and scheduling module implemented in Python
- Report on work package

7.5.4. Work Package 4: Replanning

Description

This work extends the allocation algorithm developed in the previous package, by allowing agents to decommit from already scheduled tasks as described by Rizzo [63].

Deliverables

- Implement replanning in Python.
- Report on work package

7.5.5. Work Package 5: Repositioning

Description

The goal of this work package is to integrate repositioning into the agent-based model. The initial step involves doing theoretical work to determine the specific implementation approach. While it is clear that Monte Carlo methods in reinforcement learning will be utilised, there are still numerous options to consider. These options include selecting between on-policy or off-policy methods and choosing between first-visit or every-visit methods. Additionally, it is necessary to define the system states, actions, and rewards. Then, an appropriate algorithm must be selected for the implementation. Finally, the module must be implemented in Python.

Deliverables

- Repositioning module
- Report on work package

7.5.6. Work Package 6: Verification, Validation and Analysis

Description

At this point, the full model is in place. The first step is to verify the model through extensive debugging, testing, and stressing the model. Then, proper validation methods must be chosen through literature study. Once a proper plan for validation is established, the model can be validated.

Finally, experiments are set up and conducted for analysis. The results of these experiments must be analysed and a sensitivity analysis must be conducted to determine their reliability. **If time allows**, improvements to the model could be implemented to increase computation speed or to increase realism in the model. Finally, an article has to be written about the model and the conclusions that were drawn.

Deliverables

- Draft article
- Report on work package

7.5.7. Work Package 7: Finalisation

Description

This is the final phase of the thesis. At this point, the green light meeting was successful. Hence, the article and the thesis document can be completed and a presentation can be prepared.

Deliverables

- Article
- Presentation

III

Supporting work

1 Verification

This chapter performs a comprehensive verification of our model. We start by investigating the model's response to various parameter variations to confirm its coherence with expectations. Subsequently, we subject the model to stress tests using extreme values to assess its robustness and ensure performance.

1.1. Sensitvity

This section focuses on analyzing the model's sensitivity to parameter variations, aiming to verify its behavior under diverse conditions. Various assumptions are tested, as well as overall sensitivity

1.1.1. On Board Time

One notable feature of the Adapted TeSSI scheduling algorithm is its ability to allow Unmanned Aerial Vehicles (UAVs) to be on board for multiple flight legs. While a constraint is in place to prevent orders from remaining on board for extended periods, it is valuable to gain insights into the actual time spent on board.

The sensitivity analysis presented in Figure 1.1 illustrates how the number of drones in the network affects the average time a UAV spends on board with order. The total number of orders remained constant throughout the 20 simulations per parameter setting. Consequently, a decreasing number of drones indicates a higher workload per drone. As anticipated, the analysis demonstrates that the time spent on board increases when there is a smaller number of drones. This phenomenon is attributed to the higher workload, leading to more complex routes where orders are more likely to be on board for several flight legs.

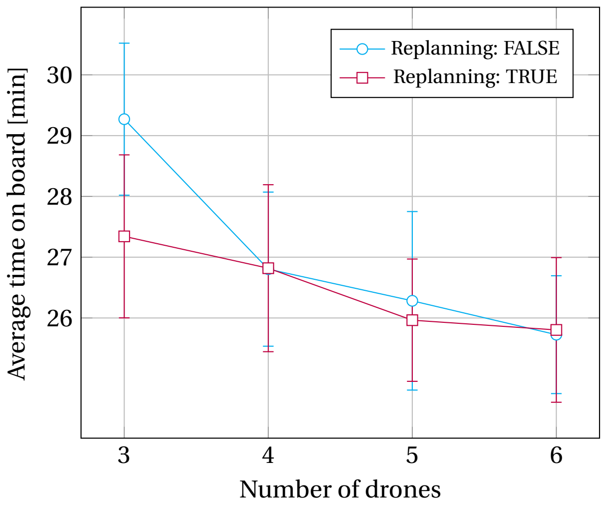


Figure 1.1: Average time spent on board of a UAV by an order for the Adapted TeSSI scheduling algorithm with and without replanning.

Additionally, we investigate the impact of replanning on the efficiency of our system. We anticipate replanning will simplify complex routes by reassigning orders to different UAVs, thus simplifying the overall operation. We employ the Mann-Whitney U test and the Vargha Delaney A measure to substantiate our findings. Detailed results from these statistical tests are presented in Table 1.1.

In this table, the parameter A_{FT} indicates the likelihood of obtaining higher values from the population without replanning compared to the population with replanning. Analyzing Figure 1.1, it becomes apparent that, for scenarios involving 4 to 6 drones, replanning does not significantly impact the time a UAV spends on board. This observation is supported by the p-values provided in Table 1.1, which prevents us from rejecting the null hypothesis that the two populations are identical.

However, when considering a fleet of 3 drones, we can confidently assert that replanning does have a tangible effect on the time spent on board. This conclusion is further supported by both the A- and p-values, signifying a reduction in on-board time when replanning is implemented.

Number of drones	**U-value**	**p-value**	A_{FT}**-value**
3	54	< 0.01	0.865
4	203	0.532	0.508
5	181	0.302	0.548
6	185	0.335	0.463

Table 1.1: Details of Mann-Whitney U test and Vargha Delaney A measure for results shown in Figure 1.1.

1.1.2. Battery Swap

One fundamental assumption within our model is the ability to swap the drone agent's battery at every operational location. This assumption enables drone agents in our system to maintain continuous operation. In this section, we delve into the implications of this assumption by examining the average distance a drone covers during flight. Specifically, we explore whether the assumption would significantly impact our system's performance if not upheldfor instance, if a drone could only exchange batteries after completing two or three flights.

Figure 1.5 illustrates the average flight distance across algorithm configurations and workloads, quantified by the number of drones allocated relative to the demand volume, which is kept constant. Notably, the graph reveals minimal discrepancies among the four algorithm configurations studied. Moreover, the aggregated data showcases an average flight distance of approximately 13.1 kilometers. This distance represents roughly one-seventh of a drone's full operational range. Furthermore, it accounts for approximately one-third of the maximum distance achievable during a single flight leg, 43.2 kilometers. This observation suggests that our model inherently tends to steer operations away from prolonged flight distances, aligning with operational feasibility considerations.

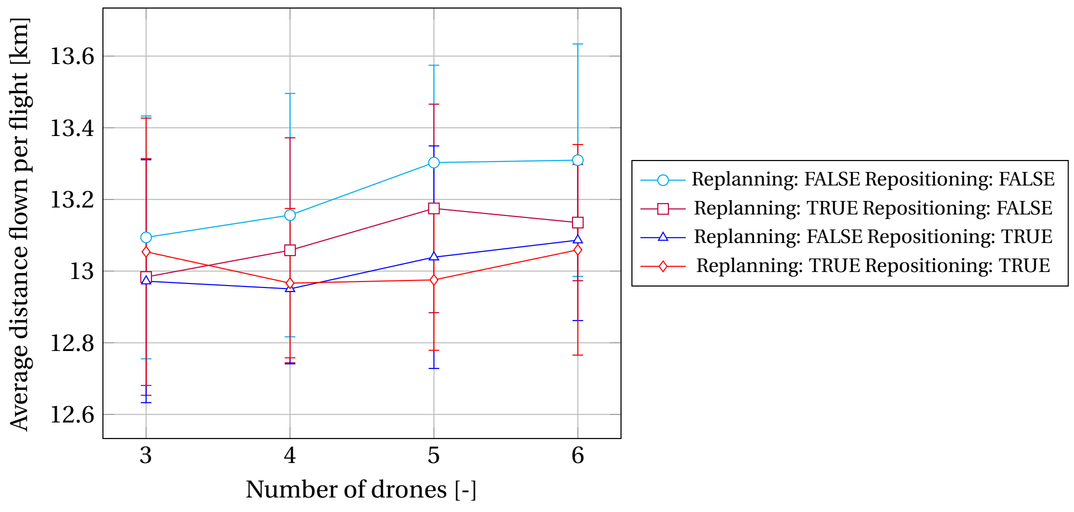


Figure 1.2: Average distance flown for different parameter configurations.

1.1.3. Turnaround Time Length

autoreffig:TAT2, Figure 1.4, and Figure 1.2 illustrate the model's sensitivity to variations in turnaround time. As expected, both the utility (fraction of time with payload on board) and the availability (fraction of orders performed by drone agents) decrease with higher turnaround times.

However, an interesting observation is the decrease in cost per flight with an increase in turnaround time. This phenomenon occurs because a higher turnaround time can lead to a holding pattern, where orders are not executed immediately. Subsequently, when another order arises, reoptimization occurs, allowing the model to benefit from synergies between tasks. This effect becomes less pronounced for higher workloads.

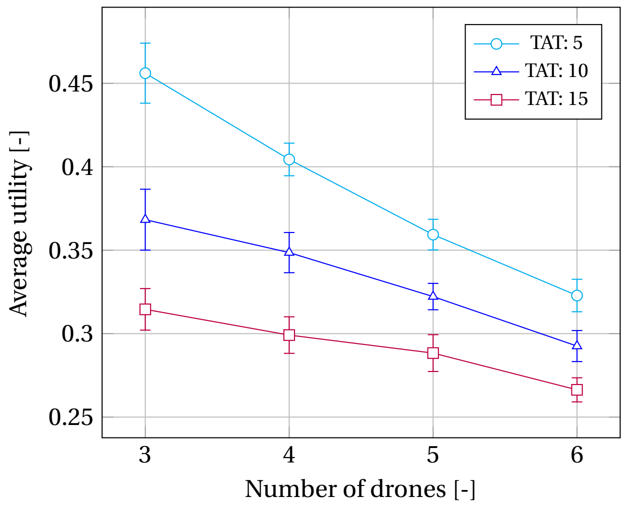


Figure 1.3: Average Utility for turnaround times.

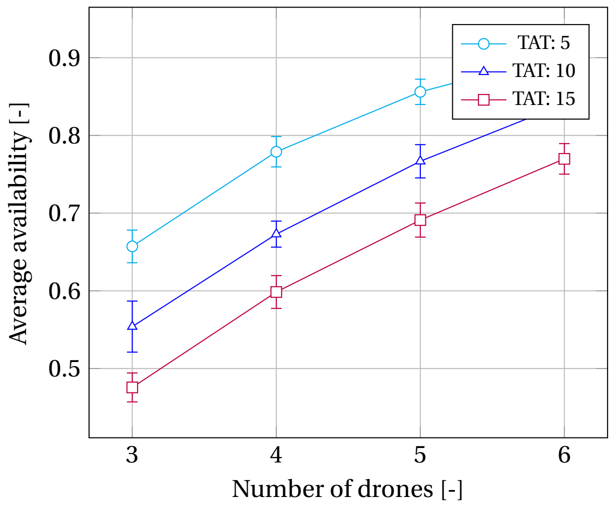


Figure 1.4: Average Availability for turnaround times.

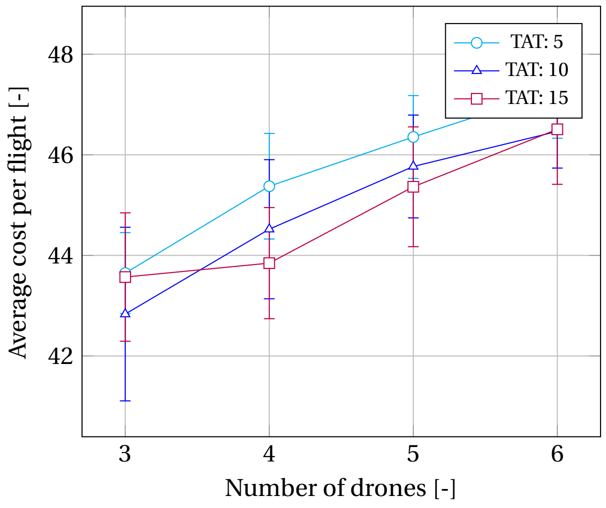


Figure 1.5: Average Value for different turnaround times.

1.1.4. Speed

Figure 1.6 illustrates the model's sensitivity to variations in speed concerning the KPI availability. The plot distinctly demonstrates that availability increases as speed increases.

However, it is notable that this effect diminishes as speeds become increasingly higher, suggesting that there may not be a significant advantage to employing UAV models with higher speeds.

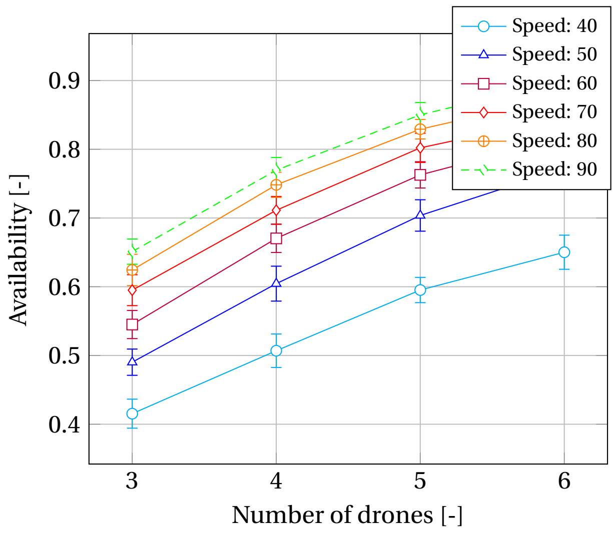


Figure 1.6: Average availability replans for different speeds under different workloads.

1.2. Extreme Values

To comprehensively verify our model, we conduct stress tests involving extreme values for crucial input parameters. Parameters selected for testing encompass the number of drones, turnaround time (TAT), and drone agent speed. While numerous other parameters are subject to variation, their purpose is to provide insights rather than identify anomalous behavior. The subsequent subsections outline the specific tests undertaken and their respective methodologies.

1.2.1. Number of Drones

This section evaluates the model's performance under extreme values of the number of drones, encompassing scenarios with no drones, a single drone, and a surplus of drones.

No Drones at All

Scenario: Only cars are available to fulfill orders.
Results: The model executed without encountering any errors, and the output data aligned with expectations.
Performance: All deliveries were successfully completed using cars.
Conclusion: Drones are not essential for model execution in this scenario.

Single Drone

Scenario: A single drone is employed for simulation.
Result: The model executed flawlessly without errors, yielding expected output data.
Performance: On average, the drone completed 62 orders per simulation day, and the entire day's simulation was processed within 30 seconds.
Conclusion: The model effectively operates with a singular drone, suggesting that the number of drones need not be plural for model utilization.

Abundance of Drones

Scenario: 100 drones are employed for simulation.
Result: The model executed flawlessly without errors, yielding expected output data.
Performance: The drone agents performed all published requests. However, due to the large number of drones the simulation was significantly slower, about 1.5 minutes per day.
Conclusion: The model can handle large numbers of drone agents, but is slowed down by the larger number of computations that have to be performed.

1.2.2. Turnaround Time

This section assesses the impact of varying turnaround time (TAT) values on the model's performance. The minimum TAT is set to 1 minute to prevent the occurrence of subtours in the scheduling optimization algorithm. Additionally, scheduling constraints dictate that flights cannot start and end at the same timestamp. Besides, it is unrealistic to disregard turnaround time entirely. We further examine the effects of an extended TAT for a configuration using 5 drone agents.

2-hour TAT

Scenario: In this scenario, each drone agent participating in the simulation operates with a specified TAT of 2 hours.
Result: The model executed smoothly without encountering any errors, producing the anticipated output data.
Performance: Drone agents handled a small portion of the requests, primarily focusing on tasks located near their assigned positions.
Conclusion: The model adeptly handles turnaround time considerations, demonstrating consistency in its performance without encountering any anomalies.

1.2.3. Speed

The analysis in subsection 1.1.4 demonstrates the effects of speed for quite a large range already. This part looks into what extrimities speed of 1 km/h and a speed of 200 km/h do with the model. Simulations are just like before performed with the same demand scenario and 5 with 5 drone agents.

1 km/h

Scenario: Each drone agent is restricted to a speed of 1 km/h.
Result: The model faced challenges due to onboard time constraints, as travel time exceeded the maximum allowed time onboard. After temporarily removing these constraints, the model was able to correctly provide information.
Performance: Drone agents were only able to execute same-day delivery orders that required a single flight. These were less than 1 % of the full number of orders.
Conclusion: While the model demonstrated its capabilities at low speeds, other constraints prevented the model from providing information. Evaluating performance at these speeds necessitates a reconsideration of the constraints present.

200 km/h

Scenario: Each drone agent can move at a speed of 200 km/h.
Result: The simulation ran smoothly, producing the anticipated outcomes.
Performance: As indicated in subsection 1.1.4, availability at this speed only marginally increased for the same number of drones. This can be attributed to the timing of order arrivals, which are better addressed by increasing drone quantity rather than speed. Additionally, while variable costs decreased due to shorter flight durations, fixed costs rose as orders were promptly executed upon publishing, rather than being delayed due to capacity limitations.
Conclusion: The model effectively accommodates high UAV speeds, delivering accurate results.

2 Algorithmic Design choices

This appendix provides insights into the decision-making process behind the algorithm's design. First, section 2.1 explores the initiation point for re-optimization of an existing schedule in the Adapted TeSSI algorithm. Then, section 2.2 delves into the details of the reallocation algorithm.

2.1. Adapted TeSSI

We investigate the functionalities of the Adapted TeSSI algorithm, specifically its ability to reoptimize an agent's schedule while it still has unfinished tasks. Unfinished tasks are allocated tasks that have already been picked up but have yet to be delivered. Hence, tasks allocated to the agent but not yet picked up are not considered unfinished tasks.

To analyze the usefulness of this ability, we implement a counter-algorithm that chooses another starting point for its optimization. Where the version that can handle unfinished tasks starts as soon as the current flight is finished, the counter algorithm starts when the current task set of the agent is entirely different from the previous time step. This event indicates a full pickup and delivery round has just been completed. Figure 2.1 and Figure 2.2 visualize the previously described starting points.

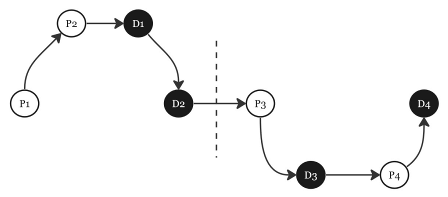


Figure 2.1: Re-optimization starts after a full pickup and delivery loop is finished.

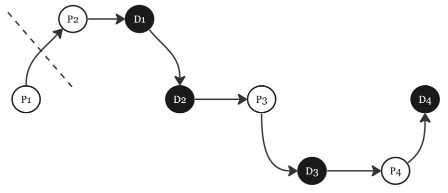


Figure 2.2: Re-optimization starts directly after the current flight is finished.

We evaluate the algorithms using three key performance indicators (KPIs): cost, availability, and utility. To assess cost, we analyze the cost per flight. Availability is measured by determining the proportion of orders that drone agents are capable of executing. As for utility, we consider the fraction of time a drone spends in flight with payload onboard.

Figure 2.3, Figure 2.4, and Figure 2.5 show the respective performance on these KPIs for varying numbers of drones using 60 simulations per parameter setting. During these simulations, three different demand scenarios are tested to ensure that conclusions are generally valid.

Upon reviewing Figure 2.4 and Figure 2.5, it is apparent that the counter-algorithm outperforms the algorithm designed to handle unfinished tasks. Regarding cost implications, as depicted in Figure 2.5, the counter-algorithm demonstrates a notable advantage in operational expenses. Statistical analyses reinforce this observation. Specifically, the Mann-Whitney U test yields p-values consistently below 0.001 across all drone quantity scenarios, rejecting the null hypothesis that the samples are indifferent. Furthermore, the

Vargha-Delaney A measure indicates an average probability of 0.75 that the original algorithm results in a higher cost per flight.

Additionally, Figure 2.4 illustrates that the counter-algorithm facilitates a higher volume of orders per drone. Nevertheless, the difference in performance is marginal. For a statistical significance level set at $\alpha = 0.05$, most computed p-values exceed the threshold, indicating a lack of statistical significance.

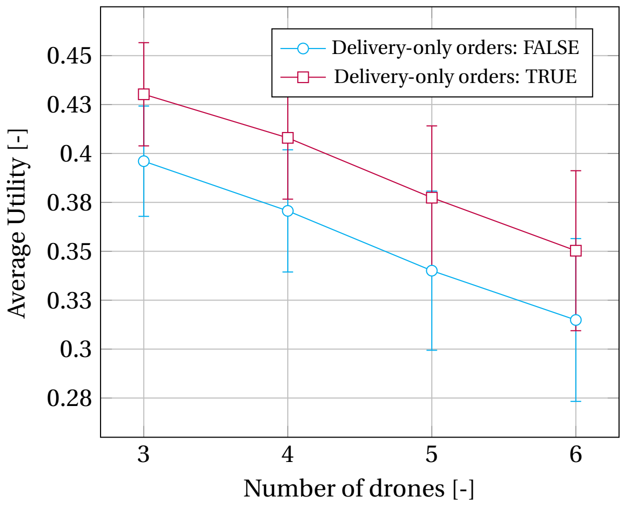


Figure 2.3: Average utility of drone agents for different workloads.

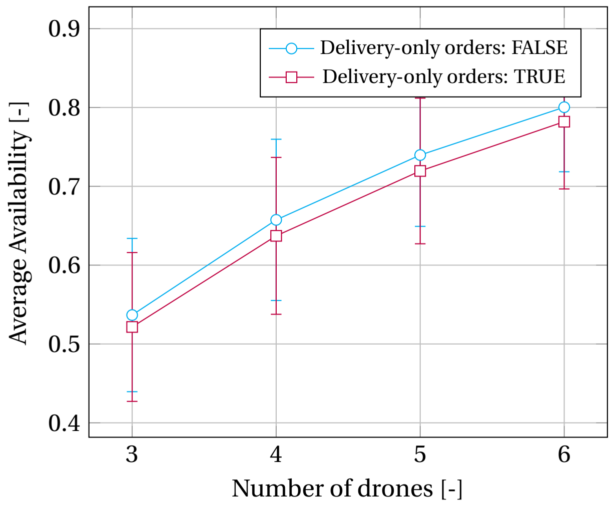


Figure 2.4: Average availability of drone agents for different workloads.

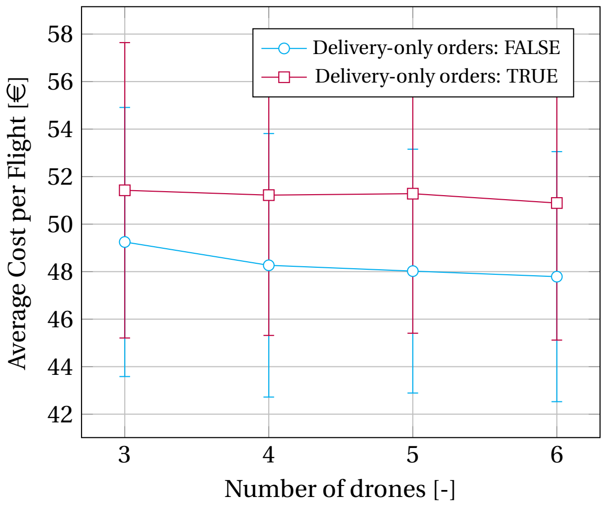


Figure 2.5: Average cost per flight for different workloads.

The results just described go against our expectations. We expected that introducing more planning flexibility would increase our algorithm's performance on the respective KPIs. Figure 2.3 explains this event. The figure shows that our original algorithm does have a higher utility than the counter algorithm. Still, utility is defined as the time spent with payload onboard. Hence, an explanation could be that handling unfinished tasks in schedule optimization results in constraints that require unnecessarily complex routes, therefore deducting the benefits of added flexibility. Furthermore, moving the start of re-optimization further back will delay the execution of the newly published task.

2.2. Reallocation

This section explores the reallocation algorithm, which assesses pending pre-allocated tasks to ensure their assignment to the appropriate agent. Due to the computational demands of reallocation, the algorithm employs a heuristic approach for task selection instead of trying to reallocate every pending task. Specifically, subsection 2.2.1 details the determination of the threshold employed in this heuristic, while subsection 2.2.2 examines the criteria for identifying tasks suitable for reallocation.

2.2.1. Reallocation Threshold

The figures below depict how the algorithm performs across various thresholds and workloads. In Figure 2.6, we observe the impact of thresholds on percentage cost savings. The graph demonstrates a consistent downward trend across all thresholds, suggesting that task reallocation is most effective during high workloads. Additionally, with the exception of the threshold set at 0.5, most thresholds exhibit similar performance. As a result, the threshold set at 0.5 is excluded from further consideration.

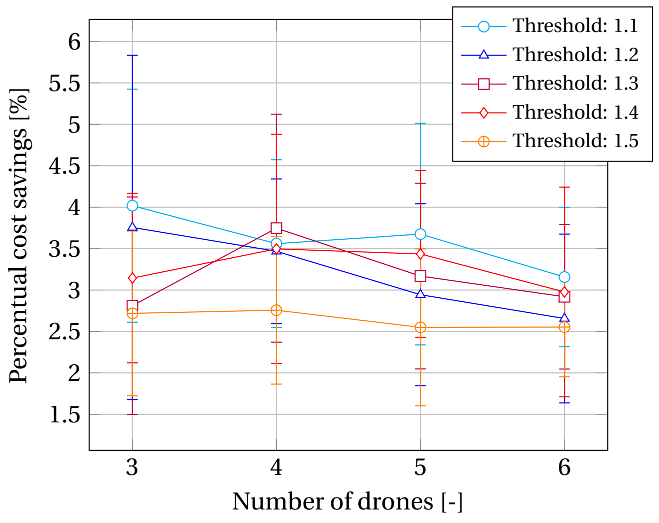


Figure 2.6: Percentual cost gain for different replanning thresholds under different workloads.

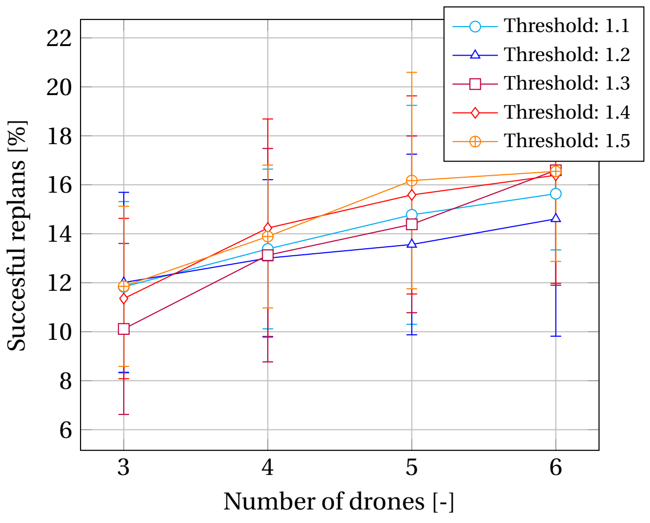


Figure 2.7: Succesful replans for different replanning thresholds under different workloads.

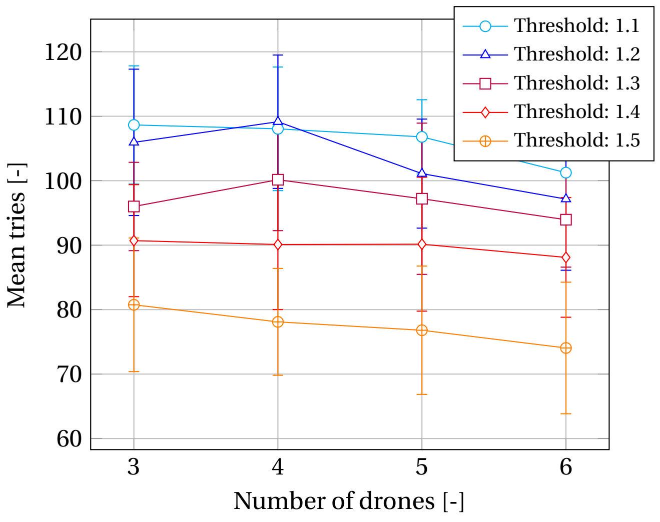


Figure 2.8: Average number of selected tasks for replanning for different thresholds and number of drones.

Furthermore, the graph depicted in Figure 2.7 illustrates the effectiveness of the algorithm, measured by the proportion of tasks selected that were successfully replanned. This graph also reveals a trend: the algorithm's effectiveness increases as the workload diminishes. However, the variation in effectiveness across different thresholds is not particularly pronounced.

Based on the analysis of the two figures, it is evident that there is minimal performance advantage in selecting one threshold over another. However, as previously mentioned, replanning tasks demands significant computational resources. Figure 2.8 demonstrates a noticeable difference in performance between the thresholds, guiding us towards choosing the threshold that minimizes the number of tasks selected for replanning. Given that the threshold of 0.5 is not under consideration, we have established the standard threshold at 0.4.

2.2.2. Type of Reallocated Tasks

Analyzing the chosen threshold, we focus on the specific tasks that are being reallocated. From Figure 2.9, we observe that semi-urgent tasks are predominantly the ones being reassigned. This is due to the fact that urgent tasks have tighter time constraints, making them difficult to reallocate. On the other hand, same-day tasks have the flexibility to be executed at any time during the day, fitting more easily into an agent's schedule. Consequently, semi-urgent tasks, which carry a sense of urgency but are not as tightly bound by time, emerge as suitable candidates for reallocation.

Considering the average fraction of reallocated urgent tasks is zero across all workload settings, we have decided to exclude them from the replanning process altogether to enhance its effectiveness.

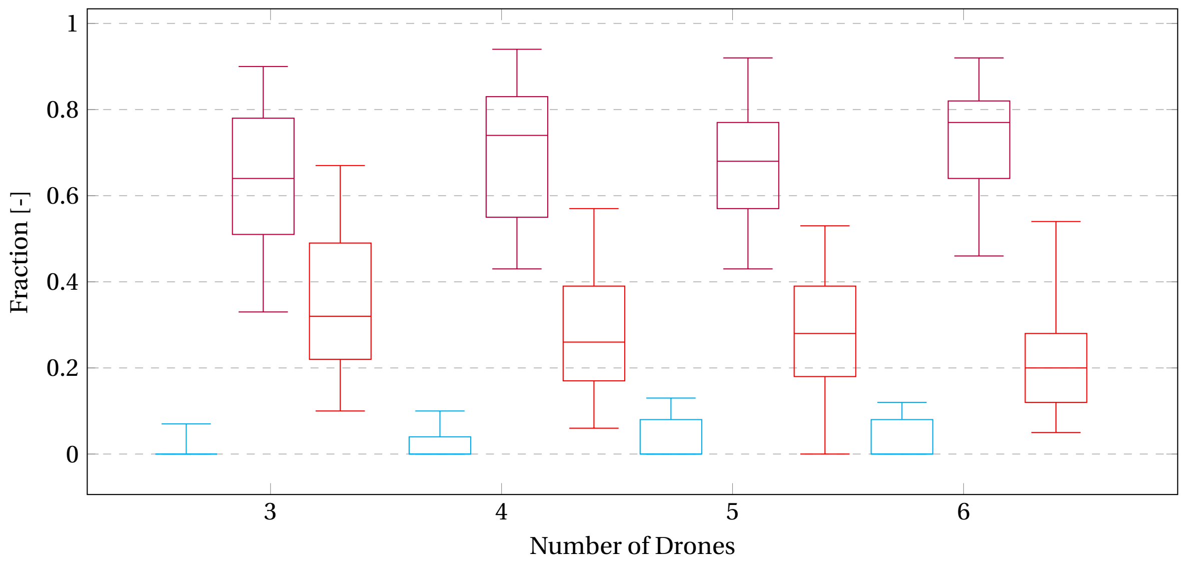


Figure 2.9: Boxplot of composition of reallocated tasks based on urgency levels. The levels displayed are urgent (cyan), semi-urgent (purple) and same-day (red).

3
Determination of Number of Simulations

To establish the number of simulations, we ensured a stable coefficient of variation across the most variable parameter setting for each KPI. This chapter summarizes the most variable parameter settings observed in each experiment for three key performance indicators. Furthermore, the evolution of the coefficient of variation for these settings is displayed.

3.1. Experiment A: Demand Spread

Table 3.1: Overview of most variable parameter setting for each KPI

KPI	Number of drones	Demand spread	Reallocation	Repositioning
Cost	22	Low	True	True
Utility	18	Low	False	True
Delivery success rate	26	Low	False	False

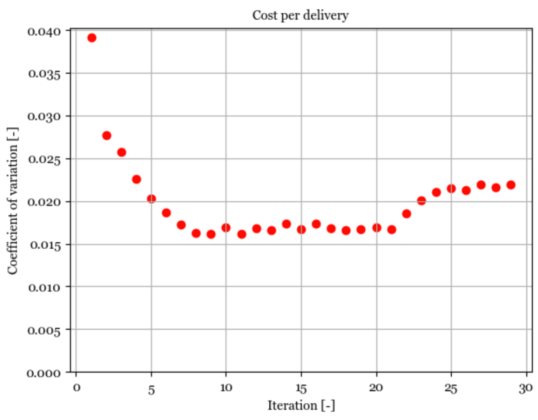


Figure 3.1: Evolution of coefficient of variation for cost per delivery KPI.

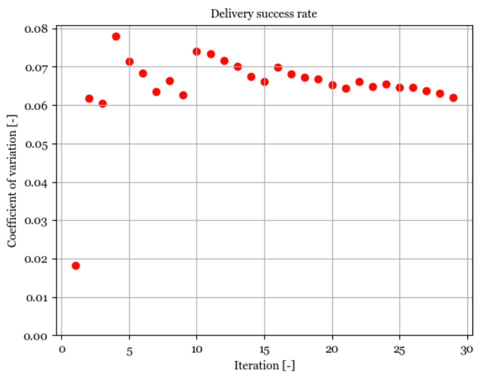


Figure 3.2: Evolution of coefficient of variation for delivery success rate KPI.

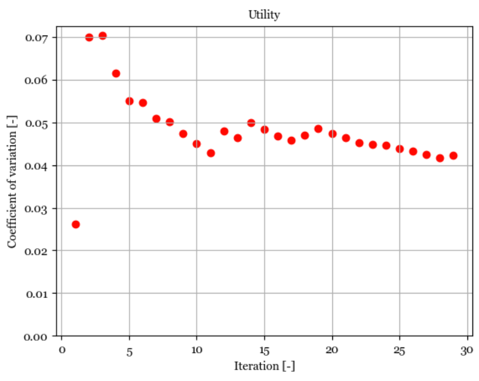


Figure 3.3: Evolution of coefficient of variation for utility KPI.

3.2. Experiment B: Urgency Levels

Table 3.2: Overview of most variable parameter setting for each KPI

KPI	Number of drones	Fraction urgent orders	Reallocation	Repositioning
Cost	28	80%	True	True
Utility	20	80%	False	False
Delivery success rate	28	80%	False	False

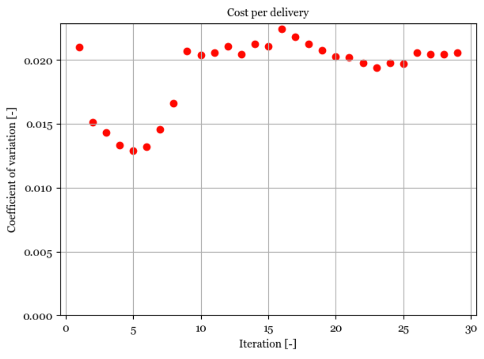


Figure 3.4: Evolution of coefficient of variation for cost per delivery KPI.

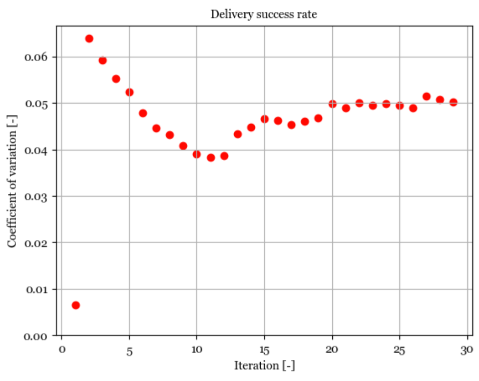


Figure 3.5: Evolution of coefficient of variation for delivery success rate KPI.

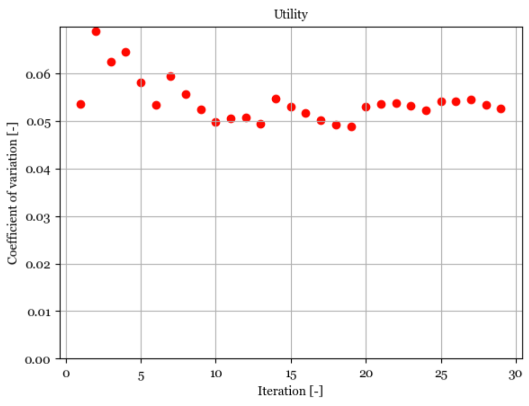


Figure 3.6: Evolution of coefficient of variation for utility KPI.

3.3. Experiment C: Hybrid Fleet

Table 3.3: Overview of most variable parameter setting for each KPI

KPI	Number of drones	Fraction short range drones	Fraction long range drones
Cost per delivery	26	0.2	0.8
Utility	26	0	1
Delivery success rate	26	0.2	0.8

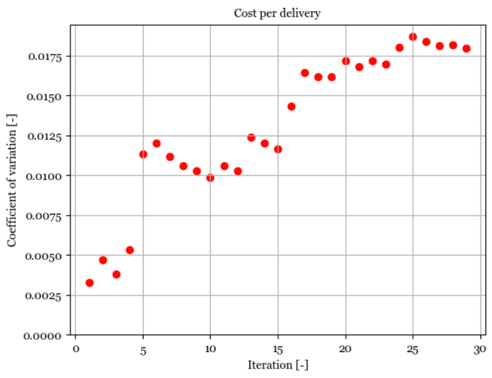


Figure 3.7: Evolution of coefficient of variation for cost per delivery KPI.

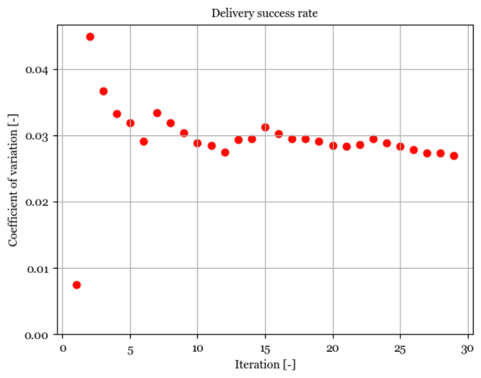


Figure 3.8: Evolution of coefficient of variation for delivery success rate KPI.

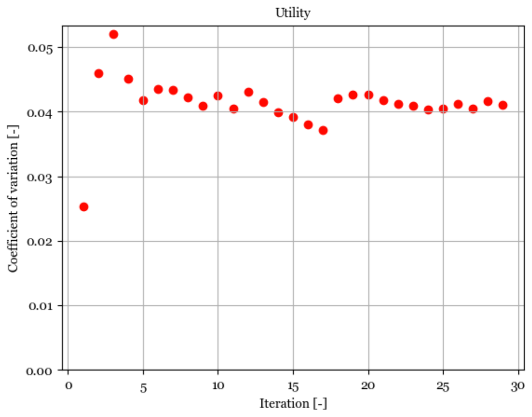


Figure 3.9: Evolution of coefficient of variation for utility KPI.

Bibliography

[1] NZa-Magazines 04. Stand van de zorg 2022. https://magazines.nza.nl/nza-magazines/2022/04/3-trends-in-sectoren, October 2022. Visited on 3-4-23.

[2] C. Agrali and S. Lee. The multi-depot pickup and delivery problem with capacitated electric vehicles, transfers, and time windows. *Computers & Industrial Engineering*, 179, May 2023.

[3] ANWB. Testvluchten medische transport per drone van start. https://www.anwb.nl/nieuws/maa/testvluchten-medisch-transport-per-drone-van-start, November 2020. Visited on 3-4-23.

[4] ANWB. We staan vaker stil dan voor corona. https://www.anwb.nl/verkeer/nieuws/nederland/2023/april/filezwaarte-eerste-kwartaal-2023, April 2023. Visited on 3-4-23.

[5] J. Aurambout, K. Gkoumas, and B. Ciuffo. A drone hop from the local shop? where could drone delivery as a service happen in europe and the usa, and how many people could benefit from it? *Transportation Research Interdisciplinary Perspectives*, 16(100708), December 2022.

[6] Avy. The launch of a new aera. https://avy.eu/stories/the-launch-of-a-new-aera/, November 2021. Visited on 5-4-23.

[7] Avy. Avy aera 3. https://avy.eu/our-integrated-solution/new-aera/, 2021. Visited on 5-4-23.

[8] Avy. Avy dock. https://avy.eu/our-integrated-solution/avy-dock/, 2021. Visited on 5-4-23.

[9] Avy. Avy medkit. https://avy.eu/our-integrated-solution/avy-medkit/, 2021. Visited on 5-4-23.

[10] M. Ayamga, S. Akaba, and A. Apotele Nyaaba. Multifaceted applicability of drones: A review. *Technological Forecasting and Social Change*, 167(120677), June 2021.

[11] D. Banik, N. Ullah Ibne Hossain, K. Govindan, F. Nur, and K. Babski-Reeves. A decision support model for selecting unmanned aerial vehicle for medical supplies: context of covid-19 pandemic. *The International Journal of Logistics Management*, 34(2):473–496, March 2022.

[12] G. Berbeglia, J. Cordeau, and G. Laporte. Dynamic pickup and delivery problems. *European Journal of Operational Research*, 202(1):8–15, April 2010.

[13] G. Binetti, D. Naso, and B. Turchiano. Decentralized task allocation for heterogeneous agent systems with constraints on agent capacity and critical tasks. In *Proceedings of the 2012 IEEE International Conference on Robotics and Biomimetics*, pages 1627–1632, 2012.

[14] R. Boere. Drone vliegt straks bloed en medicijnen naar ziekenhuis. https://www.parool.nl/amsterdam/drone-vliegt-straks-bloed-en-medicijnen-naar-ziekenhuis b6e3442d/?referrer=https://medicaldroneservice.nl/, March 2022. Visited on 3-4-23.

[15] F. Borghetti, C. Caballini, A. Carboni, G. Grossato, R. Maja, and B. Barabino. The use of drones for last-mile delivery: A numerical case study in milan, italy. *Sustainability*, 14(3), February 2022.

[16] V. Bélanger, A. Ruiz, and P. Soriano. Recent optimization models and trends in location, relocation, and dispatching of emergency medical vehicles. *European Journal of Operational Research*, 272:1–23, 2019.

[17] G. Campuzano, E. Lalla-Ruiz, and M. Mes. *Computational Logistics*, volume 13557 of *Lecture Notes in Computer Science*, chapter The Dynamic Drone Scheduling Delivery Problem, pages 260–274. Springer, September 2022.

[18] S. Chen. Multi-agent planning and coordination for automated aircraft ground handling. Master's thesis, Delft University of Technology, August 2022.

[19] T. Chen, B. Zhang, H. Pourbabak, A. Kavousi-Fard, and W. Su. Optimal routing and charging of an electric vehicle fleet for high-efficiency dynamic transit systems. *IEEE Transactions on Smart Grid*, 9(4):3563–3572, July 2018.

[20] X. Chen, M. Ulmer, and B. Thomas. Deep q-learning for same-day delviery with vehicles and drones. *European Journal of Operational Research*, 298(3):939–952, May 2022.

[21] H. Choi, L. Brunet, and J. P. How. Consensus-based decentralized auctions for robust task allocation. *IEEE Transactions on Robotics*, 24(4):912–926, August 2009.

[22] S. Hoon Chung, B. Sah, and J. Lee. Optimization for drone and drone-truck combined operations: A review of the state of the art and future directions. *Computers & Operations Research*, 123(105004), November 2020.

[23] B. Coelho, V. Coelho, I. Coelho, L. Ochi, R. Haghnazar, D. Zuidema, M. Lima, and A. da Costa. A multi-objective green uav routing problem. *Computers and Operations Research*, 88:306–315, April 2017.

[24] P. Cohn, A. Green, M. Langstaff, and M. Roller. Commercial drones are hear: The future of unmanned aerial sytems. https://www.mckinsey.com/industries/travel-logistics-and-infrastructure/our-insights/commercial-drones-are-here-the-future-of-unmanned-aerial-systems/, December 2017. Visited on 17-3-23.

[25] European Commission. Communication from the commission to the european parliament, the council, the european economic and social committee and the committee of the regions ’a drone strategy 2.0 for a smart and sustainable unmanned aircraft eco-system in europe’. https://transport.ec.europa.eu/system/files/2022-11/COM_2022_652_drone_strategy_2.0.pdf, November 2022. Visited on 19-3-23.

[26] A. Cornell, B. Kloss, D. Presser, and R. Riedel. Drones take to the sky, potentially disrupting last-mile delivery. https://www.mckinsey.com/industries/aerospace-and-defense/our-insights/future-air-mobility-blog/drones-take-to-the-sky-potentially-disrupting-last-mile-delivery, January 2023.

[27] W. de Jager. Anwb en postnl willen donorbloed en medicatie gaan transporteren met drones. https://www.dronewatch.nl/2019/11/21/anwb-en-postnl-willen-donorbloed-en-medicatie-gaan-transporteren-met-drones/, November 2019. Visited on 3-4-23.

[28] M. Dell’Amico, R. Montemanni, and S. Novellani. Matheuristic algorithms for the parallel drone scheduling traveling salesman problem. *Annals of Operations Research*, 289:211–226, March 2020.

[29] E. Demir, A. Syntetos, and T. van Woensel. Last mile logistics: Research trends and needs. *IMA Journal of Management Mathematics*, 33:549–561, June 2022.

[30] Maurits Dogterom. Medical drone service. Powerpoint, March 2023.

[31] C. Dong, A. Akram Dan Andersson, P. Arnäs, and G. Stefansson. The impact of emerging and disruptive technologies on freight transportation in the digital era: current state and future trends. *The International Journal of Logistics Management*, 32(2):386–412, January 2021.

[32] K. Dorling, J. Heinrichs, G. Messier, and S. Magierowski. Vehicle routing problems for drone delivery. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 47(1):70–85, January 2017.

[33] J. Eun, B. Duk Song, S. Lee, and D. Lim. Mathematical investigation on the sustainability of uav logistics. *Sustainability*, 11(21), October 2019.

[34] B. Gerkey and M. Matari. A formal analysis and taxonomy of task allocation in multi-robot systems. *The International Journal of Robotics Reserach*, 23(9):939–954, September 2004.

[35] J. Gómex-Lagos, B. Rojas-Espinoza, and A. Candia-Véjar. On a pickup to delivery drone routing problem: Models and algorithms. *Computers & Industrial Engineering*, 172(Part B), October 2022.

[36] M. Hamid, M. Mahdi Nasiri, and M. Rabbani. A mixed closed-open multi-depot routing and scheduling problem for homemade meal delivery incorporating drone and crowd-sourced fleet: A self-adaptive hyper-heuristic approach. *Engineering Application of Artificial Intelligence*, 120, April 2023.

[37] M. Hassanalian and A. Abdelkefi. Classifications, applications, and design challenges of drones: A review. *Progress in Aerospace Sciences*, 91(0376-0421):99–131, May 2017.

[38] B. Heap and M. Pagnucco. Repeated sequential single-cluster auctions with dynamic tasks for multi-robot task allocation with pickup and delivery. In M. Klusch, M. Thimm, and M. Paprzycki, editors, *Multiagent System Technologies*, pages 87–100, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg. ISBN 978-3-642-40776-5.

[39] F. Hildebrandt, B. Thomas, and M. Ulmer. Opportunities for reinforcement learning in stochastic dynamic vehicle routing. *Computers & Operations Research*, 150, February 2023.

[40] H. Huang, C. Hu, J. Zhu, M. Wu, and R. Malekian. Stochastic task scheduling in uav-based intelligent on-demand meal delivery system. *IEEE Transactions on Intelligent Transportation Systems*, 23(8):13040–13054, August 2022.

[41] L. Johnson, S. Ponda, H. Choi, and J. How. Improving the efficiency of a decentralized tasking algorithm for uav teams with asynchronous communication. In *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, August 2010.

[42] B. Kiers. Ernst kuipers gelooft heilig in samenwerking en concentratie. https://www.zorgvisie.nl/ernst-kuipers-gelooft-heilig-in-samenwerking-en-concentratie/, January 2022. Visited on 3-4-23.

[43] S. Koenig, P. Keskinocak, and C. Tovey. Progress on agent coordiantion with coopeative auctions. In *Proceedings of the National Conference on Artificial Intelligence*, volume 3, January 2010.

[44] G. Korsah, A. Stentz, and M. Dias. A comprehensive taxonomy for multi-robot task allocation. *The International Journal of Robotics Research*, 32(12):1495–1512, 2013.

[45] Y. Li, M. Liu, and D. Jiang. Application of unmanned aerial vehicles in logistics: A literature review. *Sustainability*, 14(21), November 2022.

[46] Y. Liu. An optimization-driven dynamic vehicle routing algorithm for on-demand meal delivery using drones. *Computers & Operations Research*, 111:1–20, 2019.

[47] G. Macrina, L. Di Puglia Pugliese, F. Guerriero, and G. Laporte. Drone-aided routing: A literature review. *Transportation Research Part C: Emerging Technologies*, 120(102762), November 2020.

[48] Matternet. Matternet launches drone delivery operations at labor berlin in germany. https://mttr.net/images/Matternet_Press_Release-Labor_Berlin_Drone_Program-2020.11.17.pdf, November 2020. Visited on 27-3-23.

[49] A. Mormont. Ambulance drone. https://www.tudelft.nl/io/onderzoek/research-labs/applied-labs/ambulance-drone, Unknown. Visited on 27-3-23.

[50] M. Moshref-Javadi and M. Winkenbach. Applications and research avenues for drone-based models in logistics: A classification and review. *Expert Systems with Applications*, 177(114854), September 2021.

[51] A. Nasrollahzadeh, A. Khademi, and M. Mayorga. Real-time ambulance dispatching and relocation. *Manufacturing & Service Operations Management*, 20(3):467–480, April 2018.

[52] M. Nguyen, G. Thi-Huong Dan, M. Hoàng Hà, and M. Pham. The min-cost parallel drone scheduling vehicle routing problem. *European Journal of Operational Research*, 299(3):910–930, June 2022.

[53] M. Nisingizwe, P. Ndishimye, K. Swaibu, L. Nshimiyimana, P. Karame, V. Dushimiyimana, J. Musabyimana, C. Musanabaganwa, S. Nsanzimana, and M. Law. Effect of unmanned aerial vehicle (drone) delivery on blood product delivery time and wastage in rwanda: a retrospective, cross-sectional study and time series analysis. *The Lancet*, 10(4):564–569, April 2022.

[54] E. Nunes and M. Gini. Multi-robot auctions for allocation of tasks with temporal constraints. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 29, pages 2110–2116, February 2015.

[55] A. Apotele Nyaaba and M. Ayamga. Intricacies of medical drones in healthcare delivery: Implications for africa. *Technology in Society*, 66(101624), August 2021.

[56] Abu Dhabi Media Office. Abu dhabi to use drone technology for medical supply transfer and delivery. https://www.mediaoffice.abudhabi/en/health/abu-dhabi-to-use-drone-technology-for-medical-supply-transfer-and-delivery/, September 2021. Visited on 27-3-23.

[57] International Civil Aviation Organisation. *ICAO Cir 328, Unmanned Aircraft Systems (UAS)*. Number 978-92-9231-751-5 in ISBN. INTERNATIONAL CIVIL AVIATION ORGANIZATION, 999 University Street, Montréal, Quebec, Canada H3C 5H7, 2011.

[58] A. Otto, N. Agatz, J. Campbell, B. Golden, and E. Pesch. Optimization approaches for civil applications of unmanned aerial vehicles (uavs) or aerial drones: A survey. *Networks*, 72(4):411–458, March 2018.

[59] S. Ponda, J. Redding, H. Choi, J.P. How, M. Vavrina, and J. Vian. Decentralized planning for complex missions with dynamic communication constraints. In *Proceedings of the 2010 American Control Conference*, pages 3998–4003, 2010.

[60] Swiss Post. Swiss post drone transport in the healthcare sector. https://www.post.ch/en/about-us/innovation/innovations-in-development/drones?shortcut=opp-en-about-us-company-innovation-swiss-post-s-innovations-for-you-drones, 2023. Visited on 26-3-2023.

[61] W. Powell. *Approximate Dynamic Programming.* Wiley, second edition, 2011.

[62] B. Rao, A. Goutham Gopi, and R. Maione. The societal impact of commercial drones. *Technology in Society*, 45:83–90, May 2016.

[63] P. Rizzo. Auction-based task allocation for online pickup and delivery problems with time constraints and uncertainty. Master's thesis, Delft University of Technology, February 2021.

[64] H. Sabino, R. V.S. Almeida, L. Baptista de Moraes, W. Paschoal da Silva, R. Guerra, C. Malcher, D. Passos, and F. G.O. Passos. A systematic literature review on the main factors for public acceptance of drones. *Technology in Society*, 71(102097), November 2022.

[65] R. Saleu, L. Deroussi, D. Feillet, N. Grangeon, and A. Quilliot. An iterative two-step heuristic for the parallel drone scheduling traveling salesman problem. *Networks*, 72(4):407–559, December 2018.

[66] R. Saleu, L. Deroussi, D. Feillet, N. Grangeon, and A. Quilliot. The parallel drone scheduling problem with multiple drones and vehicles. *European Jouranl of Operational Research*, 300(2):571–589, July 2022.

[67] Sanquin. Anwb en postnl testen medical drone voor sanquin en erasmus mc. https://www.overleaf.com/project/6400b78a8f5f4d6197822e91, June 2021. Visited on 3-4-23.

[68] S. Sawadsitang, D. Niyato, P. Siew Tan, P. Wang, and S. Nutanong. Multi-objective optimization for drone delivery. *IEEE 90th Vehicluar Technology Conference*, September 2019.

[69] H. Sayarshad and J. Chow. A scalable non-myopic dynamic dial-a-ride and pricing problem. *Transportation Research Part B*, 81:539–554, November 2015.

[70] V. Schmid. Solving the dynamic ambulance relocation and dispatching problem using approximate dynamic programming. *European Journal of Operational Research*, 2196(3):611–621, June 2012.

[71] Medical Drone Service. Foto's tetvluchten drone. https://medicaldroneservice.nl/nl/fotos-drone/, July 2022. Visited on 3-4-23.

[72] Medical Drone Service. Amu-led: Samen in één luchtruim. https://medicaldroneservice.nl/nl/amu-led-samen-in-een-luchtruim/, September 2022. Visited on 3-4-23.

[73] Medical Drone Service. Over medical drone service. https://medicaldroneservice.nl/nl/over-ons/, 2023. Visited on 2-4-23.

[74] Medical Drone Service. Medical drone service. https://medicaldroneservice.nl/nl/, 2023. Visited on 2-4-23.

[75] Medical Drone Service. Veelgestelde vragen. https://medicaldroneservice.nl/nl/faq/, 2023. Visited on 2-4-23.

[76] Medical Drone Service. Anwb, postnl, lvnl en kpn starten met dronevluchten over digitale snelweg in de lucht. https://medicaldroneservice.nl/nl/anwb-postnl-lvnl-en-kpn-starten-met-dronevluchten-over-digitale-snelweg-in-de-lucht/, February 2023. Visited on 3-4-23.

[77] G. Skaltsis, H. Shin, and A. Tsourdos. A survey of task allocation techniques in mas. In *Proceedings of the 2021 International Conference of Unmanned Aircraft Systems (ICUAS)*, pages 488–497, June 2022.

[78] R. Sutton and A. Barton. *Reinforcement Learning: An Introduction.* The MIT Press, second edition, 2015.

[79] D. Takahashi. Ups uses matternet drones to deliver covid-19 vaccines. https://venturebeat.com/ai/ups-uses-matternet-drones-to-deliver-covid-19-vaccines/, August 2021. Visited on 27-3-2023.

[80] A. Troudi, S. Addouche, S. Dellagi, and A. El Mhamedi. Sizing of the drone delivery fleet considering energy autonomy. *Sustainability*, 10, July 2018.

[81] J. Turner, Q. Meng, G. Schaefer, A. Whitbrook, and A. Soltoggio. Distributed task rescheduling with time constraints for the optimization of total task allocations in a multirbot system. *IEEE Transactions on Cybernetics*, 48(9):2583–2597, September 2018.

[82] M. Ulmer and B. Thomas. Same-day delivery with heterogeneous fleets of drones and vehicles. *Networks*, 72(4):475–505, December 2018.

[83] T. van Barneveld. The minimum expected penalty relocation problem for the computation of compliance tables for ambulance vehicles. *INFORMS Journal on Computing*, 28(2):370–384, April 2016.

[84] T. van Barneveld, S. Bhulai, and R. van der Mei. A dynamic ambulance management model for rural areas. *Health Care Management Science*, 20(2):165–186, June 2017.

[85] J. van Haasteren. Design and analysis of a uav assisted medical emergency delivery system. Master's thesis, Delft University of Technology, June 2022.

[86] Rijksinstituut voor Volksgezondheid en Milieu. Duurzame zorg en preventie. https://www.rivm.nl/over-het-rivm/strategisch-programma-rivm/duurzame-zorg-en-preventie, Unknown. Visited on 3-4-23.

[87] D. Weikert, C. Steup, and S. Mostaghim. Survey on multi-objective task aloocation algorithms for iot networks. *Sensors*, 23(1), 2023.

[88] Zipline. Zipline press kit. https://www.flyzipline.com/press-kit, Unknown. Visited on 27-3-23.