A simulation-based performance evaluation model for decision support on drone location and delivery scheduling

Zabih Ghelichi and Monica Gentili
Department of Industrial Engineering, University of Louisville, Louisville, Kentucky, USA, and
Pitu Mirchandani
School of Computing and Augmented Intelligence, Arizona State University, Tempe, Arizona, USA

Abstract

Purpose – This paper aims to propose a simulation-based performance evaluation model for the drone-based delivery of aid items to disaster-affected areas. The objective of the model is to perform analytical studies, evaluate the performance of drone delivery systems for humanitarian logistics and can support the decision-making on the operational design of the system – on where to locate drone take-off points and on assignment and scheduling of delivery tasks to drones.

Design/methodology/approach – This simulation model captures the dynamics and variabilities of the drone-based delivery system, including demand rates, location of demand points, time-dependent parameters and possible failures of drones' operations. An optimization model integrated with the simulation system can update the optimality of drones' schedules and delivery assignments.

Findings – An extensive set of experiments was performed to evaluate alternative strategies to demonstrate the effectiveness for the proposed optimization/simulation system. In the first set of experiments, the authors use the simulation-based evaluation tool for a case study for Central Florida. The goal of this set of experiments is to show how the proposed system can be used for decision-making and decision-support. The second set of experiments presents a series of numerical studies for a set of randomly generated instances.

Originality/value – The goal is to develop a simulation system that can allow one to evaluate performance of drone-based delivery systems, accounting for the uncertainties through simulations of real-life drone delivery flights. The proposed simulation model captures the variations in different system parameters, including interval of updating the system after receiving new information, *demand parameters*: the demand rate and their spatial distribution (i.e. their locations), *service time parameters*: travel times, setup and loading times, payload drop-off times and repair times and *drone energy level*: battery's energy is impacted and requires battery change/recharging while flying.

Keywords Simulation, Optimization, Delivery drone, Humanitarian logistics

Paper type Research paper

1. Introduction

The uniqueness of the disaster scenarios poses critical challenges to the delivery of aid packages to people trapped in cutoff regions which include the destruction of transportation networks, need for timely delivery, scarcity of human resources and rapid changes of the situation. Unmanned aerial vehicles (UAVs), commonly known as drones, have recently attracted much attention as a promising solution for the delivery of aid items in disaster-affected areas (Otto *et al.*, 2018). Distinct characteristics of drones can offer multiple potentials to address the challenges associated with the timely delivery of aid items in disaster-affected areas. Effectively, drones are aerial vehicles capable of reaching cutoff regions in disaster-affected areas as they are not restricted to land-based transportation networks. Furthermore, drones are deemed as fast, effective and pilotless alternatives to traditional last-mile delivery modes, like trucks, and do not require expensive and sophisticated launching infrastructure (Otto *et al.*, 2018; Rejeb *et al.*, 2021).

Like most real-world systems, drone-based delivery systems involve many sources of variabilities. Undoubtedly, one of the most critical sources of uncertainty in disaster scenarios is the

This work has been supported by the Logistics and Distribution Institute (LoDI) at the University of Louisville. Coauthor Mirchandani was partially supported by the US Department of Homeland Security under Grant Award Number 17STQAC00001-06.

Received 22 April 2023
2 November 2023
15 December 2023
Accepted 6 February 2024

Journal of Humanitarian Logistics and Supply Chain Management
Emerald Publishing Limited [ISSN 2042-6747]
[DOI 10.1108/JHLSCM-04-2023-0036]

information related to the demand, including the number of demand points and their corresponding geographical locations. During the very first few hours after a disaster strike, there is usually not enough information about the number and location of people who may need immediate attention. In such situations, as time goes by, the rescue teams may receive more updates and calls for help from people stuck in the disaster-affected areas. Therefore, the decisions on the scheduling of drone trips need to be frequently updated according to the arrival of new sets of information. Other serious sources of uncertainty originate from the operational performance of drones themselves. Even given an exact distance between a drone platform and a demand point, a drone's flight duration and drop-off time may vary as different fluctuating factors, like wind and temperature, can impact its flight speed (Kim *et al.*, 2018).

To address various factors associated with drone-based delivery of aid items to disaster-affected areas, optimization techniques are widely used in practice and theory. Although optimization models can offer powerful decision-making tools and provide recommendations, they often fail to capture the randomness and dynamics of such a system since the underlying model assumptions usually oversimplify the problem for algorithmic tractability. On the other hand, simulation modeling allows the representation and examination of a system through simulating real-life situations with minimal assumptions (Fu *et al.*, 2015). In a drone-based delivery of aid items in disaster scenarios, the high level of fluidity of the system and variability of the parameters necessitates developing simulation tools to evaluate alternative solutions and strategies and prepare for different future contingencies. For example, a simulation model can evaluate the performance of different routing strategies based on possible enroute weather changes.

Another major application of a simulation-based tool is the opportunity of performing analytical studies to improve solutions obtained from an optimization model. Indeed, optimization models typically fail to adequately capture the dynamics and uncertainties of complex systems; therefore, the optimum solutions derived by these models may not be the best solution when it comes to real-life situation realizations (Fu *et al.*, 2015). Thus, simulation models can be integrated with optimization methods to offer a simulation-optimization tool that can improve the solutions obtained by an optimization model. For instance, as will be done in this paper, neighborhood search algorithms can be integrated with the simulation model to find solutions that outperform the solutions obtained from an optimization model alone.

This paper presents a simulation-based performance evaluation model for the designing of a system for timely delivery of aid packages via a fleet of drones. The goal is to develop a decision support framework that allows one to evaluate performance of drone-based delivery systems, accounting for the uncertainties through simulations of real-life drone delivery flights. In this regard, we leverage a simulation model to predict contingencies, evaluate and improve the drone-based delivery of aid items in humanitarian logistics. The proposed simulation model captures the variations in different system parameters, including *Interval of updating the system* after receiving new information, *Demand parameters*: the demand rate and their spatial distribution (i.e. their locations), *Service time parameters*: travel times, setup and loading times, payload drop-off times and repair times and *Drone energy level:* battery's energy is impacted and requires battery change/recharging while flying.

We can summarize the main contributions of this paper in the following three categories:

1 *Decision support framework*: This research introduces a significant contribution by designing a decision support framework, moving beyond simple optimization or simulation models. We introduce a simulation-based performance evaluation model for effective decision-making in drone location and delivery scheduling. Our innovative approach leverages a simulation model to evaluate and enhance drone-based delivery systems. The flexibility of our framework allows for easy adoption of different strategies, scenarios and optimization algorithms.
2 *Simulation-based performance evaluation tool*: The goal is to design a simulation model to perform analytical studies, support decision-making process, verify the performance and applicability of different solutions, evaluate alternative strategies and predict future contingencies and scenarios based on the current situation. Our underlying simulation model considers the stochasticity induced by the fluidity of demand information and variability in drone operations. In this paper, we study different configurations, scenarios and strategies to showcase the effectiveness and applicability of our research.
3 *Simulation-optimization procedure*: This paper leverages the simulation model to learn from the inner dynamics of the system and improve the decisions made by optimization models beforehand. We introduce a simulation-optimization procedure that aims at improving the drone platform locations using a simple but effective algorithm.

This paper stands out from the literature of drone-based delivery of aid items in humanitarian logistics in different ways. Rather than using merely optimization models, we propose a decision-support tool that brings optimization and simulation models together to improve the decision-making process. In this model, the optimization is embedded within the simulation model and optimizes the delivery lists and routing decisions as the realizations are happening. We assume multiple scenarios, strategies and uncertainty sources in our system (see Section 5). Furthermore, our model presents a generic framework which can easily adopt different predesigned logistics systems, evaluate them and improve them through a simulation-optimization procedure (see Section 6.1).

The remainder of this paper is as follows: Section 2 reviews the most relevant literature on the drone-based delivery of aid items in humanitarian logistics and discusses the original contributions of this paper. In Section 3, we formally describe the problem addressed and elaborate on the scope and assumptions. To provide a more understandable picture of the situational context of our problem and goals, Section 4 provides an illustrative example. Section 5 presents the simulation-based performance evaluation model. Subsection 5.1 elaborates on the simulation model and its components. In Subsection 5.2, we briefly review our timeslot formulation for optimizing the drone scheduling problem. This section also introduces multiple efficient algorithms to solve the drone scheduling problem in a timely manner. Section 6 presents a series of experimental analyses for several instances and a case study of Central Florida. Finally, Section 7 summarizes our findings and suggests some future research directions.

2. Literature review

In this section, we review the relevant literature on the problem of drone-based delivery of aid items in humanitarian logistics and discuss different optimization and simulation models designed to address this problem.

2.1 Delivery drones in practice

Over the past few years, there has been a growing interest in the application of UAVs for the delivery of aid and medical items in both practice and theory. Further, many companies across the world have initiated drone-based delivery projects to provide remote communities with medical supplies. In Africa, DHL developed a drone-based delivery system to provide last-mile delivery services for medical items and lab specimens in regions with poor transportation infrastructures (Kaplan, 2020). In the USA, during the COVID-19 pandemic, Zipline and Matternet were authorized to use drones to deliver medical items, e.g. prescription medications and personal protective equipment, to hospitals and retirement communities (Takahashi, 2021; Vincent, 2021).

The unique potential of drones has also given rise to research efforts on drone-based delivery of aid items in humanitarian logistics. In this regard, operations researchers are developing a variety of platform location optimization models and solution approaches to address the critical aspects dealing with the logistics of a drone-based delivery humanitarian aid. Given the limitations and uniqueness of drone systems and the urgency of emergency scenarios, critical factors include limited coverage range of drones, limited payload capacity of drones, timeliness of delivery, limited resources and capacities and uncertainty in several aspects of the system.

2.2 Delivery drones in operations research

In one major line of research, several researchers studied the possibility of integrating drones' operation with other transportation modes, particularly trucks, to compensate for the limited operational range of drones. In 2015, Murray and Chu (2015) proposed a mathematical formulation for a synchronized truck-drone delivery system where drones perform delivery tasks in tandem with a truck. In that system, a truck supports the drone's operation while the truck acts as a moving depot. Some researchers have continued this line of research by developing optimization models for different combinations of synchronized operations of drones and trucks. Recently, Dayarian *et al.* (2020) proposed a synchronized drone-truck delivery system where the drone supports the truck operation by resupplying its load. For a comprehensive review of truck-drone delivery models, see e.g. Chung *et al.* (2020) and Macrina *et al.* (2020).

In another research line, some researchers focused on developing optimization models for the routing and scheduling problems in drone delivery systems as variants of the traveling salesman problem (TSP) and vehicle routing problem (VRP). Detailed reviews on TSP and VRP variants of drone delivery models are also given in Otto *et al.* (2018) and Macrina *et al.* (2020).

2.3 Delivery drones in humanitarian logistics

In the context of drone-based delivery of humanitarian aid items, variants of facility location problems have been widely used to address locational aspects of such systems. Given the scarcity of resources, limited coverage range of drones and high level of urgency, it is crucial to optimally locate and deploy the available facilities so that all the demand points can be sufficiently covered as fast as possible. In this context, the facility location problem mainly involves finding the optimum number, locations and corresponding service areas for depots (Chowdhury *et al.*, 2017), charging stops (Ghelichi *et al.*, 2021), medical centers (Kim *et al.*, 2017) and drone launching platforms (Gentili *et al.*, 2022).

The limited payload capacity of drones implies that drones cannot carry more than one payload in each trip. In this regard, some studies considered the scheduling and sequencing of separate deliveries for a fleet of drones (Gentili *et al.*, 2022). Recently, Ghelichi *et al.* (2021) proposed a timeslot formulation to optimally locate drone charging stations and concurrently scheduling and sequencing individual drone trips for the delivery of medical items in rural and suburban areas.

A barely addressed critical aspect of a drone-based delivery of aid items is uncertainty. In many disaster scenarios, the lack of information, fluidity of impacting events and instability of the situation induces a high level of uncertainty. Besides the volatility of the situation, a drone's operational performance itself may be affected by different factors, e.g. temperature, which can pose additional uncertainty with respect to the drone flight time and completion of the deliveries. Failing to account for these uncertainties may result in suboptimal or ineffective system response.

In the context of the drone-based delivery of humanitarian aid items, uncertainty is barely studied. Kim *et al.* (2019) developed a chance constraint formulation to address the problem of locating drone facilities while considering uncertain flight distance. Kim *et al.* (2018) developed a robust optimization approach to address the problem of drone scheduling problem under battery duration uncertainty induced by the air temperature. Inspired by demand uncertainty in disaster scenarios, Zhu *et al.* (2022) presented a two-stage robust optimization model to address the location-allocation problem for delivery drones. Recently, Ghelichi *et al.* (2022) introduced a chance-constrained stochastic optimization model to address the problem of timely delivery of aid-items when the set of demand locations is unknown. They developed a multi-stage solution approach that integrates heuristic and nonparametric techniques to solve this delivery problem.

2.4 A discussion on the literature and main contributions

Gauging the literature discussed above, the majority of studies have focused on optimizing drone location and scheduling problems offline in advance. The underlying models mostly fail to adequately capture the dynamics and realizations of uncertainties and random events as they occur (Fu *et al.*, 2015; van Steenbergen and Mes, 2020). As mentioned above, the application of drone delivery drones involves multiple sources of uncertainty, which stem from both drone system operations and the fluidity of disaster scenarios. On the one hand, the information about demands in a disaster scenario, such as the number of demand location, their corresponding location and the demand realization time intervals, is very hard to predict, if not impossible. On the other hand, drones' operational performance can be impacted by such factors. In this regard, a stochastic simulation model that simulates the drone-based delivery system, which includes the dynamics of real-life situations, can be a viable tool to for evaluating and improving the performance of the drone delivery system within a framework of a simulation-optimization approach.

In the context of humanitarian logistics, simulation models have been used in different research studies. To keep the focus of our literature review on the intersection of delivery drones and humanitarian logistics, we skip reviewing these studies. However, interested readers may refer to (D'Uffizi *et al.*, 2015; Krejci, 2015; Mosterman *et al.*, 2014; Paz-Orozco *et al.*, 2023; Yale *et al.*, 2020).

Despite the importance of integration of optimization and simulation models in designing drone-based delivery of aid items in humanitarian logistics, the body of literature on this topic is thin. Interested readers on the intersection of optimization and simulation models may refer to the Handbook of Simulation Optimization (Fu *et al.*, 2015). For the evaluation of drone-based humanitarian logistics systems, van Steenbergen and Mes (2020) proposed a generic simulation framework. Kim and Lim (2020) developed a real-time routing and rerouting model for the flight of delivery drones under uncertain flight times. They dealt with uncertainty in the operational performance of drones in a nonemergency context. Fikar *et al.* (2016) developed a simulation and optimization decision-support tool to analyze last-mile delivery of relief packages in disaster-affected areas via off-road vehicles and drones. They proposed a mixed-integer programming model to minimize the average lead time over all requests and subsequently designed an agent-based simulation and optimization to analyze the problem settings.

Our goal is to design a simulation-based decision-support tool to perform analytical studies, support the decision-making process, verify the performance and applicability of different solutions, evaluate alternative strategies and predict future contingencies and scenarios based on the current situation. Our underlying simulation model considers the stochasticity induced by the fluidity of demand information and variability of drone operations.

We first focus on the design of a simulation-based performance evaluation model for drone-based delivery of aid items in disaster scenarios. Then, we integrate a drone scheduling algorithm into the evaluation model, where the optimization objective is to allocate and sequence an optimum set of trips for each drone so that a measure of total disutility/cost is minimized. We define total disutility as the sum of delivery and waiting time plus a penalty for unserved demands within the planning horizon. For the sake of clarity, we refer to the latter optimization problem as the "scheduling problem" and the set of ordered deliveries assigned to the drones in the fleet as "schedule." To solve the scheduling problem in real time, we develop and compare three heuristic algorithms and study their impact on the overall performance for decision-making.

3. Problem definition

We consider a disaster scenario where a set I of m drone platforms sites are located to provide timely delivery of aid items to disaster-affected areas. Examples of such aid items include water, food and medications. A *drone platform* is a structure hosting one dedicated drone. The drone platforms provide launching and landing mechanisms, charging equipment, loading mechanism and aid items for the drones. For the sake of brevity, we refer to drone platforms as "platforms." Due to the limited payload capacity of drones, we assume each drone can carry one payload at a time.

Each drone starts its trip from its dedicated platform, flies to a demand location, delivers its load and returns to its corresponding platform. We also assume each demand location requires exactly one delivery. We assume that the drone platforms are already located. The optimum location of these m drone platforms can be determined by solving drone location models proposed by Gentili *et al.* (2022) and Ghelichi *et al.* (2022). Below, we elaborate on the situational context that we try to address in this paper. Table 1 shows the notations used in this section.

Due to the fluidity of the situation and the lack of information during the first few hours after a disaster strike, we consider a scenario where initially the set of demand locations are not known, but information about each demand location is received at timestamps t. This set of information includes the number of demand locations and the corresponding coordinates of each demand location. The demand locations appear over time based on a spatial probability map, which gives the probability of demand at each point on the plane. Let $\mathcal{J}_t = \{1,2,\ldots,n_t\}$ be the set of demand locations received by timestamp t, where n_t is the number of demands. We represent by (x_j, y_j) the coordinates of the demand point $j \in \mathcal{J}_t$. At timestamp t, upon receiving the new set of information, the drone scheduling model assigns an ordered set of trips $S_i^t = \{j|z_{ij} > 0\}$ to each drone in the fleet, where z_{ij} is a binary variable that takes value 1 if demand location $j \in \mathcal{J}_t$ is assigned to drone $i \in I$ and 0 otherwise. We denote by $\Omega_t = \cup_{i \in I} S_i^t$ the schedule assigned to the fleet of drones $i \in I$ at timestamp t. To efficiently schedule drone trips, we adopt the drone scheduling model introduced by Ghelichi *et al.* (2022) and compare different efficient algorithms to solve the drone scheduling problem in real time. This model optimally schedules and sequences a set of trips for each drone in the fleet such that a measure of disutility (cost) is minimized (details given in Section 5.2). In this case, the disutility is a function of the delivery and waiting time for each demand location, which is nondecreasing with respect to time.

Let random variable δ be the length of the time interval after which the system receives a set of new information. We refer to random variable δ as the "Update Interval." That is, at every δ interval, the system is updated with a new set of information, i.e. demands and their locations. Subsequently, the task list and schedule of the drones must be updated to include the newly received demands' at this moment. We assume that demands continuously arrive within the time interval $[t, t+\delta]$, and each demand point has a specific arrival time. However, all the demands received within time interval $[t, t+\delta]$ are scheduled concurrently at the end of the interval when we update the system state at $t+\delta$. Thus, there is a waiting time such $w_j \leq \delta$ for each demand point j, which is the time between the moment the

Table 1 Table of notations

Notation	Description
$I = \{1,\ldots,m\}$	Set of located drone platforms
$J_t = \{1, 2,\ldots,n_t\}$	Set of demand locations received by timestamp t
K_t	Set of demands already scheduled but not served by timestamp t
F_t	Set of demand locations served by timestamp t
$S_i^t = \{j \mid z_{ij} > 0\}$	Ordered set of trips assigned to drone i at timestamp t
$z_{ij} \in \{0,1\}$	Equals to 1 if demand $j \in J_t$ is assigned to drone $i \in I$ and otherwise
$\Omega_t = \bigcup_{i \in I} S_i^t$	The schedule assigned to the fleet of drones at timestamp t
n_t	Number of demands received by timestamp t
δ	Random time interval
w_j	Waiting time for each demand point j until it is served; $w_j \leq \delta$
T_{max}, D_{max}	Drone coverage range in terms of time and distance, respectively

Source: Table created by Zabih Ghelichi

demand point j occurs until its delivery is scheduled at timestamp $t + \delta$. At any update time, the *current state* of the system must be identified. The current state shows the set of the current location of each drone and the distance from their corresponding platforms and the status of the demands already scheduled at the previous timestamp. Once the system is updated, the current timestamp will be equal to $t + \delta$.

At any timestamp such t, the status of the demands scheduled at the previous timestamp can be labeled as either *served* or *being served* (i.e. a drone is already flying toward it) or *scheduled* but not served yet. If a demand location is already served or being served, we remove it from the set of demand locations that are to be scheduled or rescheduled at the update time. We let F_t denote the set of all demand locations served by timestamp t. However, if there is any demand scheduled at the previous timestamp, i.e. $j \in \mathcal{J}_{t-\delta}$, such that its delivery is scheduled but not served yet, we combine this demand with the set of newly received demands at timestamp t, i.e. $\mathcal{J}_t$. We let K_t denote the set of demands already scheduled but not served until timestamp t. Therefore, at timestamp t, the set of demands to be scheduled is updated to $\mathcal{J}_t = K_t \bigcup$ *unserved demands in* $\mathcal{J}_t$. Then, our scheduling algorithm again finds an optimum set of trips for each drone and updates Ω_t. We repeat the same process over and over until the end of the planning horizon. Note that our scheduling model needs to account for the fact that some demands in K_t have been waiting since the previous timestamp. Therefore, we incorporate this additional waiting time in the objective function of the scheduling model.

The variations in drone operational times are another source of variability. Due to the variations in the drone velocity, we assume the travel time follows a normal distribution with a known mean and variance (Kim and Lim, 2020); this distribution could be updated when new data is collected. Each drone also requires a setup time for service, loading and battery swapping/charging between two consecutive trips. The setup time of drones and drop-off time at the demand location are assumed to follow a known distribution (discussed in the results section). Drones' operational flight range is limited by the maximum coverage range of T_{max} in terms of time and D_{max} in terms of distance. In the system studied, we assume all the drones are identical. We remark that one of the most prominent strengths of our simulation-based evaluation is that it is relatively straightforward to change most of the operational performance distributions and fleet mixtures to analyze other scenarios.

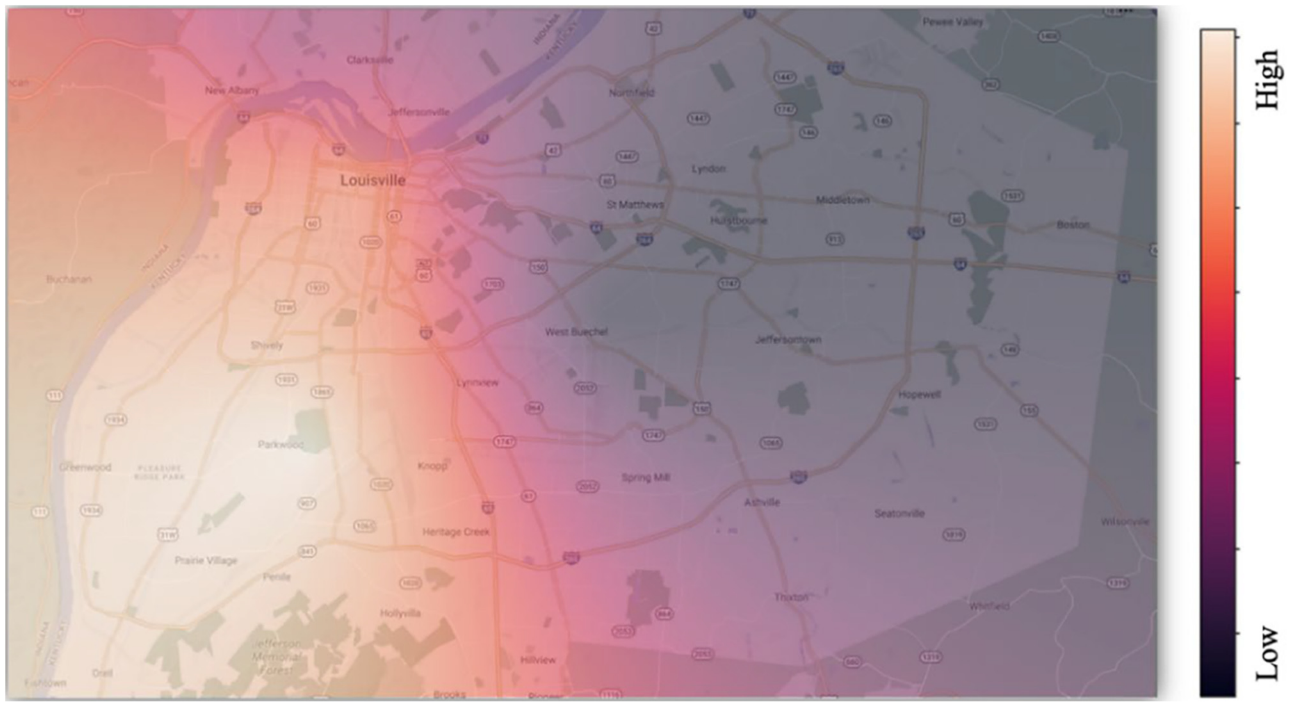
Drone flights may face other nondeterministic effects. Factors such as low temperatures and battery efficiency variations can induce interruptions in the battery performance (Kim *et al.*, 2018). In our simulation model, we assume such factors can impact the energy level of the drone and influence the success of completing some deliveries. To capture these effects, we assume drones may face a battery failure with a known distribution and failure rate. Subsection 5.1 presents two alternative drone routing strategies when they face battery failures.

4. An illustrative example

To provide a clear overview of the associated drone logistics problem and issues of managing the fleet, we present an illustrative example in this section. Assume after a disaster event, two drone platforms are located to dispatch aid packages to demand locations. The optimum location of these platforms can be determined by solving drone location models similar to the ones in Gentili *et al.* (2022). The system is provided with a probability map that gives the chance of having demand at each point throughout the disaster-affected area. We assume demands are arriving based on this probability map. Figure 1 illustrates such a probability map where areas with lighter colors have a higher probability of a request for aid. Although such a probability map can be provided by some agencies that observe such disaster situations, e.g. the trajectory of hurricanes given by the National Hurricane Center (NHC, 2024), the Kriging method can also be implemented to obtain this probability map. For more details about the Kriging technique, the reader may refer to Oliver and Webster (1990).

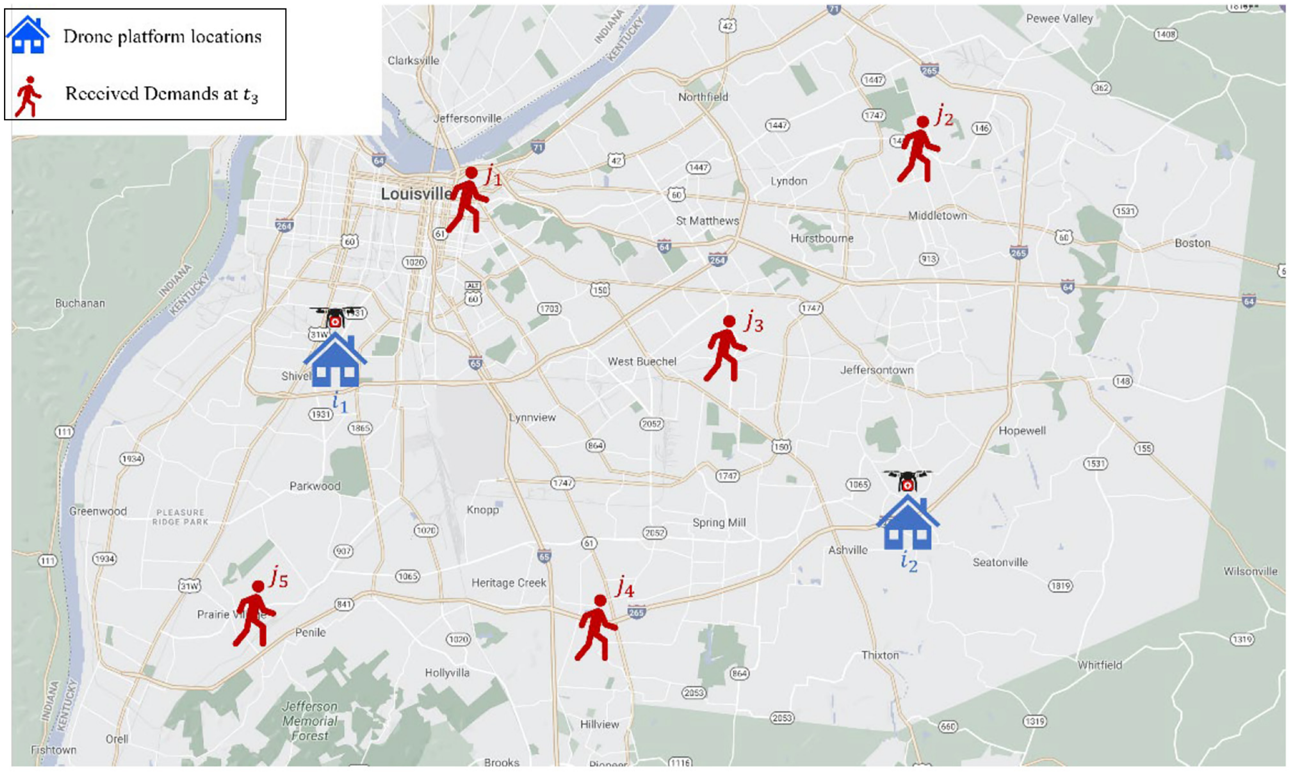
At some timestamp t_1 (refer to Figure 2), a set of new information that includes five demand locations and their corresponding locations (red icons), i.e. $\mathcal{J}_{t_1} = \{j_1, j_2, j_3, j_4, j_5\}$, is available. Upon receiving the information, a schedule for the fleet of drones is generated, where each drone is assigned an ordered list of scheduled deliveries. The scheduling model in Ghelichi *et al.* (2022) can, for example, be used to find the schedules for each drone in the fleet. Table 2 shows the ordered list of deliveries (or schedule) for drones i_1 and i_2

Figure 1 Illustration of a probability map



Source: Figure created by Zabih Ghelichi

Figure 2 Illustration of the system status at timestamp t_1



Source: Figure created by Zabih Ghelichi

Table 2 System status at timestamp t_1

Drone	Current distance to the platform	Schedule
i_1	0	j_1, j_5, j_4
i_2	0	j_2, j_3

Source: Table created by Zabih Ghelichi

and their current distance from their platforms at the timestamp t_1.

Once the deliveries are assigned, drones depart their platforms heading to the first demand location in their task list. Although the scheduling model assumes a flight duration between any two points, the real flight duration might be impacted by different factors, such as wind and air density. Therefore, the real flight duration for each drone on each trip can be deemed as a random variable.

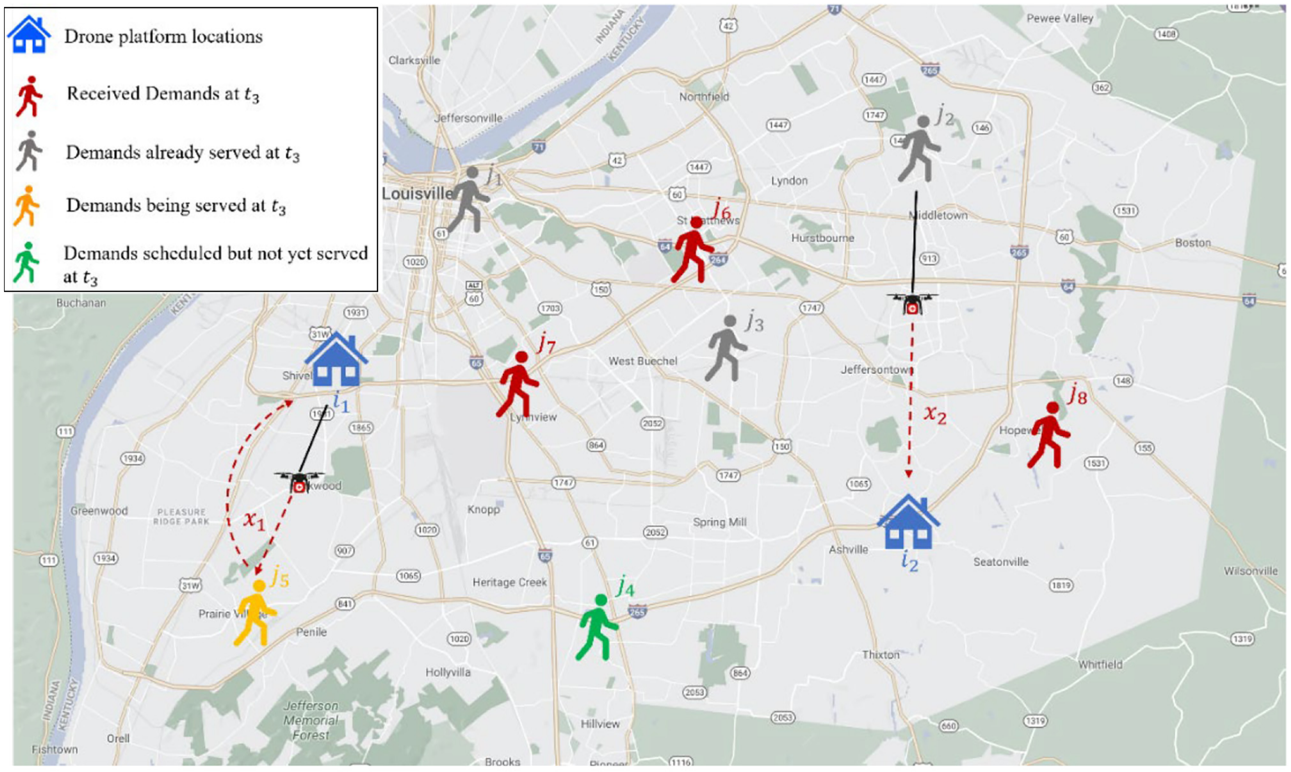
At this point, we simulate drone flights. The simulation model will provide realizations of drone setup times, flight times and load drop-off times. Furthermore, we assume the possibility of facing battery failure during any trip. Given drone's battery might be impacted by an internal or external factor, like temperature, we need to continuously monitor the energy required to finish the trip and the remaining amount of energy in the drone's battery.

Now, after some time interval δ, the rescue team receives a new set of information at timestamp $t_2 = t_1 + \delta$. Within the time interval $[t_1, t_2]$, the drones have been performing the deliveries based on their assigned schedules. Figure 3 shows the current position of drones at the timestamp t_2 and the status of the demand points. At timestamp t_2, there are four different types of demand locations:

1 demand locations from the timestamp t_1 that are already served (shown in gray);
2 demand locations from the timestamp t_1 that are being served (shown in orange);
3 demand locations from the timestamp t_1 that are already scheduled but not yet served (shown in green); and
4 new demand locations at timestamp t_2 (shown in red).

Figure 3 also shows that drone at platform i_1 is on the way toward demand location j_5, and drone at platform i_2 is returning to its platform after serving demand location j_2. Observe that, at the timestamp t_2, the demand locations j_1, j_2 and j_3 are *served*, j_5 is *being served*, j_4 is *scheduled but not served yet*, and the new demand locations j_6, j_7 and j_8 have requested aid packages, i.e. $F_t = \{j_1, j_2, j_3, j_5\}$, $K_{t_2} = \{j_4\}$ and $\mathcal{J}_{t_2} = \{j_6, j_7, j_8\}$. At this time, the rescue team needs to update the schedule of drones based on the current position of drones as well as the unserved demand points and new demand locations. By combining the unserved demand points from the previous timestamp and new demand locations in one pool, i.e. $\mathcal{J}_{t_2} = \mathcal{J}_{t_2} \cup K_{t_2}$, the scheduling model generates an updated schedule of deliveries for each drone. Not that, at this point, the scheduling model needs to take the current position of the drones into account to generate the schedule for each drone. At timestamp t_2, the current distance of drone i_1 and i_2 from their platforms are x_1 and x_2, respectively. We assume once a drone

Figure 3 Illustration of the system status at timestamp t_2


Drone platform locations
Received Demands at t_3
Demands already served at t_3
Demands being served at t_3
Demands scheduled but not yet served at t_3
j_1
j_2
j_3
j_4
j_5
j_6
j_7
j_8
i_1
i_2
x_1
x_2
Louisville

Source: Figure created by Zabih Ghelichi

departs its platform to serve a specific demand point, it must finish its trip, and the schedule for that trip cannot be changed. Table 3 shows the current state of drones and the updated schedule. The system repeats this process over the planning horizon.

5. Simulation-based performance evaluation model

In this section, we describe the details of our simulation system for drone-based delivery of aid packages. In this system, we integrate a scheduling model within a simulation model, which interactively exchanges information such that the output of the model is fed as an input to the other model and vice versa.

5.1 Simulation model

The proposed simulation model captures the variability in the following sources:

- *interval of updating the system* after receiving new information;
- *demand parameters*: the demand rate and the spatial distribution;
- *time parameters*: travel time, setup and loading time, payload drop-off time and fixing time; and
- *drone energy level*, and possibility of battery failure while delivering.

Figure 4 schematically illustrates the simulation platform procedure. At every update interval, the simulation model evaluates the status of the system. In this stage, the simulation model identifies the current configuration and requirements of the system including the information about the new demands, the status of the previously received demands, the status of drones and drones' current locations. It is worthy to note that the update interval δ is a random variable independent from drones' operation. Then, this information is fed into a scheduling model, which generates a set of decisions corresponding to the system status. The output of the scheduling model is an ordered list of deliveries fed back into the simulation model. In this way, the simulation and scheduling model continuously interact with each other. Together, they obtain a set of decisions from the scheduling model and the realizations of the uncertain parameters to enable the simulation model to simulate the flights and deliveries for the fleet of drones (yellow box in Figure 4).

To cope with the drone battery failure and energy level disruption, we discuss two alternative strategies for routing the drones when they face a potential interruption. This part of the simulation is devised to make real-time rerouting decisions in response to uncertainties that may impact drones' battery performance and, in turn, drone coverage ranges.

The first strategy is "Avoid & Return" (A&R), which enforces the system to avoid the risk of losing a drone – by enforcing the drone to return to its platform if it cannot complete a delivery. Figure 5 shows the schematic details of the A&R strategy. In this scheme, a monitoring system constantly observes the energy level and the status of each drone in the fleet. Once a drone departs from its platform toward an assigned demand location j, the monitoring system starts to observe the drone's status. If the drone faces an issue that impacts its energy level, for example, a sudden drop in the temperature or failure in the battery, then the system reestimates the energy required to complete the trip. If the energy is enough to complete the trip – we say a drone *completes its trip* when it successfully performs the delivery and returns to its corresponding platform – then the drone will continue its trip, and the monitoring system will keep observing the drone status until the delivery is completed. Once the delivery is completed, the demand location j is added to the set of served demands F_t and the drone returns to its platform. Otherwise, the drone will be immediately forced to return to its platform, and the delivery for demand location j will be scheduled in the next timestamp, i.e. j is added to set K_t.

Table 3 System status at timestamp t_2

Drone	Current distance to the platform	Schedule
i_1	x_1	j_6, j_7
i_2	x_2	j_4, j_8

Source: Table created by Zabih Ghelichi

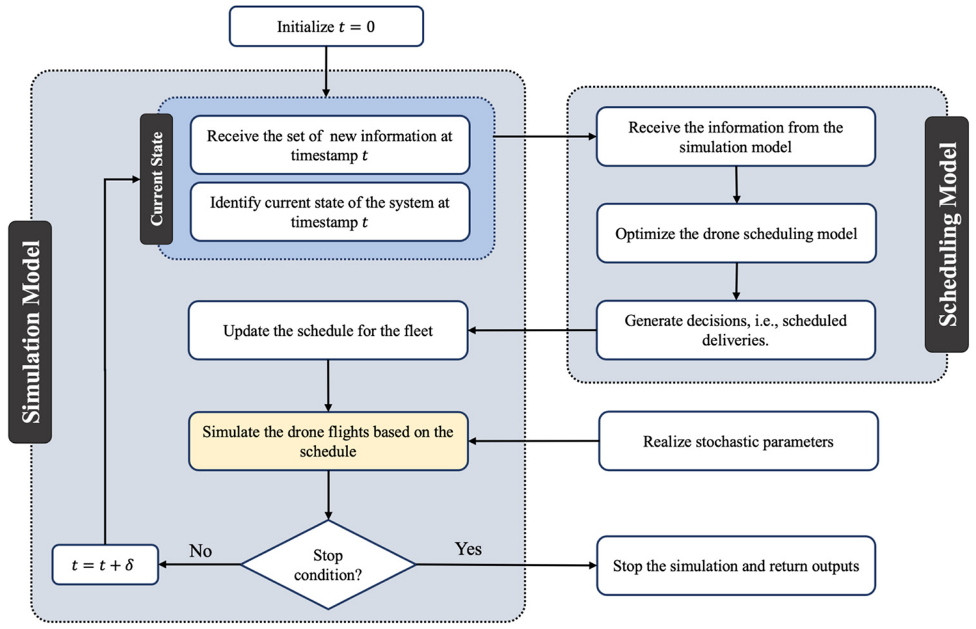
The simulation model realizes the interruptions at an assumed failure rate. A failure function can be designed to capture these interruptions. For instance, we can simulate the failure by introducing interruptions with a uniform distribution within 2 to 5 min interval and 10% chance of a major impact. Given the flexibility of the proposed framework, one may assume a different failure function that better fits their case. The drone energy consumption rate is assumed to be a function of drone mass, payload weight, estimated distance to fly and drone's specification (Figliozzi, 2017).

An alternative routing option is a "Push & Retrieve" (P&R) strategy, where the system tends to accept some level of risk when facing a potential failure in the drone battery. In contrast to the A&R scenario, where a drone immediately returns to its platform when the energy level is not enough to complete a trip, the P&R strategy considers the possibility of making the delivery even though the drone may not be able to return to its platform. That is, the P&R strategy accepts the risk of losing the drone at the expense of making the delivery.

Figure 6 schematically shows the P&R strategy and its components. In this scheme, after facing an interruption, the system estimates the required energy to complete the trip, i.e. making delivery and return. If the remaining energy is enough to complete the trip, the drone continues its trip and the monitoring system keeps observing the flight. Otherwise, the system evaluates the possibility of making the delivery by estimating the energy required to reach the demand point. If the energy is not enough to make the delivery, the drone immediately returns to its platform, we add the demand location j to the set K_t for scheduling it later again. If the energy is sufficient to make the delivery but not enough to make the return flight, the drone will be allowed to continue its trip, make the delivery, and stay at the demand location until it is retrieved. This drone will not be available for future deliveries until it is retrieved. In this strategy, we initialize a binary variable "Return," which equals to "True" if the drone returns to its platform after having served the demand location, and *False* if the drone stays at the demand location after having served the demand and cannot return.

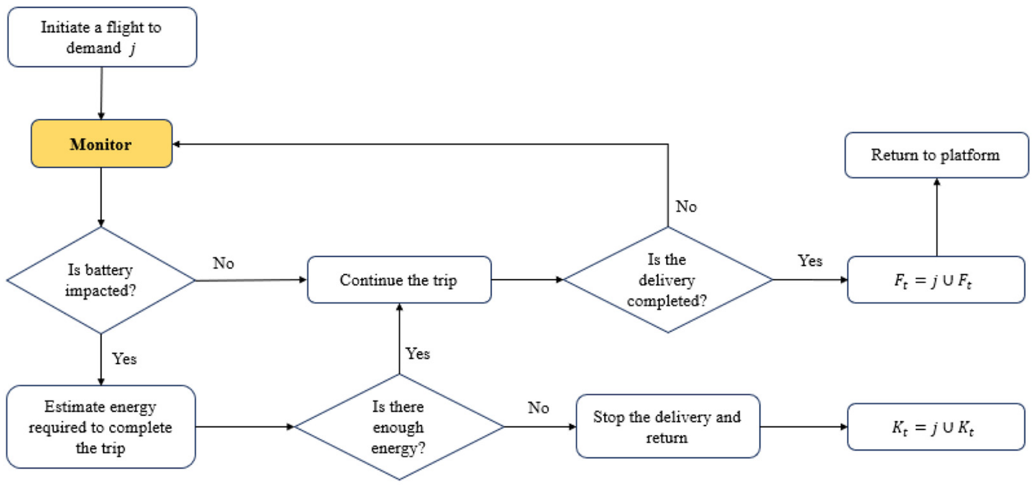
The system keeps monitoring the flight until the delivery is completed. Upon the completion of the delivery, the demand

Figure 4 Illustration of the proposed simulation model


Initialize $t = 0$
Simulation Model
Current State
Receive the set of new information at timestamp t
Identify current state of the system at timestamp t
Update the schedule for the fleet
Simulate the drone flights based on the schedule
Stop condition?
No
Yes
$t = t + \delta$
Scheduling Model
Receive the information from the simulation model
Optimize the drone scheduling model
Generate decisions, i.e., scheduled deliveries.
Realize stochastic parameters
Stop the simulation and return outputs

Source: Figure created by Zabih Ghelichi

Figure 5 Illustration of the Avoid & Return routing strategy



Source: Figure created by Zabih Ghelichi

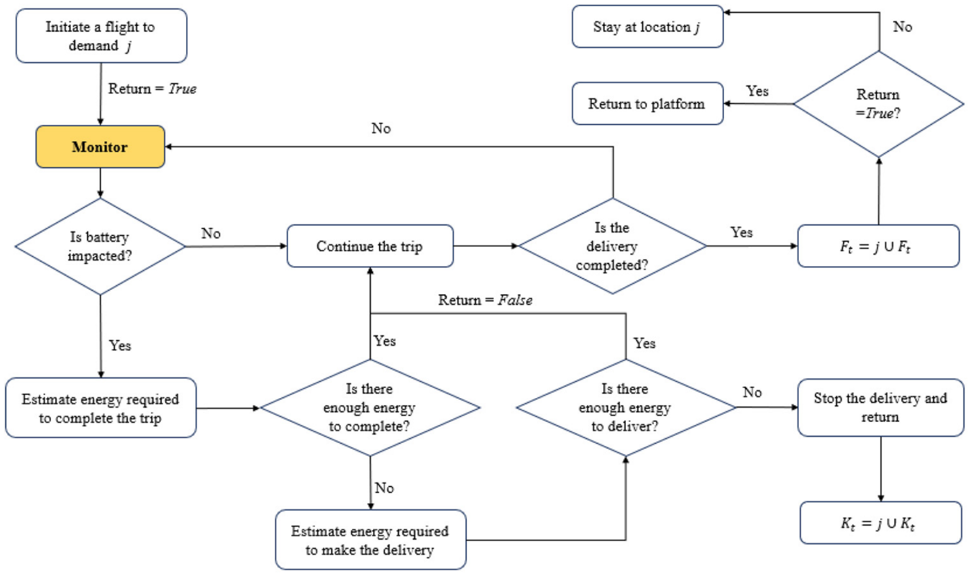
location j is added to the set of served demands F_t. At this point, if *Return* is equal to "True," the drone will return to its platform; otherwise, the drone needs to stay at the demand location j until it is retrieved. Technically, when a drone cannot come back to its platform, we have lost that drone in the system. In this situation, the "retrieval time" is the time required to either pick up the failed drone, bring it back to the platform and recharge/change its batteries or alternatively add a new drone to the system. Let r_{ij} denote the retrieval time of drone i from demand location j. We assume that the retrieval time is a function of roundtrip from platform i to demand location j and plus a preparation time.

5.2 Scheduling model

In this section, we describe an optimization model to solve the drone scheduling problem. Given a set of prelocated drone platforms and demand locations, the scheduling model generates an optimum ordered list of deliveries for each drone in the fleet. This model minimizes a measure of disutility/cost, which is nondecreasing in terms of time. Examples of such disutility/costs include delivery times, perishability, deadlines and waiting times. We adopt the timeslot drone scheduling formulation developed by Gentili *et al.* (2022) to optimally schedule drone trips.

Given a set I of m drone platforms and a set $\mathcal{J}$ of n demand locations, the problem is to optimally schedule and sequence

Figure 6 Illustration of the Push & Retrieve routing strategy


Initiate a flight to demand j
Return = *True*
Monitor
No
Is battery impacted?
No
Continue the trip
Is the delivery completed?
Yes
$F_t = j \cup F_t$
Return =*True*?
Yes
Return to platform
No
Stay at location j
Yes
Estimate energy required to complete the trip
Is there enough energy to complete?
Yes
Return = *False*
Yes
Is there enough energy to deliver?
No
Stop the delivery and return
$K_t = j \cup K_t$
No
Estimate energy required to make the delivery

Source: Figure created by Zabih Ghelichi

individual trips for each drone in the fleet and concurrently determine the assignment of deliveries to each drone over an h hours operational period. The objective is to minimize the total delivery and waiting time of the served demands plus a penalty for unserved demand locations. By adopting timeslots of length ϵ, we denote by $T = \{1, 2, \ldots, \frac{60h}{\epsilon}\}$ the index set of all timeslots over an h hour operational period. Let d_{ij} be the number of timeslots that a drone requires to make a roundtrip between platform $i \in I$ to demand location $j \in \mathcal{J}$. Assuming roundtrip is limited by the drone's flight range T_{max}, we let the binary parameter $a_{ij} \in \{0,1\}$ be equal to 1 if $d_{ij} \le T_{max}$; and 0 otherwise. We also assume that a drone requires one timeslot between every two consecutive trips for setup and charging operations. Given that requests for demands may arrive at any point in time and we schedule the deliveries at discrete intervals, we let w_j be the waiting time of a demand at location j, from the time that it sends a request for service until the delivery is scheduled. Binary decision variable $x_{ijt} \in \{0,1\}$ is equal to 1 if a drone from platform $i \in I$ returns to its corresponding platform at timeslot $t \in T$ after having served demand location $j \in \mathcal{J}$. In this setting, a demand at location $j \in \mathcal{J}$ may not receive a delivery due to either being out of coverage range, i.e. $\nexists\ i \in I : \ a_{ij} = 1$ or because it is impossible to schedule a delivery within h hours. Either way, we assign a penalty μ as the disutility for the unserved demand. An additional binary decision variable $y_j \in \{0,1\}$ determines if a demand location $j \in \mathcal{J}$ is served or not:

$$Min \sum_{i \in I} \sum_{j \in \mathcal{J}} \sum_{t \in T} x_{ijt} \left(t - \frac{d_{ij}}{2} + w_j \right) + \sum_{j \in \mathcal{J}} (1 - y_j)\mu \quad (1)$$

$$\sum_{i \in I} \sum_{t \in T} x_{ijt} \le 1 \ \forall\, j \in \mathcal{J} \quad (2)$$

$$\sum_{k \in \mathcal{J}: j \neq k} \sum_{\tau \in T: \tau = t - d_{ij}}^{t} x_{ik\tau} \le \hat{M}(1 - x_{ijt}) \quad \forall i \in I,\ j \in \mathcal{J},\ t \in T \quad (3)$$

$$\sum_{j \in \mathcal{J}} \sum_{t \in T: t \le d_{ij}} x_{ijt} = 0 \quad \forall i \in I \quad (4)$$

$$y_j \le \sum_{i \in I} \sum_{t \in T} x_{ijt} a_{ij} \quad \forall j \in \mathcal{J} \quad (5)$$

$$x_{ijt} \in \{0, 1\} \quad \forall i \in I,\ j \in \mathcal{J} \quad (6)$$

$$y_j \in \{0, 1\} \quad \forall j \in \mathcal{J} \quad (7)$$

The objective function minimizes the total delivery and waiting time plus a penalty associated with unserved demand locations. We refer to this measure as "disutility." Constraint (2) ensures each demand point is served at most once. Constraint (3) account for the scheduling and sequencing of drone trips. Specifically, Constraint (3) state that while drone i is serving demand location j, it cannot be any other trip until it completes its delivery at timeslot t. That is, if drone i is scheduled to serve demand location j at time t, it is booked from time $t - d_{ij}$ to t. In these constraints, $\hat{M} = \frac{d_{ij}}{\min_i\{d_{ik}\}}$ represents a maximum number of trips that a drone can make during the timespan of $[t - d_{ij.}, t]$. Constraint (4) guarantees that the completion of the first delivery of the day for each drone i cannot be less than its roundtrip. Constraint (5) determines if the demand location j is served. Finally, Constraints (6) and (7) specify integrality of the decision variables x_{ijt} and y_j.

Given that the simulation model responds in real time, it is essential to develop and implement time-efficient solution methods to solve the drone scheduling problem in real time. In this study, to solve the scheduling model in real time, we implement three approximate solution algorithms and compare their results. The first two solution algorithms are the greedy algorithm (GA) and the iterated greedy algorithm (IGA) (see Algorithms 1 and 2, respectively). Let's denote by $\Omega_i = \{1,2,\ldots j\}$, the set of scheduled demand locations assigned to drone i. GA starts with a sorted list of demand locations in descending order with respect to the sum of the total roundtrip time and waiting time denoted by P_j. Then, for each demand location j in the sorted list, GA selects the drone i^* such that by assigning the demand j to the end of the schedule of drone i^*, the total disutility is minimum:

```
Algorithm 1. Greedy Algorithm (GA)
1.  Initiate the set of unscheduled demand locations $J_t$.
2.  Sort demand locations in descending order of $P_j = \sum_{i\in I} d_{ij} + w_j$
3.  For each demand $j \in J_t$:
4.      Find the drone $i^* \in I$ to be assigned demand $j$ such that the total disutility is minimized.
5.      Assign demand $j$ to drone $i^*$
6.      Update the drone schedule: $\Omega_{i^*} \leftarrow \Omega_{i^*} \cup \{j\}$.
7.      Remove demand $j$ from the set of unserved demands.
8.  end
```

IGA extends the GA method by iterating over successive destruction and construction phases. In IGA, l_{max} denotes the maximum number of iterations and C_{ijk} denotes the completion time of serving demand location j if it is assigned to the *kth* position of the schedule of drone i. Given an initial feasible solution, the destruction phase randomly removes one demand, i.e. its task of delivery, from the scheduled set of deliveries for each drone and forms the set of unscheduled demands $\mathcal{J}'$. Then, the construction phase repairs and improves the solution by finding the position k^* on the schedule of drone i^* such that $C_{i^*jk^*}$ is minimum. IGA repeats this process over l_{max} iterations. For more information about IGA, we refer to Ghelichi *et al.* (2022):

```
Algorithm 2. Iterated Greedy Algorithm (IGA)
1.  Find a feasible schedule $\Omega_i$ for each drone $i \in I$ (for example using GA)
2.  while $l \le l_{max}$ do
3.      For each drone $i \in I$:
4.          Randomly remove a demand $j$ from the schedule of drone $i$:  $\Omega_i \leftarrow \Omega_i - \{j\}$
5.          Assign the removed demands to a set of unscheduled demands: $J' \leftarrow J' \cup \{j\}$
6.      end
7.      For each demand $j \in J'$:
8.          For each drone $i \in I$:
9.              Find the best position $k$ on the schedule of drone $i$ to assign demand $j$ and update $C_{ijk}$
10.         end
11.         Find $i^*, j,\ k^* \leftarrow argmin_{i,j,k}\{C_{ijk}\}$
12.         Insert delivery $j$ to $k^*$ position of the schedule of drone $i^*$
13.         Update the drone schedule: $\Omega_{i^*} \leftarrow \Omega_{i^*} \cup \{j\}$.
14.         Remove demand $j$ from the set of unserved demands.
15.     end
16.     $l \leftarrow l + 1$
17. end
```

The third algorithm is a myopic first in first out (FIFO) approach that assigns the first available demand in the set of unserved demands to the first available drone. FIFO starts with a sorted list of demands in descending order of waiting times. For demand locations with identical waiting time, the algorithm sort them again in descending order of the sum of the total roundtrip time, denoted by P'_j. Then, for each demand in the list of sorted demands, FIFO assigns the first demand in the list to the first available drone in the fleet:

```
Algorithm 3. First-In-First-Out (FIFO)
1.  Initiate the set of unscheduled demand locations $J_t$.
2.  Sort the demand locations in descending order of waiting time $w_j$.
3.  For demand locations $j$ with identical arrival time, sort them in descending order of $P'_j = \sum_{i\in I} d_{ij}$
4.  Assign the first demand location to the first available drone $i$ that can reach to the demand point.
5.  Remove the demand location from the set of unscheduled deliveries: $J_t \leftarrow J_t - \{j\}$.
6.  Update the schedule of the drone: $\Omega_i \leftarrow \Omega_i \cup \{j\}$.
7.  Return to step 4 until all the demands are assigned.
```

6. Experimental analyses

In this section, we discuss a series of analytical studies to shed light on the potential benefits that the proposed simulation performance evaluation tool can provide for decision-makers. All the numerical studies were conducted on a macOS with 64 GB memory and a 3.3 GHz Intel Core i5 processor. We have organized our experimental analyses into two subsections. In the first set of experiments (Subsection 6.1), we use our simulation-based evaluation tool for a case study for Central Florida. The goal of this set of experiments is to show how the proposed system can be used for decision-making and decision-support. The second set of experiments (Subsection 6.2) presents a series of numerical studies for a set of randomly generated instances. This section studies the trade-off among multiple parameters in our computational experiments.

The setting of these parameters are as follows unless it is mentioned otherwise:

- In all our experiments, we run a total of 50 replications of the simulation model for each problem instance and report on the average values.
- We assume the scheduling models in the simulation are solved by using the IGA algorithm with 20 iterations.
- All drones are identical and can fly at the maximum speed of 60km/h with a maximum operational range of 80 km, i.e., $D_{max} = 80$ km. The amount of energy that a drone consumes to make a roundtrip is calculated based on the energy function developed by Figliozzi (2017). It is assumed that the drone's battery has a nominal operational range of 80 km.
- We consider a failure rate of 10%. A failure rate of 10% means every time that the drone faces a battery issue, there is a 10% chance the battery has significant damage which can interrupt its flight. Interruptions are simulated in time intervals of every 5 to 10 min.

In this section, we evaluate the performance of the systems by reporting on different measures from the simulation model. These measures include the total disutility [objective function in Models (1)–(7)], waiting time, the percentage of served demands and computational times of the scheduling model (CPT), described as follows. The total disutility is the sum of the total time from when the demand arrives until it is served for served demand locations, plus a penalty of unserved demand points. Waiting time is measured as the time from when a demand arrives to the system until it is served. The percentage of the served demands is calculated as the total number of served demand at the end of the simulation divided by the total number of received demands. For each measure in each set of experiments, we report on the average CPU values over all the replications of simulations.

6.1 Case study for Central Florida

One of the most important features of such a system is the provision of an agile decision-making tool that can evaluate alternative solutions and strategies and support the process of decision-making. In this section, we perform analytical studies on the results obtained from platform location models to improve the decisions through a simulation-optimization procedure (Section 6.1.1), compare the performance of alternative platform optimization location models, i.e. deterministic and stochastic programming approaches (Section 6.1.2) and study the effect of using different drone routing strategies, i.e. A&R and P&R strategies (Section 6.1.3).

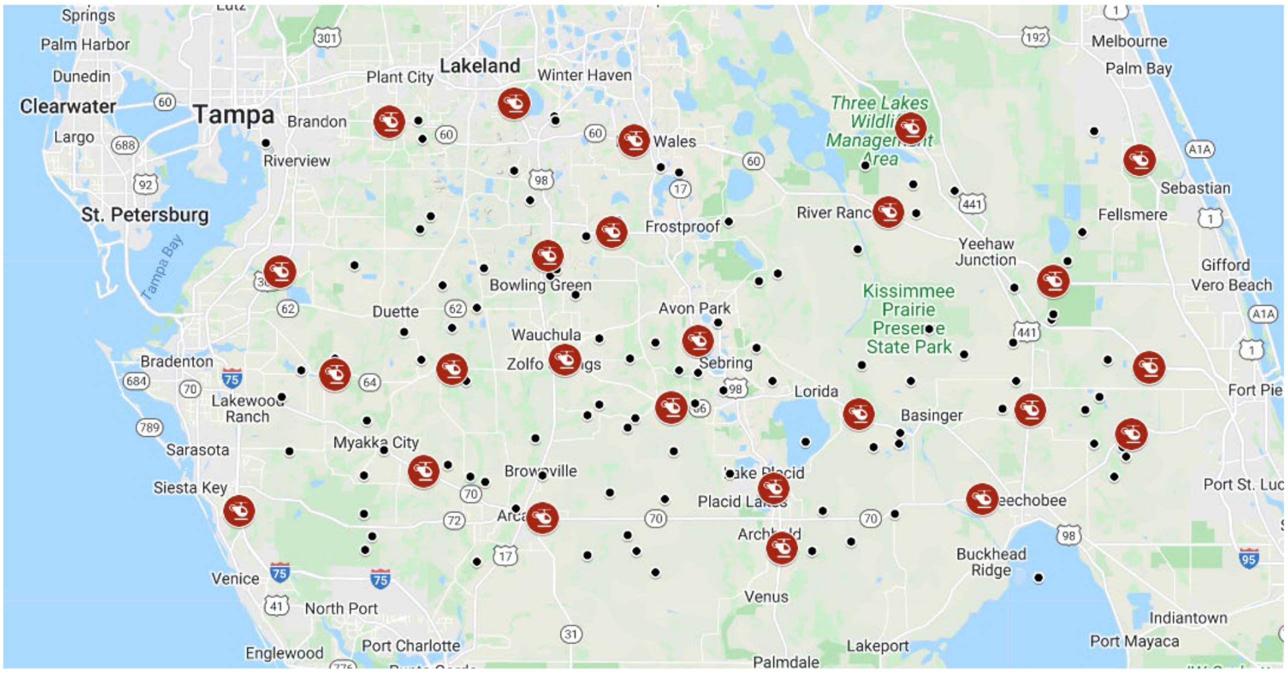
Following the experiments in Gentili *et al.* (2022), we consider a case study for Central Florida with 25 candidate drone platforms and 100 demand locations. Figure 7 shows the geographical locations of the candidate drone platforms (red icons) and demand locations (black icons).

In this set of experiments, we consider two alternative platform location models: a deterministic model, proposed by Gentili *et al.* (2022), and a stochastic model, proposed by Ghelichi *et al.* (2022). For convenience, we refer to these two models as the “deterministic” and “stochastic” models, respectively. The deterministic model assumes all the demand locations will have demand. On the other hand, the stochastic model assumes that not all the demand locations will necessarily have a demand; that is, it assumes that each demand location has a specified demand uncertainty. The outcome of these two models is a set of optimum locations for a given number of drone platforms, i.e. m, to serve demand locations in the disaster-affected areas. The stochastic model needs an uncertainty parameter $\alpha \in [0,1]$, which we set to a value 0.9 in our computational experiments. Based on Ghelichi *et al.* (2022), the stochastic model better shows its contribution over the deterministic model when the confidence level is relatively high, i.e. $\alpha = 0.9$. Therefore, we found this instance, $\alpha = 0.9$, a more compelling case in our analyses when evaluating and improving the performance of deterministic and stochastic models under our simulation model. For more information about the stochastic approach and its performance, refer to Ghelichi *et al.* (2022).

6.1.1 Analytical studies of the simulation-optimization procedure

In this section, we report on the results of the set of experiments we conducted designed to show the potential of the proposed simulation-based performance evaluation tool for the improvement of an initial decision. Our goal is to show how the simulation model can be easily integrated into an optimization

Figure 7 Geographical locations of candidate drone platforms (red icon) and potential demand locations (black icon)



Source: Figure created by Zabih Ghelichi

procedure to offer a tool that can be used for improving initial decisions. Specifically, we assumed that the demand points are uniformly generated between 5 to 10 locations in 20 to 40-min intervals within the disaster-affected area. We first select m drone platform locations by using the platform location model proposed by Gentili *et al.* (2022), a deterministic mathematical formulation to determine the optimum locations of a predefined number m of drone platforms in a disaster-affected area.

Based on this model, we find the optimum solutions for instances where $m = 8$, 13 and 18 for the case study of Central Florida. A solution to this problem is a list of m locations where the drone platforms must be located. We denote by λ_m^* the set of optimum platform locations when m drone platforms are selected using this deterministic model. Successively, we perform a set of analytical studies by evaluating the k-opt neighbors to λ_m^*. A k-opt neighbor to the optimum solution with m platforms is a list of m locations that differs from λ_m^* in k elements. In this study, we exhaustively explore the 1-opt neighbors of the optimum solution of the platform location models to find a better solution. The 1-opt neighborhood for the instances of $m = 8$, 13 and 18 include 136, 156 and 126 neighbors, respectively. By λ_m^i, we denote the i^{th} neighbor in the 1-opt neighborhood of λ_m^*. Once we have explored all the neighbors to λ_m^*, it is possible identify other solutions which can outperform the current solution, if any exists.

In this process, we can evaluate the performance of the system for each neighbor based on a set of measures, such as total disutility value, total waiting time and percentage of served demands, and check if there exists a neighbor (solution) that performs better in terms of all the defined measures compared with the current solution. We refer to the neighbors which outperform the optimum solution simply as "better neighbors". The following algorithm shows the general simulation-optimization procedure we applied for a general k-opt neighborhood. Note that, in the results reported in this section, we evaluated all the 1-opt neighbors and performed only one repetition of the algorithm (Table 4).

Table 5 shows some results of the simulation model for the optimal solutions λ_m^*, from the deterministic model, and those 1-opt neighbors that outperforms this optimum solution for different values of m. Table 5 clearly shows the significance of using the proposed simulation model in the decision-making process. This table shows that there are multiple solutions which are better than the initial chosen solution obtained from

Table 4

Simulation-optimization procedure

```
1. Solve the platform location problem by using an optimization model
2. Find the optimum set of platforms $\lambda_m^*$
3. Explore the k-opt neighborhood of $\lambda_m^*$
4. Run the proposed simulation model for $\lambda_m^*$ and its neighbors $\lambda_m^i$
5. Evaluate the performance of each solution based on a set of measures
6. Identify neighbors $\lambda_m^j$ that outperforms $\lambda_m^*$ in terms of all measure
   (if any exists)
7. $\lambda_m^* = \lambda_m^j$
8. Repeat Steps 2 to 6 until a stopping criterion is met
```

Source: Table created by Zabih Ghelichi

the deterministic model. For example, consider the scenario when eight drone platforms were located. The deterministic model suggested the solution λ_8^* [see Figure 8(a)]. However, the simulation-optimization method could find a solution such λ_8^{14} that outperforms λ_8^* in terms of all measures, i.e. total disutility, disutility per demand, waiting time per demand and percentage of served. In this way, we can observe that a simple 1-opt neighborhood search can improve the obtained results in all aspects. The reason for this observation lies in the fact that the simulation model accounts for a higher level of uncertainty compared to the deterministic platform location optimization model. For instance, the simulation model accounts for the waiting time of the demand from when it arrives to the system until it is served along with the possible failure in completing the deliveries.

Another interesting observation can be made by comparing the difference between the optimum solution, i.e. λ_m^*, and its better neighbors, i.e. λ_m^i, with respect to different number of available drone platforms (m). Table 5 shows that with a lower number of available platforms, i.e. $m = 8$, the difference between the optimum solution and its better neighbors is more significant. However, as the number of platforms increases, this difference gets smaller, especially in terms of total disutility. One major reason for this observation is that when there are few drone platforms available, one change in the location of a drone platform can have a nonnegligible impact on the entire system as it can significantly change the travel times and coverage of demand location. On the other hand, when there is a larger number of drone platforms available, the system is more robust against uncertainties in demand realizations. By robustness, we refer to a more solid and dependable system.

Figure 8(a) shows the locations of selected drone platforms in the optimum solution, i.e. λ_8^*, for $m = 8$, where the red icons represent the selected platforms and black dots are the nonselected locations. On the other hand, Figure 8(b) shows a better neighbor, i.e. λ_8^{14}, where the gray and blue icons are the platforms, which are removed and added in the neighbors, respectively. An interesting observation can be made by looking at the location of the new drone platforms in the better neighbors, i.e. blue icons. Figure 8(b) shows that the new platforms in the better neighbors are more prone to be placed on the edge of the set of candidate drone platforms. In other words, compared to the optimum solution in Figure 8(a), the drone platforms in better neighbors' solutions are more widely distributed, i.e. less clustered, within the disaster-affected area. The reason for this observation can be traced back to the impact of the drone locations on the waiting time. Indeed, a more distributed set of drone platforms tends to reduce the waiting time for each demand location compared to the case where all the drone platforms are clustered. Therefore, we can conclude that in the case of adding a new platform to the system, it would be more beneficial to add the drone platform closer to the edges of the set of candidate drone platforms rather than toward the middle of the location sites. In this way, we can provide a better balance in the waiting time for different areas of the disaster-affected area. Similar observations can be done also for the cases when $m = 13$ and 18.

Let's now study the drone location problem using the stochastic optimization approach proposed by Ghelichi *et al.* (2022), where the set of demand points is unknown. A solution

Table 5 Results of the simulation of the 1-opt neighborhood search for the deterministic model

M	Solution	Total disutility	Disutility per demand	Waiting time per demand	Percentage of served (%)
8	λ_8^*	18,001.88	207.48	96.64	76.91
	λ_8^{14}	16,045.79	196.02	93.27	78.59
	λ_8^{22}	17,045.82	201.41	91.91	77.19
	λ_8^{24}	17,598.81	202.27	93.62	77.36
	λ_8^{87}	17,137.78	200.23	92.97	77.65
13	λ_{13}^*	7,580.63	91.78	59.78	93.33
	λ_{13}^{51}	7,300.64	88.95	59.53	93.86
	λ_{13}^{111}	7,361.62	90.31	59.76	93.63
18	λ_{18}^*	6,177.50	72.59	51.26	95.55
	λ_{18}^{29}	5,752.01	67.25	50.70	96.60
	λ_{18}^{68}	5,509.63	68.20	50.44	96.3
	λ_{18}^{104}	5,900	71.94	50.89	95.61

Source: Table created by Zabih Ghelichi

to this model is also a set of m drone platforms. To differentiate the solutions of the stochastic model from the deterministic one, we denote by γ_m^* the set of optimum platform locations when m drone platforms are selected using the stochastic optimization model. Given the set of drone platforms obtained from this model, Table 6 shows the results of the simulation model for the optimum sets of drone platforms and their 1-opt better neighbors for different values of m.

Comparisons between Tables 5 and 6 show that the different platform location optimization methods resulted in different optimum solutions, i.e. λ_m^* and γ_m^*. We can observe that although the optimum solutions in the deterministic model, i.e. λ_{m}^*, are better than their counterparts in the stochastic model, i.e. γ_{m}^*, the 1-opt neighborhood search could find solutions in the neighborhood of the optimum solution obtained with the stochastic model can outperform the optimum solutions and their better neighbors in the deterministic case. For example, γ_8^{102} has a waiting time of 88.84 and outperforms any other solution obtained for $m = 8$ in Table 5. Indeed, we can observe that evaluating the solutions from an alternative approach can help us to improve the solutions. In the next section, we have a more in-depth discussion on the impact of using alternative platform location models for different cases.

In conclusion, our findings in this section show how the proposed simulation performance evaluation tool can be effectively used to improve an initial selection of platform locations obtained using platform location optimization models and how the simulation tool can be of a great value to improving the locational decisions. We observed that a simple 1-opt neighborhood search could significantly improve the solutions obtained initially by the deterministic location models. Therefore, we can expect that higher degrees of k-opt neighborhood search methods or more sophisticated algorithms can result in even better solutions. We also observed that simulating the drone deliveries for the solutions obtained from different platform location optimization models can result in different solutions. In this regard, we can use the proposed simulation tool to evaluate the solutions obtained from different models and choose the best system correspondingly. We will further discuss this point in the next section.

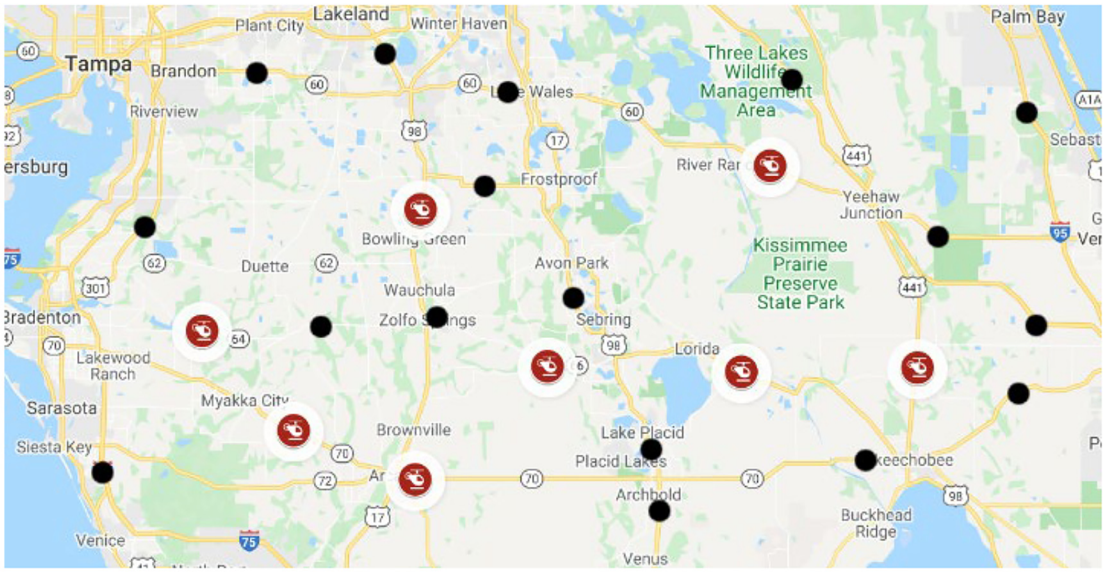
6.1.2 Computational studies to evaluate alternative solution strategies
This section aims to show the potential of the proposed simulation system for evaluating and validating the solutions obtained from different platform location optimization models. For this, we conducted a series of simulations to study the solutions obtained from the deterministic and stochastic models for different cases.

Table 7 compares the average total disutility and waiting time of the deterministic and stochastic models over all runs when the number of platforms to be located is equal to 8, 13 and 18, and where a total number of approximately 50 and 80 demand locations are received within an 8-h planning horizon. We assumed the intervals of updates are uniformly distributed within the range of [20,40] min. The reason for considering a different number of total demand locations is because not necessarily all the identified demand locations will have demand in a real-world scenario. Therefore, we wanted to understand how these different models perform when a low number of demands, i.e. 50, occur, and a high number of demands, i.e. 80, occur. Another factor we considered is the number of platforms which impact the performance of the system.

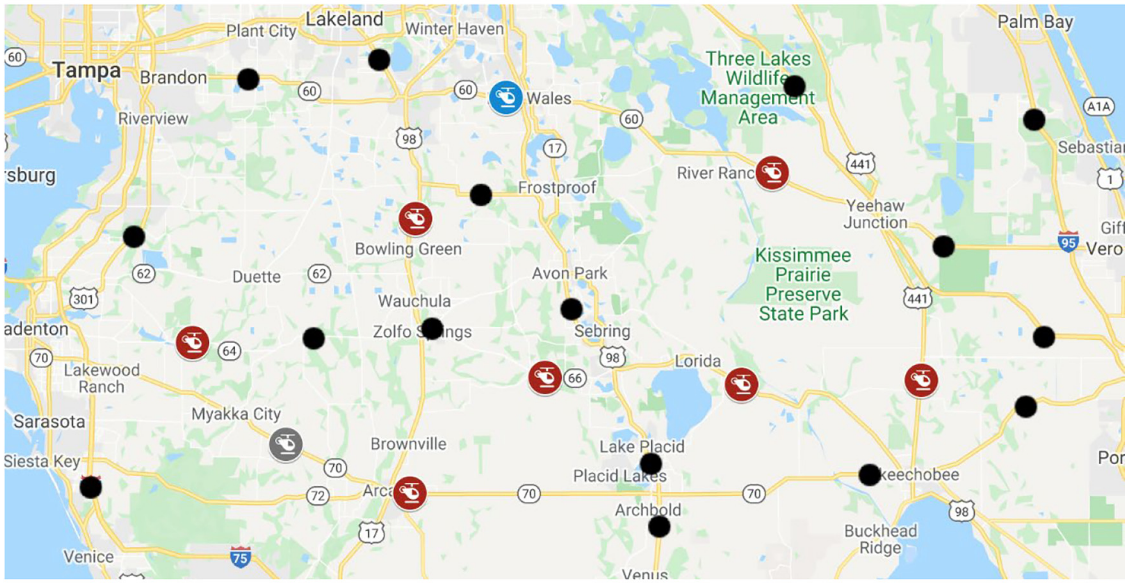
Table 7 reveals interesting findings on the performances of different models and their requirements. First, we can observe that the only case where the solution from the stochastic model outperforms the deterministic models is when there are a small number of platforms to locate, i.e. $m = 8$, and a small number of potential demand locations. This observation is in line with the finding of Ghelichi *et al.* (2022), where with more limited resources and demands, a stochastic model can result in better solutions. That is, a stochastic model for decision-making under uncertainty can capture the effects of the stochasticity on the set of demand locations and, in turn, lead to more robust solutions. This is especially important when the resources, e.g. the number of available platforms, are limited as the importance of the decisions on where to locate drone platforms become more relevant.

However, when the number of available platforms increases, we can see that the deterministic model results in higher quality solutions. This is mainly because a larger number of platforms can provide better coverage and accelerate the delivery

Figure 8 Platform locations in the optimum solution and its 1-opt neighbors for the deterministic model



(a)



(b)

Notes: The gray icon represents the removed platform and blue icons shows the newly added platform in the neighbors; (a) the location of drone platform in λ_8^*; (b) the location of drone platforms in λ_4^{14}
Source: Figure created by Zabih Ghelichi

operations; the deterministic model, where it is assumed that all the demand locations require deliveries, covers all the contingencies and, in turn, can produce better solutions.

Another interesting observation can be found through looking at the results with respect to the number of demands. We can observe that with higher demand 80, regardless of the number of available drones, the deterministic model outperforms the stochastic model. The reason for this observation lies in the fact the deterministic model assumes all the demand locations will require deliveries. Therefore, when we expect a larger number of locations from the set of all locations will require deliveries, the deterministic model tends to better capture the requirements of the system when it is compared to the stochastic model which ignores some scenarios.

6.1.3 Computational studies to evaluate "avoid & return" vs "push & retrieve" routing strategies

The proposed simulation system's capability of studying different routing strategies to prepare for battery-related

Table 6 Results of the simulation of the 1-opt neighborhood search for the stochastic model

m	Solution	Total disutility	Disutility per demand	Waiting time per demand	Percentage of served
8	γ_8^*	18,526.85	213.66	98.01	75.9
	γ_8^{17}	17,367.76	203.77	92.77	76.87
	γ_8^{23}	17,066.72	200.54	94	77.80
	γ_8^{98}	16,596.96	192.91	95.54	79.71
	γ_8^{102}	17,169.55	200.84	88.84	76.66
	γ_8^{104}	16,647.72	197.13	93.51	78.41
13	γ_{13}^*	8,171.06	95.92	61.74	92.87
	γ_{13}^{50}	7,803.24	94.42	60.35	92.90
	γ_{13}^{56}	7,863.05	92.29	59.45	93.16
	γ_{13}^{105}	7,581.25	90.64	60.36	93.69
18	γ_{18}^*	6,523.57	75.34	52.14	95.17
	γ_{18}^{25}	6,185.61	71.57	50.99	95.71
	γ_{18}^{29}	6,305.68	73.79	50.65	95.18
	γ_{18}^{32}	6,101.36	72.07	50.82	95.57
	γ_{18}^{59}	6,137.18	71.58	51.05	95.72
	γ_{18}^{114}	6,115.30	73.31	52.08	95.58
	γ_{18}^{119}	6,144.33	70.41	51.75	96.11

Source: Table created by Zabih Ghelichi

Table 7 Results of comparison between deterministic and stochastic models

m	Total demands	Model	Total disutility	Total waiting time
8	50	Deterministic	7,247.76	4,156.55
		Stochastic	6,644.31	3,946.71
	80	Deterministic	18,001.88	8,353.87
		Stochastic	18,302.71	8,558.71
13	50	Deterministic	3,685.16	2,945.96
		Stochastic	3,806.56	2,923.35
	80	Deterministic	7,580.64	4,911.83
		Stochastic	8,070.49	5,190.49
18	50	Deterministic	3,051.41	2,725
		Stochastic	3,257.50	2,825.50
	80	Deterministic	6,177.50	4,353
		Stochastic	6,437.24	4,526

Source: Table created by Zabih Ghelichi

interruptions is studied in this section. In particular, we compared the *A&R* versus *P&R* routing strategies when facing an interruption in drones' battery performance. In this set of experiments, we only used the deterministic model. Update intervals are assumed to be uniformly distributed between [20,40], where 5 to 10 demand locations are uniformly generated within each time interval. Table 8 reports on the total disutility, waiting time per demand location and percentage of the served demand for different combinations of the number of located platforms (m), the coverage range of drone (D_{max}), i.e. the battery capacity of the drone, the routing strategies and the drone retrieval times when the failure rate is 10%.

For P&R experiments, we set the retrieval time r_{ij} equal to the roundtrip, i.e. d_{ij}, to retrieve the drone plus a lead time, i.e. 60, 30 and 0 min. Nevertheless, one may consider the case if the failed drone is not retrievable anymore due to serious damages or lack of resources to retrieve the failed drone. In this case, an alternative case is to add a new drone to the system and desert the failed drone. However, this decision may depend on many factors including the availability of new drones, their price as well as the time required to get a new drone. To account for this scenario, we consider a scenario where acquiring a new drone takes 10 min. In case of completely losing a drone and not being able to add a new one, we can simply set r_{ij} (length of the planning horizon) equal to a very large value, which implies that the platform i is not operational anymore.

Table 8 compares the performance of the two routing systems with long-range ($D_{max} = 80$ km) and short-range $D_{max} = 30$ km) drones. As it was expected, when longer-range drones are used, the system significantly performed better in terms of total disutility and percentage of covered demand points compared to the case with short-range drones – for both the routing systems. However, it is surprising that the waiting time for each demand point for long-range drones is significantly higher than the short-range ones. The reason for this observation can be traced back to the fact that with longer-range drones, the system can cover more demands, and, in turn, more tasks will wait for service for each drone. Therefore, more demands can be served at the expense of a higher average waiting time for the demands. The percentage of the served demands in Table 8 also confirms this analysis. We can also observe that the percentage of served demands for cases with long-range drones is significantly higher than those for short-range drones.

In addition, we can observe that the difference between the solution of long-range and short-range systems, especially in terms of waiting time per demand and percentage of served demands, decreases as the number of platforms (m) increases. To provide a clearer picture of this observation, Figure 9 compares the waiting time per demand between long-range and short-range drones for different values of m and different routing strategies. The x-axis shows the number of available drone platforms (m) under each strategy and the y-axis shows

Table 8 Obtained results for A&R and P&R strategies with a failure rate of 10%

m	Coverage range (D_{max})	Strategy	Drone retrieval time (r_{ij}) (min)	Total disutility	Waiting time per demand (min)	% of served demand points
8	80 km	A&R	–	32,890	118	65
		P&R	$60 + d_{ij}$	35,788	124	61
			$30 + d_{ij}$	34,891	121	62
			d_{ij}	32,660	116	65
			10	28,428	108	71
	30 km	A&R	–	36,754	46	43
		P&R	$60 + d_{ij}$	36,869	47	43
			$30 + d_{ij}$	35,972	44	44
			d_{ij}	35,627	41	44
			10	35,627	41	44
13	80 km	A&R	–	13,340	68	90
		P&R	$60 + d_{ij}$	16,353	75	86
			$30 + d_{ij}$	15,571	73	87
			d_{ij}	13,340	68	90
			10	10,994	62	93
	30 km	A&R	–	28336	40	57
		P&R	$60 + d_{ij}$	28,773	39	56
			$30 + d_{ij}$	28,106	38	57
			d_{ij}	28,543	37	56
			10	27,439	37	58
18	80 km	A&R	–	8,970	54	95
		P&R	$60 + d_{ij}$	8,970	54	95
			$30 + d_{ij}$	8,855	53	95
			d_{ij}	8,740	52	95
			10	7,843	49	96
	30 km	A&R	–	20,930	38	70
		P&R	$60 + d_{ij}$	20,930	38	70
			$30 + d_{ij}$	20,263	37	71
			d_{ij}	20,148	36	71
			10	20,033	35	71

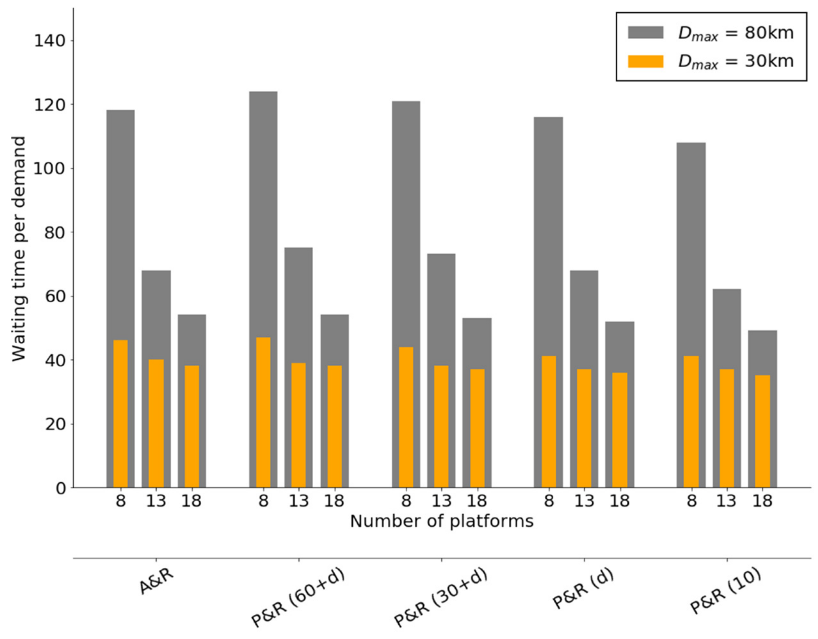
Source: Table created by Zabih Ghelichi

the corresponding waiting time per demand. This figure clearly reveals that regardless of the strategy, the difference of the waiting time per demand between long-range and short-range systems constantly decreases with respect to increasing the number of platforms. For example, for $m = 8$ and A&R strategy, the difference between the waiting time per demand of $D_{max} = 80$ km and $D_{max} = 30$ km is equal to 72, while the same comparison for $m = 13$ and $m = 18$ shows the difference between waiting times per demand are 28 and 16, respectively. The reason for this observation is rooted in the fact that a higher number of drone platforms can bring more demand locations in the coverage area of drones as more drone platforms can compensate for the limited coverage range. Roughly speaking, the higher degree of coverage provided by a large number of drone platforms decreases the advantage of long-range drones over short-range drones. Therefore, when there are a large number of drone platforms, each demand point has a higher chance to receive service from multiple drone platforms, which can reduce its waiting time, though drones may have a short coverage range.

More insights can be gained by comparing the results for the A&R and P&R routing strategies. Table 8 shows that compared to the A&R strategy, the relative performance of the P&R strategy correlated to the drones' coverage range (D_{max}), the number of available drones (m) and drone retrieval time (r_{ij}) after losing a drone. When longer-range drones are used, i.e. $D_{max} = 80$ km, we can observe that the A&R strategy generally performs better than the P&R scenarios. For any value of m in Table 8, the total disutility, waiting time per demand and the percentage of the served demands of the A&R strategy are either approximately equal or better than those from the instances of P&R strategy with different values of r_{ij}. The reason for this observation lies in the fact that long-range drones can fly further distances, and in case of losing a drone at far demand location, it takes a very long time to retrieve that drone. Furthermore, as the long-range drones can fly for a longer time, drones are more prone to multiple disruptions in each trip which also adds to the low performance of the P&R strategy. On the other hand, when shorter-range drones ($D_{max} = 30$ km) are used, the P&R strategy, especially with lower values of drone retrieval time (r_{ij}), outperforms the A&R strategy.

Of course, the lower value of drone retrieval time the better the P&R strategy performs. We assumed the lowest value of returning a drone to be equal to $r_{ij} = d_{ij}$, which means

Figure 9 Comparison of waiting time per demand for the A&R and P&R strategies with respect to different coverage ranges with a failure rate of 10%



Source: Figure created by Zabih Ghelichi

immediately after the drone *i* lands at the demand location *j*, we start to retrieve the drone from that demand location. We also studied the case when the failed drone is abandoned, and a new drone is activated after 10 min. As expected, this case significantly outperforms the A&R strategy for all the scenarios. However, this scenario depends highly on the availability of new drones and the time required to get one.

A more interesting observation can be found by comparing the results of A&R and P&R strategies with respect to a different number of available drones m. Table 8 shows that when the number of available platforms increases, the solutions obtained for both strategies converge. For example, when $m =$ 8 and $D_{max} = 80$ km, the largest difference between the waiting time per demand and the percentage of served demands of A&R and risk-take strategies are 10% and 6%, respectively. However, the same comparison for $m = 18$ and $D_{max} = 80$ km shows that the largest difference between waiting time per demand and the percentage of served demands of these two strategies are 3% and 1%, respectively.

The reason for this observation lies in the fact that a higher number of platforms bolsters the system's robustness against uncertainties induced by losing drones. That is, with more drones in the system, when one drone is lost or requires maintenance, there are many other drones that can take over the deliveries and perform the task. On the other hand, in the case of having few drones, losing a drone puts a larger burden on the entire system and increases the chance of further failures.

6.2 Numerical studies for tradeoffs among system components and parameters

This section reports on a series of numerical experiments to demonstrate the applicability of the simulation-based evaluation system and studies the trade-off among the parameters and algorithms in a drone-based delivery system in humanitarian logistics. These include embedded scheduling algorithms, updating interval after receiving new information, number of new demand arrivals in each time interval and distribution of flight and service times. A summary of the parameter setting is given in Tables 9 and 10, where notation Uniform (a,b) represents a uniform distribution with parameters a and b. Table 9 shows the parameter setting for different levels (low, medium, high and very high) of the number of demand points in each time interval and the parameter settings for different levels (high, medium, low and very low) of the update intervals. Table 10 summarizes the details and settings of different experiments carried out in this section. Specifically, in Table 10, for each experiment, the following is reported: scheduling algorithm choice, demand level, update interval level and demand to interval ratio, for example, low demand per interval to high frequency can be calculated as ratio of 12 [mean of uniform (9,15)] to 15 [mean of uniform (10,20)] equals to 0.8. The notation IGA (X) refers to the use of the IGA algorithm to solve the scheduling problem with X number of iterations, e.g. IGA (20) means the IGA algorithm with 20 iterations was used. In these experiments, we assume a set of 10 drone platforms, i.e. $m = 10$, are located on the Euclidean plane area of size 100 km × 100 km. For each experiment, a set of demand points is randomly generated on the foregoing plan.

6.2.1 Algorithm selection for drone scheduling problem

In this section, we study the effect of different scheduling algorithms, used to solve the drone scheduling optimization problem, on the quality of the obtained solutions. Our goal was to identify the best algorithm and strategy for solving the scheduling problem. One of the most important components of the proposed system is the scheduling component model. The scheduling of deliveries for drone is a problem that must be

Table 9 Table of parameter levels

Demands per interval	Level	Low	Medium	High	Very High
	Value	Uniform (9,15)	Uniform (15,25)	Uniform (25,45)	–
Update frequency (update interval)	Level	High Freq.	Medium freq.	Low freq	Very low freq.
	Value (min)	Uniform (10,20)	Uniform (20,40)	Uniform (40,60)	Uniform (70,100)

Source: Table created by Zabih Ghelichi

Table 10 Settings and details of experiments

Experiments	Scheduling algorithm	Demand per interval	Update frequency	Demand to interval ratio
Exp 1	FIFO	Medium	Medium freq.	0.7
Exp 2	GA	Medium	Medium freq.	0.7
Exp 3	IGA (5)	Medium	Medium freq.	0.7
Exp 4	IGA (20)	Medium	Medium freq.	0.7
Exp 5	IGA (50)	Medium	Medium freq.	0.7
Exp 6	IGA (100)	Medium	Medium freq.	0.7
Exp 7	IGA (20)	Low	High freq.	0.8
Exp 8	IGA (20)	Low	Medium freq.	0.4
Exp 9	IGA (20)	Low	Low freq.	0.24
Exp 10	IGA (20)	Low	Very low freq.	0.14
Exp 11	IGA (20)	Medium	High freq.	1.25
Exp 12	IGA (20)	Medium	Low freq.	0.4
Exp 13	IGA (20)	Medium	Very low freq.	0.24
EXP 14	IGA (20)	High	High freq.	2.34
EXP 15	IGA (20)	High	Medium freq.	1.17
EXP 16	IGA (20)	High	Low freq.	0.7
EXP 17	IGA (20)	High	Very low freq.	0.4

Source: Table created by Zabih Ghelichi

solved in real time and as fast as possible. However, there must be a balance between the agility in decision-making and the quality of the obtained decisions. Therefore, this section seeks to show the potential of the proposed simulation tool to identify the most appropriate scheduling algorithm which can provide good quality solutions in a reasonable amount of time.

We compare the performance of FIFO, GA and IGA algorithms. For the IGA algorithm, we consider 5, 20, 50 and 100 iterations. This section compares the results of EXP 1 to 6. Table 11 shows the average values of the disutility per demand, waiting time per demand, served demand points and computational time (CPT) of the scheduling algorithm for 50 runs of the simulation model for each algorithm. The results show that FIFO, a myopic algorithm, has the worst performance with the highest disutility and waiting time and the lowest percentage of served demands. On the other hand, IGA with 100 iterations outperforms the other algorithms as it results in the lowest disutility and waiting time and the highest percentage of served demands. Indeed, we can observe that as the scheduling algorithm gets more sophisticated [going from FIFO to IGA (100)], the quality of its solution also improves. However, there is a trade-off between the solution quality and the CPT of the algorithm.

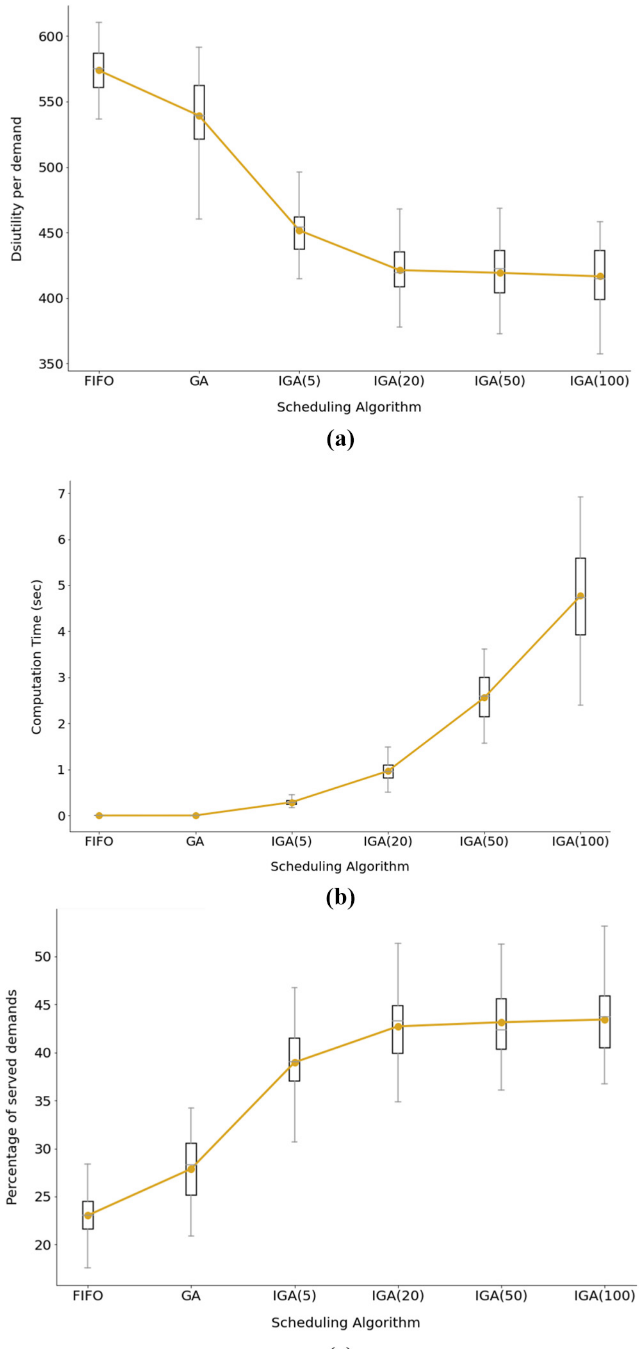
To provide a clearer picture of this trade-off, we visualized the boxplots and average value (golden line) of the disutility per demand, percentage of served demands and CPT of the algorithms in Figure 11(a) to 11(c), respectively. These figures unveil interesting observations from the performance of different algorithms. Figure 10(a) shows that the total disutility significantly decreases from FIFO to IGA (20), and, after this point, the contribution of the more sophisticated algorithm is not significant. On the other hand, Figure 11(b) illustrates that

Table 11 Results for different scheduling algorithms

Algorithm	Disutility per demand	Waiting time per demand	Served demands %	CPT of the algorithm (sec)
FIFO	573.82	204.41	23.04	0.00
GA	539.23	193.09	27.89	0.01
IGA (5)	451.61	158.70	38.98	0.29
IGA (20)	421.24	146.32	42.72	0.97
IGA (50)	419.16	146.30	43.15	2.56
IGA (100)	416.48	144.97	43.43	4.77

Source: Table created by Zabih Ghelichi

Figure 10 Comparison of the result for different scheduling algorithms

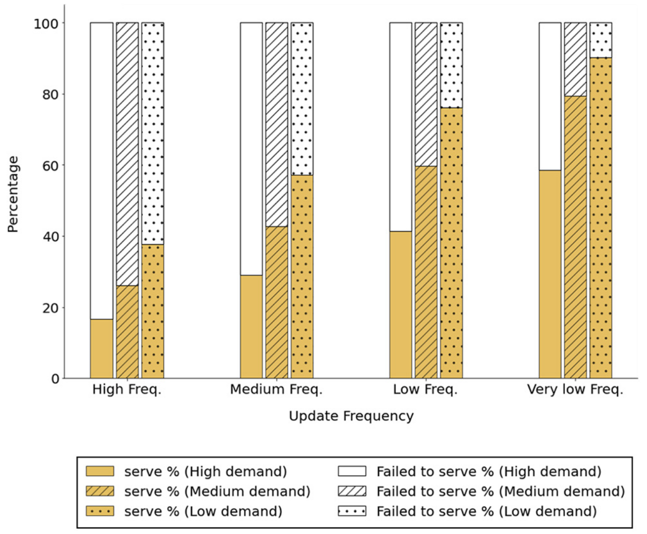


Notes: (a) Disutility per second; (b) computational times; (c) percentage of served demands
Source: Figure created by Zabih Ghelichi

the CPT of the algorithm does not significantly change up to IGA (20), while it grows after that. Not only does the computational time of algorithm increase as we move from FIFO to IGA (100) but also the boxplots show that the variance of the computational time also increases. Figure 10(c) shows that the number of served demand locations significantly grows from FIFO to IGA (20) and then hits a plateau, while CPT slightly changes from FIFO to IGA (20) and then rapidly increases.

In these figures, we can observe that the CPT of the algorithm from IGA (20) to IGA (100) goes from almost 1 s to about 5 s in average, i.e. five times higher. Although this extra computation time contributes to better quality solutions, it delays all the decisions to be made in real time, and in case of a higher number of demands, it can be an ineffective approach for a real-time decision-making tool. Another issue regarding using a time-consuming algorithm arises from the fact that the system is dynamic. That is, during the longer time that a time-consuming algorithm, such as IGA (100), takes to solve the scheduling problem, the system may have changed significantly, e.g. new information is received. Thus, the solution and schedule obtained based on the system condition considered by the algorithm at the beginning of the run is not of good quality as the system has changed and new decisions must be made. Therefore, we can conclude that IGA (20) represents a good balance between quality of solutions and computational time of the algorithm.

Figure 11 Comparison of percentage of served and failed to serve for different levels of demand and update interval



Source: Figure created by Zabih Ghelichi

6.2.2 Trade-off between update interval and demand

This section studies the impact of the length of update/time interval (δ), i.e. at the end of each time interval the system updates, and the number of demands received within each interval. Following our findings in the previous section, we use IGA (20) to solve the scheduling models.

Table 12 summarizes the results for EXP 4 and EXP 7–17. For the different combinations of update intervals of demand per interval levels, this table shows the disutility per demand location, percentage of served demands, percentage of demands failed to serve, average waiting time per demand location and CPT of the scheduling algorithm. This table reveals interesting insights into the performance of the drone

Table 12 Obtained results for combinations of different levels of demand and update interval

		Update frequency levels			
Demand per levels	**Measures**	**High freq.**	**Medium freq.**	**Low freq.**	**Very low freq.**
Low	Experiment	EXP 7	EXP 8	EXP 9	EXP 10
	Disutility per demand	446.21	330.38	216.36	137.75
	Percentage of served	37.59	57.12	76.10	90.19
	Percentage of failed to serve	62.41	42.88	23.90	9.81
	Waiting time per demand	146.64	124.58	101.65	90.67
	CPT of scheduling (sec)	1.58	0.22	0.04	0.01
Medium	Experiment	EXP 11	EXP 4	EXP 12	EXP 13
	Disutility per demand	515.93	421.24	325.88	213.57
	Percentage of served	26.02	42.72	59.73	79.37
	Percentage of failed to serve	73.98	57.28	40.27	20.63
	Waiting time per demand	160.85	146.32	132.59	114.54
	CPT of scheduling (sec)	6.17	0.97	0.22	0.04
High	Experiment	EXP 14	EXP 15	EXP 16	EXP 17
	Disutility per demand	574.57	502.83	436.26	342.65
	Percentage of served	16.57	28.91	41.37	58.56
	Percentage of failed to serve	83.43	71.09	58.63	41.44
	Waiting time per demand	174.13	161.58	154.85	143.75
	CPT of scheduling (sec)	21.56	4.69	1.38	0.30

Source: Table created by Zabih Ghelichi

delivery system and the trade-off between update interval and number of demands.

First, we can observe that for a given level of demands per interval, increasing the length of update intervals improves all the performance measures. This observation conforms to our expectation as for an identical number of demands received within each interval, a larger time interval gives enough time to the system to perform the deliveries and, in turn, the system will not face a large backlog. On the other hand, a shorter update interval, i.e. higher frequency, will result in more demands to be delayed and scheduled within the next update interval, and as a result, the disutility and system performance depreciate. That is, for example, to process 20 demand locations, a 40-min update interval will be enough to make all the deliveries and complete the tasks, while a 20-min update interval will require some demands to be delayed and scheduled within the next update interval. We refer to these delayed demands from one interval to another as "backlogs." Also, note that with lower intervals and an equal number of demands per interval, we will receive a higher total number of demands within an identical planning horizon, which itself adds to the effect of backlogs.

These backlogs are the main reason for increasing the disutility value. To shed light on this, we show the trade-off between the percentage of served/failed demands and different levels of update interval in Figure 11. In this figure, we can observe that increasing the interval significantly decreases the number of demand locations that could not be scheduled. Another evidence of this impact can be found by observing the CPT of the scheduling algorithm for different levels of update interval. Table 12 shows that the CPT of the scheduling model are tightly correlated to the percentage of the failed demand locations to serve. The backlogs accumulated within shorter intervals greatly impact the computational time of the scheduling model. That is, the accumulated backlog with lower intervals requires the scheduling model to process a larger and larger number of demand locations in each iteration of the simulation model. Additionally, we need to consider the delays added to the system due to frequent updates to all the decisions in the system.

More interesting insights can be found by comparing the results of the experiments with similar average demand to update interval ratios. Table 13 compares the waiting time per demand and CPT of scheduling problems for experiments with similar ratios. When comparing the experiments with similar demand to time interval ratio, we indeed study the frequency of update interval for a constant demand rate. For example, EXP 8 and EXP 12 are comparable as they both have a ratio of 0.4. That is, the experiment EXP 8 updates the system after every 30 min after accumulating a total of 12 demands on average, while EXP 12 waits for 50 min to accumulate a higher number of demands of 20 and then updates the system.

These comparisons are intended to provide deeper insights into the trade-off between the number of accumulated demands and the interval of updating the system. In other words, given a demand rate, how long should the system wait before updating its decisions? As we update the system later, more demands are accumulated and demands schedules are more delayed. However, one major upside for a longer update time is the possibility of accumulating more data and information and consequently obtaining decisions and schedules which are more effective and closer to the global optimum. Now, the question is: given this trade-off, which policy is more preferred, update in shorter intervals (higher frequency), with a smaller set of information on demands or wait to accumulate more information and update the system in larger intervals? The idea that exploring here contrasts with our previous discussion, where we compare the results for experiments with an identical number of demands per interval, e.g. EXP 7 to 10.

Table 13 Comparison between the experiments with similar demand to update interval ratio

Experiments	Ratio	Update frequency	Waiting time per demand	CPT of scheduling (sec)
EXP 9	0.24	Low freq.	101.65	0.04
EXP 13		Very low freq.	114.54	0.04
EXP 8	0.4	Medium freq.	124.58	0.22
EXP 12		Low freq.	132.59	0.22
EXP 12	0.4	Low freq.	132.59	0.22
EXP 17		Very low freq.	143.75	0.3
EXP 8	0.4	Medium freq.	124.58	0.22
EXP 17		Very low freq.	143.75	0.3
EXP 4	0.7	Medium freq.	146.32	0.97
EXP 16		Low freq.	154.85	1.38

Source: Table created by Zabih Ghelichi

In this regard, Table 13 shows a higher frequency of updating system, i.e. at shorter intervals, outperforms the case when larger intervals are used. All measures for EXP 9, EXP 8, EXP 12 and EXP 4 dominate the corresponding values associated with EXP 13, EXP 12, EXP 17 and EXP 16, respectively. The reason for this can be traced back to the waiting time of the demands from when they occur until they are scheduled and served. Compared to the cases with lower update frequency EXP 13, EXP 12, EXP 17 and EXP 16, the demands points are processed more frequently in EXP 9, EXP 8, EXP 12 and EXP 4; thus, the waiting time for each demand point is less in these scenarios. Note that the demands occur at different timestamps during each interval. For instance, Table 13 shows that the waiting time for each demand point in EXP 4 is 146, while EXP 16 requires each demand point to wait for 155 units of time. Another major drawback to allowing larger intervals is the accumulation of a high number of demands for scheduling, and, in turn, increasing the computational time of the scheduling model. Table 13 shows that for most of the cases the computational time of the system with lower frequency (larger intervals) is higher than the high-frequency case. With a growing number of demands, this impact can also grow and render the system less effective. In such a situation, it is better to either update more frequently or use other scheduling algorithms that require smaller run times.

This set of analyses are intended to show the potentials of the proposed simulation system for drone-based delivery of aid items in humanitarian logistics. We observed that the system has many parameters and components to study. This tool allows us to determine the systems settings and parameters, such as intervals of updating the system, selecting a suitable algorithm to solve the scheduling problem in real time and distribution of time-dependent parameters. Therefore, the proposed simulation-based system can be used as a powerful tool to identify the requirement of such a delivery system in a short amount of time, perform analytical studies and predict some future outcomes under different scenarios.

7. Summary and future directions

This paper presents a simulation-based performance evaluation model for the designing a system for timely delivery of humanitarian aid packages via a fleet of drones. The goal was to develop a simulation system that can allow one to evaluate performance of drone-based delivery systems. The applications of our system included but are not limited to performing analytical studies, supporting decision-making process, verifying the performance and applicability of different solutions, evaluating alternative strategies and predicting future contingencies and scenarios based on the current situation.

A simulation model is designed to simulate the drone flight and capture different sources of variations and uncertainties in a drone-based delivery system, including *interval of updating the system* after receiving new information, *demand parameters*: the demand rate and the geographical locations, *time parameters*: travel time, setup and loading time, payload drop-off time and fixing time, *drone energy level:* battery failure while flying. One major source of uncertainty in this system is the possibility of failures when a drone is operating, e.g. failure in the drones' battery or energy. We proposed two alternative routing strategies to deal with such failures. The first strategy requires a drone to immediately return to its platform if the system notices that the drone cannot successfully complete its trip by delivering the aid item and returning to its platform. The second strategy requires the drone to complete its delivery, even though it cannot make the return flight. In the latter case, the drone stays at the demand location until it is picked up or a new drone is added to the system.

Given a set of located drone platforms, the simulation model simulates the drone flights and delivery operations by realizing the stochastic parameters. To obtain a feasible and valid schedule for each drone, an optimization model is integrated into the simulation system that schedules drone flights for a given set of drone platforms and demand locations. The proposed optimization model is a timeslot formulation that essentially schedules and sequences individual trips for each drone in the fleet and concurrently determines the assignment of deliveries to each drone. Owing to the real-time context of this problem and the computational complexity of the drone scheduling model, we evaluated three heuristic algorithms, namely, FIFO, GA and IGA to solve the optimization model in real time.

To study different aspects of the proposed simulation model, we conducted a series of experimental analyses. The first set of experiments is designed for a case study of Central Florida. The purpose of this set of analyses is to show the capability of the proposed simulation-based performance evaluation model to support the decision-making process by improving the solutions obtained from platform location optimization models and evaluating performance of the system under alternative solutions, models and strategies. Our results and discussions highlight the importance of developing a simulation tool for supporting decision-making process, validation of system configuration and evaluating different strategies. The second set of experiments evaluates different system settings for randomly generated instances. In this section, we study the performance of multiple algorithms to solve the scheduling model. The results indicate that the iterated GA with quite a small number of iterations can find good quality solutions in a reasonable amount of time. In addition, we analyzed the trade-off between the update interval and the number of demands. Our experiments show that a higher frequency for updating the system (shorter update intervals) leads to a lower average waiting time for demand points.

While delivery drones have the potential to revolutionize the delivery of aid items in humanitarian logistics, they also come with various challenges and limitations. Examples of these limitations include payload capacity, limited range, battery technology, reliability, weather and environmental conditions, as well as authorization and regulations. To address these challenges, we can propose several future research directions. First, we should incorporate various parameters, strategies and policies for a real-world drone-based delivery system in humanitarian logistics into the simulation models. Examples of such settings include, but are not limited to, considering weather changes, no-fly zones, barrier avoidance and moving demand points. Furthermore, we assumed that the drones are dedicated to their platforms. However, the optimization and simulation models can be adjusted to route and schedule drones in a way that allows them to change their hosting platforms. Finally, as demonstrated in this paper, the proposed simulation tool can support the decision-making process by enhancing the solutions obtained from the optimization models through a simple 1-opt neighborhood search. In this regard, more sophisticated algorithms and approaches can be developed and integrated into the simulation model to further improve the designed systems through optimization models.

References

Chowdhury, S., Emelogu, A., Marufuzzaman, M., Nurre, S.G. and Bian, L. (2017), "Drones for disaster response and relief operations: a continuous approximation model", *International Journal of Production Economics*, Vol. 188, pp. 167-184.

Chung, S.H., Sah, B. and Lee, J. (2020), "Optimization for drone and drone-truck combined operations: a review of the state of the art and future directions", *Computers & Operations Research*, Vol. 123, p. 105004.

D'Uffizi, A., Simonetti, G., Stecca, G. and Confessore, P.C. (2015), "A simulation study of logistics for disaster relief operations", *Procedia CIRP*, Vol. 33, pp. 157-162.

Dayarian, I., Savelsbergh, M. and Clarke, J.-P. (2020), "Same-day delivery with drone resupply", *Transportation Science*, Vol. 54 No. 1, pp. 229-249.

Figliozzi, M.A. (2017), "Lifecycle modeling and assessment of unmanned aerial vehicles (drones) CO2e emissions", *Transportation Research Part D: Transport and Environment*, Vol. 57, pp. 251-261.

Fikar, C., Gronalt, P. and Hirsch, A. (2016), "A decision support system for coordinated disaster relief distribution", *Expert Systems with Applications*, Vol. 57, pp. 104-116.

Fu, C., Fu, C. and Michael, M. (2015), *Handbook of Simulation Optimization*, Springer, Cham.

Gentili, M., Mirchandani, P.B., Agnetis, A. and Ghelichi, Z. (2022), "Locating platforms and scheduling a fleet of drones for emergency delivery of perishable items", *Computers & Industrial Engineering*, Vol. 168, p. 108057.

Ghelichi, Z., Gentili, M. and Mirchandani, P.B. (2022), "Drone logistics for uncertain demand of disaster-impacted populations", *Transportation Research Part C: Emerging Technologies*, Vol. 141, p. 103735.

Ghelichi, Z., Gentili, M. and Mirchandani, P.B. (2021), "Logistics for a fleet of drones for medical item delivery: a case study for Louisville, KY", *Computers & Operations Research*, Vol. 135, p. 105443.

Kaplan, D. (2020), "How medical drones help save lives in Tanzania", available at: www.dhl.com/global-en/home/about-us/delivered-magazine/articles/2019/issue-3-2019/medical-drones-save-lives-tanzania.html

Kim, D., Lee, K. and Moon, I. (2019), "Stochastic facility location model for drones considering uncertain flight distance", *Annals of Operations Research*, Vol. 283 Nos 1/–2, pp. 1283-1302.

Kim, S.J. and Lim, G.J. (2020), "A real-time rerouting method for drone flights under uncertain flight time", *Journal of Intelligent & Robotic Systems*, Vol. 100 Nos 3/4, pp. 1355-1368.

Kim, S.J., Lim, G.J. and Cho, J. (2018), "Drone flight scheduling under uncertainty on battery duration and air temperature", *Computers & Industrial Engineering*, Vol. 117, pp. 291-302.

Kim, S.J., Lim, G.J., Cho, J. and Côté, M.J. (2017), "Drone-aided healthcare services for patients with chronic diseases in rural areas", *Journal of Intelligent & Robotic Systems*, Vol. 88 No. 1, pp. 163-180.

Krejci, C. (2015), "Hybrid simulation modeling for humanitarian relief chain coordination", *Journal of Humanitarian Logistics and Supply Chain Management*, Vol. 5 No. 3, pp. 325-347.

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

Mosterman, P.J., Sanabria, E., Bilgin, K., Zhang, J. and Zander, C. (2014), "Automating humanitarian missions with a heterogeneous fleet of vehicles", *Annual Reviews in Control*, Vol. 38 No. 2, pp. 259-270.

Murray, C.C. and Chu, A.G. (2015), "The flying sidekick traveling salesman problem: optimization of drone-assisted parcel delivery", *Transportation Research Part C: Emerging Technologies*, Vol. 54, pp. 86-109.

NHC (2024), "National Hurricane Center", available at: www.nhc.noaa.gov/

Oliver, M.A. and Webster, R. (1990), "Kriging: a method of interpolation for geographical information systems", *International Journal of Geographical Information Systems*, Vol. 4 No. 3, pp. 313-332.

Otto, A., Agatz, N., Campbell, J., Golden, B. and Pesch, E. (2018), "Optimization approaches for civil applications of unmanned aerial vehicles (UAVs) or aerial drones: a survey", *Networks*, Vol. 72 No. 4, pp. 411-458.

Paz-Orozco, H., de Brito Junior, I., Chong, M., Anacona-Mopan, Y., Segura Dorado, J.A. and Moyano, M. (2023), "Earthquake Decision-Making tool for humanitarian logistics network: an application in Popayan, Colombia", *Logistics*, Vol. 7 No. 4, p. 68.

Rejeb, A., Rejeb, S., Simske, H. and Treiblmaier, O. (2021), "Humanitarian drones: a review and research agenda", *Internet of Things*, Vol. 16, p. 100434.

Takahashi, D. (2021), "UPS uses matternet drones to deliver COVID-19 vaccines", available at: https://venturebeat.com/2021/08/24/ups-uses-matternet-drones-to-deliver-covid-19-vaccines/

Van Steenbergen, R. and Mes, M. (2020), "A simulation framework for UAV-aided humanitarian logistics", in *2020 Winter Simulation Conference (WSC)*, *IEEE*.

Vincent, J. (2021), "Self-flying drones are helping speed deliveries of COVID-19 vaccines in Ghana", available at: www.theverge.com/2021/3/9/22320965/drone-delivery-vaccine-ghana-zipline-cold-chain-storage

Yale, C.L.P., da Silva Frazão, M.L., de Mesquita, M.A. and Yoshizaki, H.T.Y. (2020), "Simulation applications in humanitarian logistics: a systematic literature review", in *2020 Winter Simulation Conference (WSC)*. *IEEE*.

Zhu, T., Boyles, S.D. and Unnikrishnan, A. (2022), "Two-stage robust facility location problem with drones", *Transportation Research Part C: Emerging Technologies*, Vol. 137, p. 103563.

Corresponding author

Zabih Ghelichi can be contacted at: zabihghelichi@gmail.com