Expert Systems With Applications 267 (2025) 126218

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

A matheuristic solution for efficient scheduling in dynamic truck–drone collaboration[☆]

Jinqiu Zhao[a], Yuying Long[b], Binglei Xie[a,*], Gangyan Xu[b], Yongwu Liu[a,b]

[a] School of Architecture, Harbin Institute of Technology (Shenzhen), Shenzhen, 518000, Guangdong, China
[b] Department of Aeronautical and Aviation Engineering, Faculty of Engineering, The Hong Kong Polytechnic University, 999077, Hong Kong, China

ARTICLE INFO

Keywords:
Vehicle routing problem with drone
Truck–drone collaboration
Synchronization constraints
Matheuristic
Column generation

ABSTRACT

Unmanned aerial vehicles (UAVs), or drones, have great potential for emergency response operations in areas with vulnerable road networks and infrastructure. However, the low battery capacity restricts their quick, cost-effective, and efficient aerial transportation capabilities. To overcome this drawback, hybrid systems that combine trucks and drones have emerged as a promising solution. Nevertheless, the fixed-binding mode between trucks and drones in most studies tends to impair efficiency by limiting drones' operational flexibility. This paper investigates a dynamic truck–drone collaboration (DTDC) strategy for efficient and flexible emergency response. This strategy enables drones to dynamically change take-off and landing locations on different trucks, which is beneficial in emergency scenarios with common road network disruptions. Despite its advantages, the DTDC strategy introduces additional complexity to the scheduling problem, resulting in a time-consuming solution. To enhance solution efficiency and improve the application prospects of this strategy, we propose a matheuristic to decouple DTDC's multiple synchronization constraints, separating the scheduling problem into three decision-making processes: demand allocation, truck routing, and drone scheduling. Additionally, two alternative matheuristic algorithms are designed to target accuracy and computing efficiency, respectively. The empirical results indicate that the proposed heuristics outperform the state-of-the-art solver and several metaheuristics. A sensitivity analysis confirms that improvements in drone endurance and the strategic reservation of drone parking slots on trucks can significantly improve the DTDC strategy's performance.

1. Introduction

The increasing occurrence of disasters and crises on a global scale has emphasized the necessity for efficient emergency response systems (Zhong, Shi, Fu, He, & Shi, 2010). Rapid urbanization and urban agglomeration development have compelled governments to prioritize emergency response, focusing on regulating post-disaster emergency resources and action plans. However, transportation networks and infrastructure often become vulnerable during disasters, leading to road blockages due to flooding or building collapses (Faturechi & Miller-Hooks, 2014), which restricts the use of ground vehicles for response activities (Yáñez-Sandivari, Cortés, & Rey, 2021; Yu et al., 2020). For instance, the 2021 Henan flood in China severely damaged numerous highways, hindering the delivery of relief supplies to affected communities. In 2022, the Guangdong flood in China destroyed urban roadways significantly, decreasing the effectiveness of emergency response activities and posing a risk to the safety of rescue crews during transportation. Hence, it is crucial to develop effective and flexible emergency response systems that can handle disruptions in transportation networks and infrastructure.

Advancements in drone technology offer promising solutions, providing high mobility, flexibility, and access to otherwise inaccessible areas (Al-Hilo, Samir, Assi, Sharafeddine, & Ebrahimi, 2020; Dayarian, Savelsbergh, & Clarke, 2020). Despite the potential, drones' limited service range due to inherent battery capacity constraints (Song, Park, & Park, 2022) remains a challenge. Hybrid transportation systems combining trucks and drones have emerged as a promising solution (Murray & Chu, 2015), attracting significant research interest. In such hybrid systems, the strategy for effective truck–drone collaboration is crucial. While drones offer remarkable flexibility, most existing strategies, such

[☆] This work was supported by Shenzhen Philosophy and Social Science Planning Project, China (SZ2023B007) and Natural Science Foundation of Shenzhen Municipality, China (JCYJ20210324133202006).

[*] Corresponding author.
E-mail addresses: 20b956010@stu.hit.edu.cn (J. Zhao), yuying.long@connect.polyu.hk (Y. Long), xbl@hit.edu.cn (B. Xie), gangyan.xu@polyu.edu.hk (G. Xu), yongwu.liu@connect.polyu.hk (Y. Liu).

https://doi.org/10.1016/j.eswa.2024.126218
Received 10 January 2024; Received in revised form 26 November 2024; Accepted 17 December 2024
Available online 24 December 2024

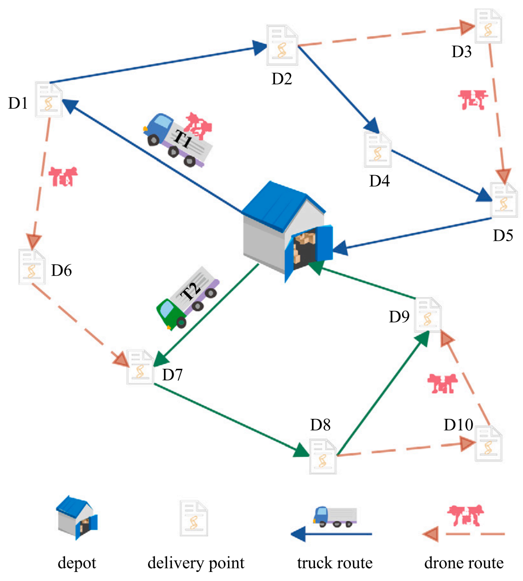


Fig. 1. An example of a DTDC solution.

as those in Gu, Poon, Luo, Liu, and Liu (2022), Moshref-Javadi, Lee, and Winkenbach (2020), Schermer, Moeini, and Wendt (2019), Wang, Poikonen, and Golden (2017), require the use of fixed and preset trucks for drone take-off and landing, limiting their mobility. This restriction results in longer waiting times and reduced system efficiency. As a result, dispatching trucks and drones efficiently becomes challenging in dynamic disaster scenarios, especially when roads or facilities sustain damage.

This paper investigates a dynamic truck–drone collaboration (DTDC) strategy that enhances the flexibility of both truck and drone routes. This strategy enables drones to dynamically change take-off and landing locations on any truck with available parking slots, which is beneficial in emergency scenarios with common road network disruptions. The DTDC strategy promotes rapid response by leveraging the mobility of both trucks and drones. Fig. 1 shows a typical DTDC scheduling solution that demonstrates the system's flexibility, allowing a drone to take off from Truck T1 at demand point D1, land on Truck T2 at demand point D7 after serving demand point D6, and then take off from Truck T2 at demand point D8. The concept of DTDC was first introduced in our prior work (Long, Xu, Zhao, Xie, & Fang, 2023), and this paper extends that research with a computationally efficient heuristic algorithm and a more comprehensive analysis.

While the DTDC strategy has demonstrated superior response capabilities, the complexity of its scheduling problem has increased substantially. The DTDC scheduling problem is a specialized Vehicle Routing Problem (VRP) variant with multiple synchronization constraints. According to Drexl's classification (Drexl, 2012), these constraints include task synchronization, which necessitates that each task be serviced exactly once; operational synchronization, which ensures spatiotemporal coordination between trucks and drones; and load synchronization, which involves the intricate logistical challenges associated with transferring drones between various vehicles. Incorporating each category of constraints introduces additional complexity and makes the problem computationally time-consuming, which is unacceptable in emergency scenarios where decision time is tight.

The main challenge of the DTDC scheduling problem is to simultaneously optimize truck routing and drone scheduling while keeping space, time, and loading synchronized. The objectives are threefold: optimizing demand allocation, establishing efficient truck routes, and scheduling drone flights. The decision variables include the drone take-off and landing schedule, the time and coordinates of these events, and the drone's parking slot assignment at the intersection point. Although a specially designed tabu search-based integrated scheduling algorithm in Long et al. (2023) can solve this problem to a certain extent, its accuracy and speed hinder its practical application in large-scale emergency response scenarios.

To enhance the solution's efficiency and improve the DTDC strategy's application prospects, we decouple the multiple synchronization constraints of DTDC, separating the scheduling problem into three decision-making processes: demand allocation, truck routing, and drone scheduling. Furthermore, we propose a robust matheuristic that integrates a column generation algorithm based on set-partitioning reformulation, enhanced with labeling and genetic algorithms. In this framework, we use the iterative process and tabu mechanism to check whether the current solution satisfies the synchronization constraint and circumvents the local optimum. Then, two alternative matheuristic algorithms are designed to target accuracy and computing efficiency, respectively. The effectiveness of these algorithms is validated through extensive numerical experiments and sensitivity analysis, providing strategic insights into DTDC management. The results demonstrate the practicality of the proposed methods and enrich our understanding of the DTDC scheduling problem.

The remainder of this paper is structured as follows: Section 2 provides a comprehensive review of relevant literature. Section 3 defines the problem and presents the associated mathematical model; Section 4 elaborates on the details of the matheuristic approach; Section 5 reports the results of a comprehensive computational study; and finally, Section 6 draws the conclusions and outlines potential directions for future research.

2. Literature review

This section will offer an extensive review and discussion of the relevant literature, with a particular emphasis on two aspects: truck–drone collaboration and VRP with multiple synchronization constraints.

2.1. Truck–drone collaboration

The concept of the truck–drone collaboration was initially introduced to address delivery optimization problems, stemming from a variant of the traveling salesman problem known as the Flying Sidekick Traveling Salesman Problem (FSTSP), introduced by Murray and Chu (2015). This problem considers a scenario where a drone collaborates with a conventional truck for parcel delivery. Another related problem, the TSP-D, allows for cyclic drone operations and was proposed by Agatz, Bouman, and Schmidt (2018). These foundational studies laid the groundwork for further research into more efficient algorithms and extended applications (Boccia, Masone, Sforza, & Sterle, 2021; Dell'Amico, Montemanni, & Novellani, 2021, 2022; Schermer et al., 2019).

Building upon the FSTSP and TSP-D frameworks, research expanded to consider scenarios involving a single truck collaborating with multiple drones. Murray and Raj (2020) explored the multiple flying sidekicks traveling salesman problem (mFSTSP), focusing on the scheduling of multiple drones and the nonlinear power consumption related to drone speed and parcel weight. The TSP-mD, a variant including more than one drone aimed at minimizing total operational costs, has been studied extensively (Jeong, Yu, Min, & Lee, 2020; Kitjacharoenchai et al., 2019; Luo, Poon, Zhang, Liu, & Lim, 2021; Moshref-Javadi et al., 2020; Poikonen & Golden, 2020).

In recent years, there has been significant progress in optimizing truck–drone collaboration through algorithmic innovations. Karaköse (2024) introduced a genetic algorithm-based approach for last-mile delivery using hybrid truck–drone systems, demonstrating improved solution quality and computational efficiency. Bian, Song, He, and Chi (2024) developed a micro-evolutionary algorithm to address the simultaneous optimization of truck routes and drone schedules in hybrid delivery systems. Boccia, Mancuso, Masone, and Sterle (2024) contributed to the field by presenting both exact and heuristic methods for the Truck–Drone Team Logistics Problem, offering valuable insights into the trade-offs between solution accuracy and computational time. Other notable advancements include variable neighborhood search (Jiang, Dai, Yang, & Ma, 2024; Sacramento, Pisinger, & Ropke, 2019), iterative optimization algorithms (Es Yurek & Ozmutlu, 2018), and exact methods such as branch-and-bound (Dell'Amico et al., 2021), column-and-row generation (Boccia et al., 2021), and dynamic programming (Bouman, Agatz, & Schmidt, 2018). These diverse approaches reflect the ongoing evolution of truck–drone collaboration strategies, highlighting the need for innovative algorithms that effectively balance solution quality and computational efficiency.

The collaboration between multiple trucks and drones, known as VRPD, was first introduced by Wang et al. (2017). This problem generalizes the FSTSP and TSP-D to scenarios involving a fleet of trucks, each carrying a drone. Sacramento et al. (2019) and Euchi and Sadok (2021) explored VRPD with a focus on minimizing operational costs through various algorithmic solutions. Further studies have addressed the complexities of scheduling and routing in systems with multiple trucks and drones, developing mathematical models and heuristic algorithms to optimize delivery operations (Di Puglia Pugliese & Guerriero, 2017; Kitjacharoenchai, Min, & Lee, 2020; Schermer et al., 2019; Tamke & Buscher, 2021).

Our prior work introduced the DTDC scheduling problem (Long et al., 2023), which allows drones to dynamically choose their take-off and landing trucks, addressing the limitations of fixed platforms. This strategy has shown significant efficacy, especially in emergency response scenarios. However, the DTDC scheduling problem presents complex resource constraints and synchronization challenges that necessitate the development of more efficient solution algorithms (Long et al., 2023).

This study builds upon the existing literature by focusing on the DTDC strategy, which permits drones to take off and land on different trucks, a flexibility not commonly seen in most previous studies. Unlike other similar works (Amine Masmoudi, Mancini, Baldacci, & Kuo, 2022; Bakir & Tiniç, 2020; Jiang et al., 2024; Long et al., 2023; Wang & Sheu, 2019), we consider the availability of drone parking slots on trucks, which is more realistic and significantly impacts routing decisions. Our approach allows for a more flexible and efficient operation of the truck–drone system, optimizing routing decisions and task allocation. A comprehensive comparison of our study with closely related works is presented in Table 1, highlighting the differences between our research in addressing the intricate interactions and synchronization challenges within the DTDC scheduling problem.

2.2. VRPs with synchronization constraints

The VRPs with synchronization constraints have garnered increased attention in logistics research (Drexl, 2012). These problems inherently possess intertwined vehicle routes, where alterations to one route inevitably influence others, occasionally rendering certain routes infeasible.

The concept of synchronization is often indispensable in settings requiring the coordinated efforts of multiple transportation resources to execute a specific task. Drexl investigated this aspect through the lens of the VRP with trailers and transshipment, which finds its application in collecting raw milk from farms (Drexl, 2012, 2013). In this model, lorries and trailers act as transport mediums; the lorries operate autonomously to transport milk, whereas trailers are lorry-dependent. The chief objective lies in cost-efficiently routing both vehicle types while satisfying various constraints, including customer demands (Meisel & Kopfer, 2012).

The development of algorithms for VRPs with synchronization constraints has made some progress. Li, Lim, and Rodrigues (2005) developed a simulated annealing technique specifically designed to address the intricate routing issues. In addition, Dohn et al. introduced a branch-and-price method (Dohn, Kolind, & Clausen, 2009), which was subsequently enhanced by Luo et al. using an advanced branch-and-price-and-cut technique (Luo, Qin, Zhu, & Lim, 2016). Research endeavors that complement the advancements in algorithmic development have enhanced the theoretical understanding of VRP. In their study, Fink et al. explored a complex area of abstract VRP that required precise coordination between personnel and vehicles (Fink et al., 2019). In this particular framework, task initiation depends on the synchronized arrival of all essential personnel, who rely on vehicle availability. Afifi, Dang, and Moukrim (2015) identified two primary goals in this field: to decrease the time it takes for jobs to begin and to minimize travel time and waiting times for both workers and vehicles. These objectives reflect a dual emphasis on efficiency and responsiveness.

Despite advancements, the existing literature primarily focuses on VRPs with single types of synchronization constraints, often simplifying them into sequential dependencies. However, the DTDC scheduling problem represents a more complex variant of VRP involving multifaceted synchronization constraints such as task, operational, and load synchronizations. Traditional approaches, such as column generation, prove insufficient for directly addressing these multi-constraint scenarios (Luo et al., 2016). Relaxation and branching methods have been used to get around this complexity (Dohn et al., 2009; Luo et al., 2016). However, there is still a need for more efficient solutions to the synchronization problems unique to the DTDC strategy.

3. Problem modeling

3.1. Problem description

This study investigates an urban emergency-response situation defined by a certain set of rescue demands, denoted as C. Specifically, a fleet of trucks equipped with relief supplies and several drones depart from a central depot. Relief items, consisting of uniformly small-sized essentials such as relief kits, medications, instant food products, and water supplies, are easily transported by drones (Sah, 2019). Hence, it is sensible to simplify the drone's endurance mileage to a predetermined safety threshold, even considering the substantial influence of the payload on a drone's energy consumption. This enables us to focus our analysis on the complex requirements for synchronization between trucks and drones rather than the variable of endurance.

Within the operational framework of DTDC, trucks follow predetermined routes in order to satisfy demands and act as mobile depots for corresponding drones. After servicing one demand point, the truck proceeds to the next without waiting for drones to take off or land. Drones can leave the truck at each demand point to serve an allocated demand. They may dynamically choose any truck to return within their endurance to recharge and reload. This strategy underscores a dynamic and efficient approach to scheduling in the context of truck–drone collaboration, enhancing the potential of urban emergency response efforts.

The problem is modeled by a graph $\mathcal{G} = (\mathcal{N}, \mathcal{A})$, where $\mathcal{N}$ encompasses the demand locations (indexed from 1 to $|C|$) and the depot (indexed as 0). The set of arcs $\mathcal{A}$ is formally defined as $\mathcal{A} := \{(i,j) | i,j \in \mathcal{N}, i \neq j\}$. Arc travel times for trucks and drones are specified as $d_{i,j}$ and $\hat{d}_{i,j}$, respectively. Each truck embarks from the depot carrying q_0 drones.

We focus on the DTDC scheduling problem, encompassing truck and drone route optimization. Unlike standard VRPs, DTDC requires intricate coordination between trucks and drones, considering the mandatory retrieval of drones post-service. Assumptions that are consistent with literature (Long et al., 2023) include: (i) Drones can charge and

Table 1
Related works on truck–drone collaboration.

Reference	#T	#D	OB	ATD	DRA	DFA	LPS	Methodology
Murray and Chu (2015)	1	1	Min. makespan					Heuristics
Wang et al. (2017)	k	nk	Min. completion time	✓				Worst-case analysis
Schermer et al. (2019)	k	nk	Min. makespan	✓				Matheuristic
Kitjacharoenchai et al. (2019)	1	n	Min. delivery time	✓				Insertion heuristics
Agatz et al. (2018)	1	1	Min. completion time			✓		Heuristics
Sacramento et al. (2019)	k	k	Min. logistics costs	✓				Variable neighborhood search
Wang and Sheu (2019)	k	k	Min. logistics costs		✓			Branch-and-price
Jeong et al. (2020)	1	n	Min. completion time					Epsilon-constraint method
Moshref-Javadi et al. (2020)	1	n	Min. waiting time	✓		✓		Tabu search, simulated annealing
Bakir and Tiniç (2020)	k	n	Min. makespan		✓	✓		Dynamic discretization discovery
Poikonen and Golden (2020)	1	n	Min. travel time	✓				Heuristics
Liu, Liu, Shi, Wu, and Pedrycz (2021)	k	n	Min. logistics costs			✓		Tabu search, simulated annealing
Luo et al. (2021)	1	n	Min. makespen			✓		Tabu search
Euchi and Sadok (2021)	k	k	Min. traveled time			✓		Hybrid genetic algorithm
Gu et al. (2022)	k	k	Min. logistics costs	✓		✓		Hybrid local search
Amine Masmoudi et al. (2022)	k	n	Max. profit	✓	✓	✓		Hybrid simulated annealing
Long et al. (2023)	k	nk	Min. makespan	✓	✓	✓	✓	Tabu search
Jiang et al. (2024)	k	n	Min. logistics costs	✓	✓	✓		Variable neighborhood search
Our study	k	nk	Min. service delay	✓	✓	✓	✓	Matheuristic

[1]**#T**: number of trucks used; **#D**: number of drones used; **OB**: Objective function; **ATD**: Trucks or drones have the potential to visit all demand points; **DRA**: Drones return to another truck; **DFA**: Drones may fly from or return to any demand points visited by trucks; **LPS**: Limited drone parking slots for trucks.

Table 2
Notations.

Notation	Description
Sets	
C	set of all demand points
$\mathcal{N}$	set of all nodes in graph $\mathcal{G}$, including depot
$\mathcal{A}$	set of arcs connecting each node in graph $\mathcal{G}$
Parameters	
$d_{i,j}$	the travel time of trucks from demand point i to j
$\hat{d}_{i,j}$	the travel time of drones from demand point i to j
T_i	the latest service time of the demand point i
t^i_{serve}	the service time of the demand point i
M	an arbitrarily large positive number
L	the endurance mileage of drones
K	the number of available trucks
N_0	the maximum allowable number of drones on a truck
α	the synthetically adjustable coefficient of the objective
c	the number of demand points
Decision Variables	
w_i	a continuous variable indicating the delay of the demand point i, $w_i \geq 0$
x^k_i	a binary variable indicating whether the demand point i is served by a truck , $x^k_i \in \{0,1\}$
x^u_i	a binary variable indicating whether the demand point i is served by a drone, $x^u_i \in \{0,1\}$
$y^k_{i,j}$	a binary variable indicating whether a truck moves from demand point i to j, $y^k_{i,j} \in \{0,1\}$
$y^u_{i,j}$	a binary variable indicating whether a drone moves from demand point i to j, $y^u_{i,j} \in \{0,1\}$
q_i	a nonnegative integer variable indicating the number of drones parked on the truck arriving in the demand point i, $q_i \in \mathbb{N}^+$
t_i	a continuous variable indicating the time at which the truck or drone arrives at the demand point i, $t_i \geq 0$
l_i	a continuous variable indicating the time at which the truck or drone departures from the demand point i, $l_i \geq 0$

restock while the truck is en route, eliminating additional time expenditures; (ii) Each demand point, barring the depot, can accommodate only one drone take-off or landing; and (iii) Truckload capacity suffices for all rescue demands, whereas drone capacity is constrained to a single point per sortie.

The objective function minimizes the service delay $w_i = (t_i - T_i)^+$ across all demand locations. Here, T_i represents the expected time for a truck or drone to arrive at point i, and t_i is the actual arrival time. Such an objective accentuates equitable emergency response, a vital attribute in public service settings.

3.2. Mixed-integer programming model

To facilitate the model establishment, we first define the notations as shown in Table 2. Unlike the 3-index model in Long et al. (2023), we employ a simpler 2-index formulation, which has minimal impact on the solver's efficiency. The DTDC's entire MILP model is built as follows:

$$\min \sum_{i\in C} w_i, \tag{1}$$

subject to

$$w_i \geq t_i - T_i, \forall i \in C, \tag{2}$$

$$\sum_{i\in C} y^k_{0,i} = K, \tag{3}$$

$$\sum_{i\in\mathcal{N}, i\neq j} y^k_{i,j} = \sum_{s\in\mathcal{N}, s\neq j} y^k_{j,s}, \forall j \in \mathcal{N}, \tag{4}$$

$$x^k_j = \sum_{i\in\mathcal{N}, i\neq j} y^k_{i,j}, \forall j \in \mathcal{N}, \tag{5}$$

$$\sum_{i\in C, i\neq j} y^u_{i,j} \leq 1, \forall j \in C, \tag{6}$$

$$\sum_{j\in C, j\neq i} y^u_{i,j} \leq 1, \forall i \in C, \tag{7}$$

$$\sum_{j\in C, i\neq j} (y^u_{i,j} + y^u_{j,i}) \leq 2 - x^k_i, \forall i \in C, \tag{8}$$

$$x^u_i + x^u_j + y^u_{i,j} \leq 2, \forall i,j \in C, j \neq i, \tag{9}$$

$$x^u_i + x^u_j - y^u_{i,j} \geq 0, \forall i,j \in C, j \neq i, \tag{10}$$

$$x^k_i + x^u_i = 1, \forall i \in C, \tag{11}$$

$$y^k_{i,j} + y^u_{i,j} \leq 1, \forall i,j \in C, j \neq i, \tag{12}$$

$$l_i + d_{i,j} - M\left(1 - y^k_{i,j}\right) \leq t_j, \quad \forall i \in \mathcal{N}, \forall j \in C, j \neq i, \tag{13}$$

$$t_i + \hat{d}_{i,j} - M\left(2 - y^u_{i,j} - x^u_j\right) \leq t_j, \quad \forall i,j \in C, j \neq i, \tag{14}$$

$$t_i + t^i_{serve} \leq l_i, \forall i \in C, \tag{15}$$

$$l_j + \hat{d}_{j,i} - M\left(2 - y^u_{j,i} - x^u_j\right) \leq l_i, \quad \forall i,j \in C, j \neq i, \tag{16}$$

$$q_i - M \cdot x^k_i \leq 0, \forall i \in C, \tag{17}$$

$$q_i \geq \sum_{s\in C, s\neq i} y^u_{i,s} - M x^u_i, \forall i \in C, \tag{18}$$

$$q_i + \sum_{s\in C, s\neq i} \left(y^u_{s,i} - y^u_{i,s}\right) \leq N_0, \forall i \in C, \tag{19}$$

$$q_i + \sum_{s\in C, s\neq i} \left(y^u_{s,i} - y^u_{i,s}\right) - M\left(1 - y^k_{i,j}\right) \leq q_j, \quad \forall i,j \in C, j \neq i, \tag{20}$$

$$q_i + \sum_{s\in C, s\neq i} \left(y^u_{s,i} - y^u_{i,s}\right) + M\left(1 - y^k_{i,j}\right) \geq q_j, \quad \forall i,j \in C, j \neq i, \tag{21}$$

$$\sum_{s\in C, s\neq i} \left(\hat{d}_{s,i} \cdot y^u_{s,i}\right) + \sum_{j\in C, j\neq i} \left(\hat{d}_{i,j} \cdot y^u_{i,j}\right) - M x^k_i \leq L, \quad \forall i \in C, \tag{22}$$

$$q_i \in \mathbb{N}^+, \forall i \in C, \tag{23}$$

$$w_i, t_i, l_i \geq 0, \forall i \in C, \tag{24}$$

$$x^k_i, x^u_i, y^k_{i,j}, y^u_{i,j} \in \{0,1\}, \forall i \in C, \tag{25}$$

$$y^k_{i,j}, y^u_{i,j} \in \{0,1\}, \forall (i,j) \in \mathcal{A}, \tag{26}$$

Constraints (3)–(5) govern the flow equilibrium for trucks, whereas constraints (6)–(8) regulate the seamless operation of drones. Constraint (3) mandates that the quantity of departing trucks from the depot is confined to the available fleet size. Constraint (4) stipulates an equality between the number of trucks arriving at and departing from each node. Constraints (9)–(10) dictate that a drone can service only a single demand point during each sortie. Constraints (11)–(12) confirm the fulfillment of each demand, either via truck or drone. Additionally, constraints (13)–(14) describe the relationship between travel and arrival times, while constraints (15) specify that service completion precedes departure from any demand point. Constraint (16) precludes drones from departing before arrival, and constraints (17)–(21) impose carrying capacity limitations for each truck's drone inventory. Specifically, constraint (17) asserts that a truck carries drones only if it services the corresponding demand point. Constraint (18) obliges a truck to deploy with an adequate number of drones, and constraint (19) caps the number of drones onboard each truck. Constraints (20)–(21) formulate the computational methods to determine per-truck drone numbers. Constraints (22) restrict each drone's operational range. Finally, constraints (23)–(26) specify the range of values for the decision variables.

A MILP solver could theoretically resolve the MILP model for the DTDC scheduling problem. However, the problem's intrinsic non-deterministic polynomial-time hardness (NP-hardness), demonstrated by our prior work (Long et al., 2023), impedes practical implementation. The complicated collaboration between trucks and drones further amplifies the existing complexity, necessitating the incorporation of numerous logical constraints, such as constraints ((13)–(14), (16)–(18), (20)–(22)). The large number of big-M coefficients introduces significant challenges for the solver, resulting in the problem being highly complex and difficult to solve. As a remedy, heuristic approaches serve as efficient alternatives, producing near-optimal solutions within acceptable computational limits.

4. Matheuristics

4.1. Problem decomposition

The analysis of the MILP model has highlighted the substantial computational challenges inherent in the DTDC scheduling problem. Matheuristics, a combination of mathematical programming and heuristic frameworks, offers a feasible approach to addressing this particular difficulty (Archetti & Speranza, 2014). The proposed model classifies the decision variables into demand allocation, vehicle routing, and drone scheduling. This process significantly reduces the computational complexity of the DTDC problem by breaking it down into the following hierarchical components:

- *Demand Allocation:* Optimal assignment of demands between trucks and drones.
- *Truck Routing:* Determination of the most efficient truck routes based on demand allocation.
- *Drone Scheduling:* Scheduling of drone activities based on demand allocation and truck routes.

The principal computational complexities of the DTDC scheduling problem stem from the dynamic decision-making processes involved in drone transfers, specifically the take-off and landing procedures. Those operations lead to interdependencies and synchronization constraints along the vehicle routes. In order to address these intricacies, we propose a matheuristic framework that divides the overall problem into two natural sub-problems:

- Allocation and Truck Routing sub-problem (ATRSP);
- Drone Scheduling sub-problem (DSSP).

In our matheuristic framework, the ATRSP and DSSP are solved sequentially and iteratively to achieve convergence towards an optimal solution for the DTDC problem, as outlined in Fig. 2. We reformulated the ATRSP as a set-partitioning model and solved it using a customized column generation algorithm, accelerated by a labeling algorithm and a genetic algorithm. Further details are presented in Section 4.2. Meanwhile, the DSSP utilizes mathematical programming techniques that consider both the endurance and the number of drones, as elaborated in Section 4.3. We have incorporated a tabu mechanism into the framework to improve the efficacy of solutions and reduce repetitive exploration.

4.2. Sub-problem 1: Allocation and truck routing problem

The ATRSP focuses on allocating rescue demands to both trucks and drones, as well as establishing a truck routing plan. Contrary to the overarching DTDC model, the ATRSP does not include constraints related to truck–drone interactions. It is possible to turn the ATRSP

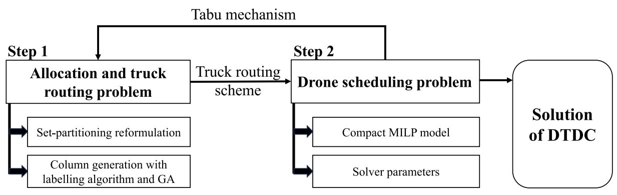


Fig. 2. Framework of the matheuristic.

into a MILP model by adding certain constraints from the DTDC model, but using solvers is difficult to do on medium- to large-scale problems because they are so complicated. These challenges are caused by the problem's NP-hard nature, but once we determine the allocation of rescue demands, we can reduce it to a classical VRP. To mitigate this issue, we propose reforming the ATRSP based on the set-partitioning model.

4.2.1. A set-partitioning reformulation for ATRSP

Following the notation presented in Section 3, we denote the rescue demands, and potential truck routes as C and Ω, respectively. The binary variable x_i^k ($i \in C$) represents whether or not rescue demand i is visited by a truck. In the same way, the variable z_r (where r belongs to the set Ω) is a binary indicator that represents the truck's use of route r. Finally, the variable p_r represents the service delay associated with route r. The ATRSP can be reformed as follows:

$$min \sum_{r\in\Omega} p_r z_r \tag{27}$$

s.t.

$$\sum_{r\in\Omega} z_r = K \tag{28}$$

$$\sum_{r\in\Omega} a_{ir} z_r - x_i^k = 0, \quad \forall i \in C \tag{29}$$

$$\sum_{i\in C} x_i^k \geq \frac{|C|}{2} \tag{30}$$

$$z_r, x_i^k \in \{0,1\}. \tag{31}$$

The objective function (27) aims to minimize service delays for a subset of rescue demands addressed by trucks. It implicitly assumes that drones handle other rescue demands without service delay, an assumption whose implications for the system's overall efficiency will be further examined in the DSSP. Constraint (28) stipulates the selection of K routes, while constraint (29) ensures each demand is exclusively associated with a single route, where $a_{ir} \in \{0,1\}$ denotes whether delivery point i belongs to route r. Constraint (30) mandates that at least half of the demands are fulfilled by trucks, consistent with assumptions from Section 3. Constraint (31) imposes binary restrictions on the decision variables.

4.2.2. Column generation algorithm

When undertaking an exhaustive enumeration of all potential routes for the set Ω, the set-partition model of ATRSP encounters computational inefficiencies due to the "curse of dimensionality" (Gilmore & Gomory, 1961). To achieve this, we utilize a Column Generation (CG) algorithm, which is inspired by the simplex method's inherent principles of iterative variable selection. The Master Problem (MP) of the CG is composed of set-partitioning formulations. The algorithm begins by solving a Restricted Linear-Relaxation Master Problem (RLMP) through the relaxation of integer constraints and consideration of a subset $\Omega' \subseteq \Omega$. The dual variables acquired through the resolution of the RLMP are then utilized to inform the formulation of a Pricing Problem (PP). The PP identifies potential columns that could be incorporated into the RLMP in an effort to improve the objective function. The algorithm continues iteratively until improvements to the objective function become unattainable, at which point it produces the optimal RLMP solution. The key aspects of the CG algorithm consist of techniques for accelerating the process, methods for deriving integer solutions, initial solution formulation, and PP construction and resolution.

Initial solution. The construction of an efficient initial solution is significant in initializing the RLMP (Lübbecke, 2010). Selecting an appropriate set of initial columns can accelerate the convergence of the CG algorithm. We use the Genetic Algorithm (GA) to create the initial columns quickly. The GA uses a streamlined genetic mechanism to do heuristic exploration within the complex solution domain, which increases the chances of finding solutions that are close to optimal. We build chromosomes using permutation coding and use the GA with enhanced elite retention (SEGA) (Affenzeller, 2001) to choose, recombine, and change people. The process of iterative evolution greatly enhances the overall quality of the initial feasible solutions.

Pricing problem construction. The PP within the CG framework serves to identify variables with negative reduced costs, thus facilitating the improvement of RLMP solutions. In our model, such variables represent feasible truck routes and their associated service times for each rescue demand. Consequently, the PP is formulated as an Elementary Shortest-Path Problem with Resource Constraints (ESPPRC) (Irnich & Desaulniers, 2005). The binary variable a_i equals 1 if rescue demand i is serviced, and y_{ij}^k equals 1 if arc (i,j) is part of the current route. The PP is mathematically defined as:

$$\text{(PP)}\ min \quad cost(r) = \sum_{i\in C} (w_i - a_i\lambda_i) - \sigma, \tag{32}$$

s.t.

$$w_i \geq t_i - T_i, \quad \forall i \in C, \tag{33}$$

$$\sum_{j\in C} y_{ij}^k - \sum_{j\in C} y_{ji}^k = \begin{cases} 1, & i = 0, \\ -1, & i = |C|+1, \\ 0, & others, \end{cases} \tag{34}$$
$$\forall i \in C \cup \{0, |C|+1\},$$

$$a_i - \sum_{j\in C} y_{ij}^k = 0, \quad \forall i \in C, \tag{35}$$

$$t_i + t_i^{serve} + d_{ij} - t_j + M \cdot (y_{ij}^k - 1) \leq 0, \quad \forall i,j \in C \cup \{0,|C|+1\}, \tag{36}$$

$$t_i, w_i \geq 0, \quad \forall i,j \in C \cup \{0,|C|+1\}, \tag{37}$$

$$a_i, y_{ij}^k \in \{0,1\}, \quad \forall i,j \in C \cup \{0,|C|+1\}. \tag{38}$$

The objective function (32) aims to minimize the route's reduced cost, parameterized by dual variables σ and λ, which are linked to constraints (28) and (29) respectively. The term w_i quantifies the delay associated with servicing each rescue demand. We introduce multiple constraints into the model to ensure the solution's feasibility and optimality. Specifically, the network-flow constraint (34) validates the route feasibility, while the linking constraint (35) enforces that each demand point is visited at most once. Time-related constraints (33), (36), and (37) compute the delay w_i using M as an upper limit. Binary restrictions on decision variables a_i and y_{ij}^k are enforced by constraint (38).

For demand points not included in the route, their service delay w_i is set to zero. This setting is justified as constraint (33) is inherently met when $a_i = 0$, and the objective function targets to minimize $\sum_i w_i$. Hence, the objective function can be reformulated as $\min \sum_{i\in C}[(w_i - \lambda_i)\cdot a_i] - \sigma$, aiming to identify a feasible route with the minimal $\sum_{i\in C_r}(w_i - \lambda_i)$, where C_r represents the set of included demand points.

Pricing problem solution. Algorithm 1 illustrates the common use of a labeling algorithm to address the PP. The algorithm assigns infinite labels to the remaining demand points and initializes the starting point's label to zero. Iteratively, the algorithm refines labels for non-dominated demand points until convergence, at which point it establishes a sub-route with minimally reduced cost.

Algorithm 1 Labeling algorithm for pricing problem

```
1: Initialize: $U \leftarrow \{[0]\}, P \leftarrow \emptyset$
2: while $U$ is not empty do
3:     Choose a partial path $Q \in U$ and delete $Q$ from $U$
4:     for Arc $(v_Q, w) \in \mathcal{A}$ starts from node $v_Q$ do
5:         if $[Q, w]$ is extensible then
6:             Extend $Q$ to $[Q, w]$ and add $[Q, w]$ to $U$
7:         end if
8:         if $w$ is the depot then
9:             $P \leftarrow P \cup \{[Q, w]\}$
10:        end if
11:    end for
12:    Delete the dominated partial paths from $U$ based on the
       dominance rule (39)–(42).
13: end while
14: Identify the optimal solution from $P$
```

In this context, the dominance rule substantially enhances the algorithm's performance. Specifically, an effective dominance rule reduces the cardinality of sets U and P during iterations, expediting the solution process. The dominance rule in the ESPPRC problem framework tells the difference between useless paths by comparing their goal values to possible extension sets $\xi(Q_1)$ and $\xi(Q_2)$ for paths Q_1 and Q_2 that share the same ending node. This rule is formally described in Proposition 1.

Proposition 1. *If partial paths $Q_1, Q_2 \in U$ satisfy Eqs. (39)–(42), Q_1 dominates Q_2, and Q_2 can be safely deleted from U without changing the optimal solution.*

$$v_{Q_1} = v_{Q_2}, \tag{39}$$

$$cost(Q_1) \le cost(Q_2), \tag{40}$$

$$\xi(Q_1) \supseteq \xi(Q_2), \tag{41}$$

$$L(Q_1) \le L(Q_2), \tag{42}$$

where v_Q denotes the last node of the partial path Q, $cost(Q)$ signifies the current reduced cost of Q, $\xi(Q)$ is the extensible set of Q, and $L(Q_1)$ is the departure time of the last node v_Q in the partial path Q_1.

Proof. To prove Proposition 1, we analyze each condition and demonstrate why Q_2 can be safely removed if Q_1 satisfies the stated dominance conditions.

1. **Node Equality** (39): Since $v_{Q_1} = v_{Q_2}$, both paths terminate at the same node. This ensures that replacing Q_2 with Q_1 will not disrupt continuity in the optimization process or change the feasibility of extensions originating from this node.
2. **Cost Condition** (40): If $cost(Q_1) \le cost(Q_2)$, choosing Q_1 over Q_2 minimizes the reduced cost of the solution. Therefore, retaining Q_1 while removing Q_2 does not increase the objective function value.
3. **Extension Feasibility** (41): Since $\xi(Q_1) \supseteq \xi(Q_2)$, all feasible extensions from Q_2 are also feasible from Q_1. Thus, replacing Q_2 with Q_1 ensures that no feasible extensions are excluded, preserving the completeness of the search space.
4. **Departure Time Condition** (42): If $L(Q_1) \le L(Q_2)$, the departure time from the last node of Q_1 is no later than that of Q_2. This ensures that any timing constraints satisfied by Q_2 are also satisfied by Q_1. Consequently, Q_1 does not introduce delays or infeasibilities when compared to Q_2.

Given that Q_1 satisfies all conditions of dominance, Q_2 can be removed from the set U without affecting the feasibility or optimality of the solution. This dominance relationship streamlines the optimization process by reducing the number of candidate paths to consider while preserving the integrity of the solution space.

This completes the proof.

Acceleration with heuristics. The labeling algorithm provides a thorough but time-consuming approach for identifying all feasible routes (Lübbecke, 2010). However, some iterations of the column-generation process may compromise its effectiveness. In such situations, the only requirement is feasibility, which is a negative reduced cost for the chosen column. We have found that heuristic techniques like the greedy algorithm, tabu search, and genetic algorithm effectively identify suitable columns under these settings. If a heuristic fails, we employ the labeling algorithm as an alternative approach. In this study, we implement the SEGA technique, previously used for generating initial solutions, to minimize the number of calls to the labeling algorithm. This approach aims to accelerate the computational process.

Acceleration with dual smoothing. The convergence of the CG algorithm in later iterations can be hindered due to the degenerate nature of the RLMP and fluctuations in dual variables, resulting in delayed convergence. In order to tackle this issue, we implement a smoothing technique as suggested by Pessoa, Sadykov, Uchoa, and Vanderbeck (2018). This study uses a smoothed dual vector, denoted as $\overline{\pi}$, to define the pricing sub-problem. The smoothed dual vector is obtained by combining the current RLMP's dual solution, denoted as π, and the current stable center, denoted as $\breve{\pi}$, using a smoothing factor $\alpha \in (0, 1)$. Upon the application of this technique, the columns that are generated can be categorized into three groups: (a) columns with negative reduced costs determined by both π and $\overline{\pi}$, which are directly included in the current RLMP; (b) columns with negative reduced costs determined by $\overline{\pi}$ but non-negative costs determined by π, requiring a gradual decrease in the value of α; (c) columns with non-negative reduced costs determined by $\overline{\pi}$, which triggers a sequential reduction in α and an update to $\overline{\pi}$.

Strategies for the integer solution. The CG involves iteratively addressing the RLMP and the PP, continuously incorporating columns with negative reduced costs into the RLMP until no such columns are available. The RLMP establishes an effective lower bound (LB) that applies to the ATRSP and the DTDC. When the RLMP solution is integral, it naturally satisfies the feasibility condition for the ATRSP, and the matheuristic can proceed to the subsequent steps. We impose a constraint on the decision variables to accommodate fractional outcomes, requiring them to be integers. This transformation converts the RLMP into an integer programming model. While the direct conversion approach may result in a less-than-optimal integral solution, the potential drawbacks can be mitigated by incorporating an optional column enumeration technique (Baldacci, Mingozzi, & Roberti, 2011). This procedure helps identify columns that could potentially enhance the quality of the integral solution.

4.3. Sub-problem 2: Drone scheduling

After establishing a feasible set of truck routes, as described in Section 4.2, consideration turns to the Drone Scheduling Sub-Problem (DSSP). Although it is possible to solve this directly with a MILP solver by fixing variables x_i^k and y_{ij}^k of the DTDC model in Section 3 to conform to the truck routing scheme, this approach is computationally inefficient due to the high-dimensional nature of the problem. As a result, we propose an alternative MILP formulation that is more compact.

A directed graph represents the situation denoted as $\mathcal{G} = (\mathcal{N}, \mathcal{A})$, where the set of arcs $A_k = \{(i,j)\}$ corresponds to the predefined routes for the trucks. Let C_k and C_u represent the sets of demands that are serviced by trucks and drones, respectively, where C_u is obtained by removing the demands in C_k from the set C. To align with the decision variables specified in Section 3, it is necessary to divide each drone operation (i, w, j) into two separate arcs, namely (i, w) and (w, j). For the DSSP, we can express the compact MILP formulation as follows:

$$\min \sum_{i\in C} w_i, \tag{43}$$

s.t.

$$w_i \geq t_i - T_i, \forall i \in C, \tag{44}$$

$$\sum_{j\in C_k, j\neq i} y^u_{i,j} = 1, \forall i \in C_u, \tag{45}$$

$$\sum_{j\in C_k, j\neq i} y^u_{j,i} = 1, \forall i \in C_u, \tag{46}$$

$$l_i + d_{i,j} \leq t_j, \forall i \in \mathcal{N}, \forall (i,j) \in A_k, \tag{47}$$

$$t_i + \hat{d}_{i,j} - M\left(1 - y^u_{i,j}\right) \leq t_j, \forall i \in C_k, j \in C_u, \tag{48}$$

$$t_i + t^i_{serve} \leq l_i, \forall i \in C, \tag{49}$$

$$l_j + \hat{d}_{j,i} - M\left(1 - y^u_{j,i}\right) \leq l_i, \forall i \in C_k, j \in C_u, \tag{50}$$

$$q_i \geq \sum_{s\in C_u, s\neq i} y^u_{i,s}, \forall i \in C_k, \tag{51}$$

$$q_i + \sum_{s\in C_u, s\neq i} \left(y^u_{s,i} - y^u_{i,s}\right) \leq N_0, \forall i \in C_k, \tag{52}$$

$$q_i + \sum_{s\in C, s\neq i} \left(y^u_{s,i} - y^u_{i,s}\right) = q_j, \forall (i,j) \in A_k, \tag{53}$$

$$\sum_{s\in C_k, s\neq i} \left(\hat{d}_{s,i} \cdot y^u_{s,i}\right) + \sum_{j\in C_k, j\neq i} \left(\hat{d}_{i,j} \cdot y^u_{i,j}\right) \leq L, \quad \forall i \in C_u. \tag{54}$$

The objective function, shown by Eq. (43), works with Eq. (44) to reduce the total service delay for all demands as much as possible by combining services provided by trucks and drones. This calculation incorporates implicit latencies that result from synchronization between trucks and drones. The constraints (45)–(54) have a resemblance to the constraints in the DTDC model discussed in Section 3, which are used to establish the necessary conditions for drone services, determine routing criteria, ensure time synchronization, and account for operational restrictions. This model provides a computational advantage by significantly reducing the constraints that use big-M coefficients. Specifically, only constraints (48) and (50) include such coefficients.

We define the *Lower Bound of Delay (LBD)* as the service delay produced from the ATRSP solution. The notation Y^u_π represents a feasible solution for the DSSP problem, with an associated objective value $f(Y^u_\pi)$. Due to the inherent interdependence between ATRSP and DSSP, it can be concluded that the inequality $LBD \leq f(Y^u_\pi)$ holds in all cases. Hence, in using optimization solvers such as Gurobi, the parameter denoted as *BestObjStop* can enhance the efficiency of the solution procedure by setting it to LBD. Consequently, the algorithm will cease execution whenever a solution is found that meets the condition $f(Y^u_\pi) \leq LBD$.

4.4. Tabu mechanism

The iterative process of solving the ATRSP and DSSP must overcome two major obstacles to efficiently solve the DTDC scheduling problem. First, the DSSP may be infeasible due to drone numbers and endurance limits, even with a feasible truck routing scheme. Second, repeatedly examining the same truck routing scheme results in redundant computation and local optima. We implement a tabu mechanism to address these challenges. According to the DSSP's feasibility status, the mechanism implements two types of tabu constraints: *Feasibility Constraints* and *Optimality Constraints*. We then modify the ATRSP to incorporate these constraints, guiding subsequent iterations towards more effective DTDC solutions.

4.4.1. Feasibility constraints

The tabu mechanism deems truck routing schemes that generate DSSP infeasibilities impermissible, preventing their recurrence. Consequently, the ATRSP incorporates the following constraints:

$$\sum_{r\in\overline{R}} z_r \leq |\overline{R}| - 1. \tag{55}$$

By prohibiting the concurrent selection of all routes in $\overline{R}$, this constraint reduces the likelihood that DSSP will become unfeasible.

Additionally, the DSSP model's structure is set by Eqs. (43)–(54), and the constraints (47) and (53) are completely based on the truck routing scheme π. The demand allocation outcomes only impact the remaining constraints, represented as C_u and C_k. Remarkably, constraint (47) influences only on the objective function, leaving the model's feasibility unaffected. Upon closer examination, two main factors often contribute to the DSSP's infeasibility: low drone endurance and insufficient drone quantities. Therefore, it is most likely that making changes to π without adjusting the demand allocation will provide little or no desired outcomes. In order to further augment the feasibility constraint, we propose an additional heuristic constraint:

$$\sum_{i\in C_u(\bar{X}_k)} x^k_i - 1 \geq 0, \tag{56}$$

where $C_u(\bar{X}_k) = \{i \in C \mid \bar{x}^k_i = 1\}$ signifies the set of demand points served by drones according to a predetermined truck routing strategy. The variable $\bar{X}_k$ represents the resulting allocation of demands. This constraint necessitates the assignment of at least one demand point in $C_u(\bar{X}_k)$ to a truck, hence increasing the probability of obtaining a suitable solution. Furthermore, the following proposition can arise:

Proposition 2. *Let $C'_u \subset C_u$. The feasibility constraints for both sets then satisfy the following relationship: the constraint for C'_u, denoted as cut_2, dominates the constraint for C_u, denoted as cut_1.*

Proof. When $\sum_{i\in C'_u} x^k_i - 1 \geq 0$, it follows that

$$\sum_{i\in C_u} x^k_i - 1 = \sum_{i\in C'_u} x^k_i - 1 + \sum_{i\in(C_u - C'_u)} x^k_i \geq \sum_{i\in(C_u - C'_u)} x^k_i \geq 0. \tag{57}$$

4.4.2. Optimality constraints

In order to reduce computational redundancy, we record the minimum total delay associated with the solution $\overline{Y_u}$ generated by a certain routing scheme. In the next optimality constraint, we encapsulate this record-keeping:

$$s \geq f_{\overline{R}}(\overline{Y_u}) \cdot \left(\sum_{r\in\overline{R}} z_r - |\overline{R}| + 1\right), \tag{58}$$

where $\overline{R} = \{r | z_r = 1\}$ denotes the set of selected routes of the truck routing scheme originating from ATRSP. This condition ensures that future exploration of the routing scheme must improve upon the previously recorded minimum total delay.

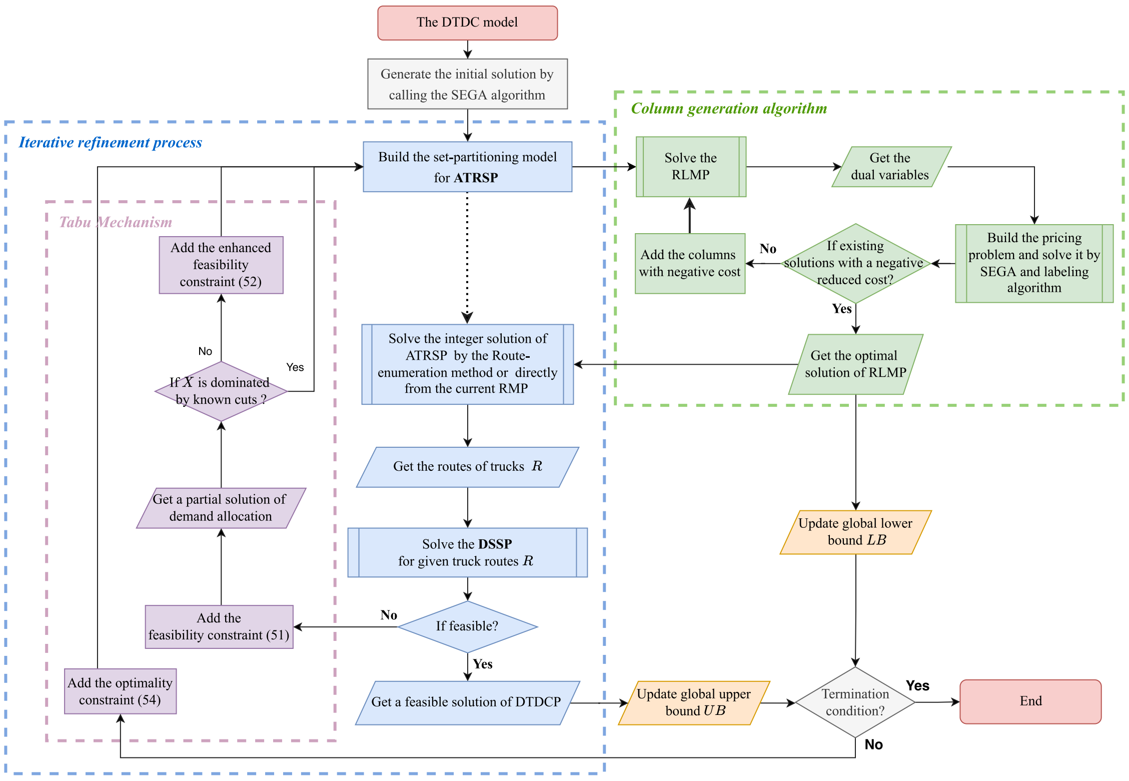

The DTDC model
Generate the initial solution by calling the SEGA algorithm
Iterative refinement process
Build the set-partitioning model for ATRSP
Tabu Mechanism
Add the enhanced feasibility constraint (52)
No
Yes
If X is dominated by known cuts ?
Get a partial solution of demand allocation
Add the feasibility constraint (51)
Add the optimality constraint (54)
Solve the integer solution of ATRSP by the Route-enumeration method or directly from the current RMP
Get the routes of trucks R
Solve the DSSP for given truck routes R
If feasible?
Get a feasible solution of DTDCP
Column generation algorithm
Solve the RLMP
Get the dual variables
Build the pricing problem and solve it by SEGA and labeling algorithm
If existing solutions with a negative reduced cost?
Add the columns with negative cost
Get the optimal solution of RLMP
Update global lower bound LB
Update global upper bound UB
Termination condition?
End

Fig. 3. Algorithm's process of the proposed matheuristic.

4.5. The overall of matheuristic

We decompose the DTDC scheduling problem into two interrelated sub-problems: the ATRSP and the DSSP. The DTDC scheduling problem is tackled using a matheuristic that applies an iterative paradigm to solve the sub-problems separately. This section provides a detailed explanation of the algorithm's procedure, demonstrating the interrelationships between the sub-problems and the approach used to solve them.

Fig. 3 shows that the algorithm starts using a genetic algorithm to develop an initial ATRSP solution. It then generates the related set-partitioning model. To enhance the scalability of computing performance and achieve an efficient solution for this model, we employ a column generation technique. Furthermore, we employ a genetic algorithm to enhance the efficiency of our solution for the pricing problem. We employ two distinct procedures to obtain an integer solution for the ATRSP. These strategies aim to balance computational efficiency and the solution's effectiveness. We construct the DSSP model using the integer solution from the ATRSP. We then directly solve this model using an available MILP solver. Following that, the ATRSP set-partitioning model incorporates two types of tabu constraints throughout the subsequent iterative process.

The iterative process terminates under one of the following conditions: (1) the upper bound (UB) from the DSSP equals the lower bound (LB) from the RLMP, indicating convergence to the optimal solution; (2) the maximum number of iterations is reached; (3) no improvement is observed after a predefined number of iterations; or (4) the computational time limit is exceeded.

The proposed matheuristic share similarities with well-known methods, such as logic-based Benders decomposition and row-and-column generating algorithms. For example, we can compare the ATRSP and DSSP to the Benders master and subproblems. Additionally, we can conceptualize tabu constraints as logic-based Benders cuts or generated rows. Nevertheless, major differences exist. Our matheuristic approach primarily decomposes due to its intrinsic relational nature, independent of a specific mathematical model structure. Moreover, using our matheuristic framework enables enhanced adaptability in optimizing techniques without necessitating strict conditions for optimality. To sum up, our suggested matheuristic approach gets rid of the need to find a strong "inference dual" that can simulate LP duality for Benders subproblems that are not convex.

The primary algorithm, denoted as Heuristic-E, incorporates a column enumeration process (Baldacci et al., 2011), enabling the identification of optimal ATRSP solutions in each iteration. While effective, this process can be computationally intensive. To address this, we propose an alternative variant, Heuristic-H, which enhances computational efficiency by omitting the column enumeration process and directly computing integer solutions for the current RLMP.

Regarding the computational complexity, the DTDC problem is $\mathcal{NP}$-hard, as established in prior work (Long et al., 2023). Consequently, unless $\mathcal{NP} = \mathcal{P}$, solving it in polynomial time is infeasible. To address this challenge, we adopt a hybrid approach that combines mathematical programming and heuristic techniques to achieve near-optimal solutions within a reasonable timeframe. It is hard to do a precise complexity analysis of the proposed heuristic and iterative algorithms because of many factors that affect them, such as the size of the problem, the number of iterations, and the settings for the parameters. However, we give an estimate of the complexity for the main parts. The ATRSP employs a set-partitioning model with a column generation approach, where the master problem's complexity is approximately $\mathcal{O}(2^R)$, with R representing the number of routes in the restricted master problem. The pricing problem, formulated as a shortest path problem with resource constraints, has a complexity of at least $\mathcal{O}(N \times M)$, where N and M are the numbers of nodes and arcs, respectively. The DSSP, modeled as a MILP, has an approximate complexity of $\mathcal{O}(2^n)$, where n is the number of demand points. The binary decision variables give rise to this complexity, but targeted problem decomposition and a focus on essential operations and constraints reduce the computational burden. The iterative matheuristic framework alternates between ATRSP and

DSSP, resulting in an overall complexity of $\mathcal{O}(k\cdot(\mathcal{O}(\text{ATRSP})+\mathcal{O}(\text{DSSP})))$, where k denotes the number of iterations. Despite the inherent complexity of these components, the proposed framework demonstrates significant scalability and computational efficiency, making it well-suited for large-scale emergency scenarios that demand rapid and adaptive scheduling solutions.

Finally, while our proposed algorithm shares a foundation in Tabu Search (TS) with the TSI algorithm from Long et al. (2023), key differences emerge in design philosophy and structure. The TSI algorithm emphasizes exploration and exploitation through adaptive tabu tenure and a specialized neighborhood structure. In contrast, our approach integrates mathematical programming with heuristic techniques, using set-partitioning for ATRSP and a MILP framework for DSSP. This decomposition strategy facilitates solving smaller, manageable subproblems, improving efficiency and solution quality. Our framework is more scalable and reliable because it uses genetic algorithms and labeling techniques to come up with initial solutions. It works especially well in complicated emergency situations that need quick and accurate coordination.

5. Computational study

Following the rules of the Design of Experiments (DoE) (Antony, 2023), a methodical approach was used to check how key parameters affected the performance of the suggested algorithm. The experimental design targeted critical factors such as drone endurance, truck–drone interaction, and computational resource utilization, which collectively encapsulate the scheduling complexity and synchronization challenges inherent in coordinating trucks and drones under diverse emergency scenarios. To ensure the findings' practical relevance, the experimental setup simulated real-world emergency conditions, including disruptions in road networks and varying drone endurance levels. This comprehensive testing framework enabled the derivation of meaningful insights into the algorithm's robustness, adaptability, and operational efficiency while maintaining computational feasibility. By balancing analytical depth with the practical constraints of computational complexity and resource availability, the approach ensured rigorous evaluation and practical applicability.

To further support the analysis, the computational environment and benchmark instances used in the study were carefully detailed. All experiments were conducted on a machine equipped with an Apple Silicon M1 Pro processor and 16 GB of RAM, operating on 64-bit MacOS 13.0. The algorithms were implemented in Python 3.9, with Gurobi 10.0 used to solve the MILP problems in single-threaded mode. Customized solver settings (`MIPFocus=2`, `Method=1`) were employed, and Gurobi's callback function was utilized to terminate unpromising processes based on current solution bounds, thereby enhancing computational efficiency.

5.1. Experimental settings

5.1.1. Benchmark instances

We employed a benchmark instance set for the VRP with delivery robots, as outlined by Chen, Demir, and Huang (2021). These instances, generated from Cardiff, UK postcodes, vary in size from 15 to 200 demand points, distributed across seven subsets. We adapted these instances to align with the current study by manipulating the "Ready Time" and "Service Time" parameters. The instances are labeled as $Cardiff\_n\_k\_o$, where n, k, and o denote the number of demand points, the number of trucks, and the instance number, respectively.

5.1.2. Parameter settings

We selected the parameters for this study following existing research and industry standards for vehicle and drone delivery. Specifically, we initially equipped each truck with two drones, up to a maximum allowable capacity of three drones. The average speed of the trucks in the city is set at 8.5 m/s. Drone parameters were based on the specifications of the Ark Octocopter Drone (https://www.sf-tech.com.cn/product/h4-quadcoper-drone), with a cruising speed of 15 m/s and a maximum range of $L = 10$ km.

5.1.3. Benchmark methods

We evaluate the performance of the proposed matheuristic framework through four benchmark methods tailored for the DTDC scheduling problem. The benchmarks include exact optimal solutions from Gurobi for smaller instances and three well-known metaheuristics – Tabu Search (TS), Variable Neighborhood Search (VNS), and Simulated Annealing (SA) – for medium to large instances. These methods were chosen for their adaptability and effectiveness in recent studies of similar problems (Kuo, Lu, Lai, & Mara, 2022; Long et al., 2023; Salama & Srinivas, 2022; Wu et al., 2022). Modifications were necessary for these algorithms to accommodate the unique aspects of the DTDC problem, such as drone parking slot availability on trucks. The adaptation process is detailed in our previous publication (Long et al., 2023), focusing on neighborhood design and iterative control strategies.

5.2. Algorithm performance

In this section, we evaluate the performance of two matheuristic algorithms, namely Heuristic-E and Heuristic-H, focusing on accuracy and efficiency, respectively. We begin by comparing Heuristic-E against the MILP model solved by Gurobi across instances of various sizes. We set a time limit of 3600 s for each run of Heuristic-E and Gurobi. Next, we compare Heuristic-H's performance to other metaheuristics, such as Tabu Search (TS) (Long et al., 2023), Variable Neighborhood Search (VNS) (Kuo et al., 2022; Salama & Srinivas, 2022), and Simulated Annealing (SA) (Wu et al., 2022).

5.2.1. Performance comparison between heuristic-e and gurobi

Table 3 presents the computational results for small-sized instances. The columns labeled "UB" and "Gap" represent the optimal upper bounds and the relative gap between the best upper and lower bounds attained by Heuristic-E and Gurobi, respectively. "Time" denotes the computational time in seconds needed to solve an instance, and "Opt_Time" represents Gurobi's computational time to find an optimal solution, assuming no proof of optimality is required. We define $\Delta_{UB} = \frac{UB_{Gurobi}-UB_{Heuristic}}{UB_{Gurobi}} \times 100$ and $\Delta_{LB} = \frac{LB_{Gurobi}-LB_{Heuristic}}{LB_{Gurobi}} \times 100$. A positive Δ_{UB} implies that Heuristic-E generates a more favorable upper bound, whereas a negative Δ_{LB} signifies a more accurate lower bound from Heuristic-E. Both algorithms quickly find optimal solutions for all instances, averaging the results for instances with 10 demand points.

Heuristic-E and Gurobi solve instances with 10 demand points to optimality within limited computational time. Heuristic-E averages a computational time of 1.2 s, roughly six times faster than Gurobi, which averages 7.8 s. For instances with 15 demand points and two trucks, Heuristic-E finds optimal solutions within a 3600-second time limit for all instances. The most computationally intensive instance (*Cardiff_15_2_15*) takes at most 737.5 s with an optimality gap of 0.07%, well within the acceptable range. In contrast, Gurobi only finds optimal solutions for 16 instances and verifies optimality for just two. It also yields suboptimal lower bounds with an average optimality gap of 61.15%. Comparing only the optimal computational times, Gurobi takes over eight times longer than Heuristic-E, averaging 57.0 s.

Heuristic-E's superiority becomes more apparent when there are 20 demand points and three trucks. It resolves all instances to optimality in an average of 17.7 s. Conversely, Gurobi optimizes only 14 instances, taking an average of 1126.9 s – approximately 60 times slower than Heuristic-E. Gurobi verifies optimality for merely three instances, with an average optimality gap of 40.56%. Additionally, we graphically represent the lower bound obtained for the $Cartiff\_15$ instances, as shown in Fig. 4. The Heuristic-E algorithm consistently outperforms Gurobi in

Table 3
Performance comparison between Heuristic-E and Gurobi on small-size instances.

Instance	Heuristic-E				Gurobi						
	UB	LB	Gap(%)	Time(s)	UB	LB	Gap(%)	Time(s)	Opt_Time(s)	Δ_{UB}(%)	Δ_{LB}(%)
10 demand points with two trucks											
Both methods are capable of finding the optimum for all instances within a short time; thus, only the average results are presented here.											
Average	**300.66**	**300.66**	**0.00**	**1.2**	**300.66**	**300.66**	**0.00**	**7.8**	**4.88**	**0.00**	**0.00**
15 demand points with two trucks											
Cardiff_15_2_01	146.84	146.84	0.00	12.4	146.84	146.84	0.00	253.8	80.7	0.00	0.00
Cardiff_15_2_02	1121.54	1121.54	0.00	27.8	1121.54	204.51	81.77	3600.1	647.5	0.00	−448.41
Cardiff_15_2_03	489.94	489.94	0.00	21.2	489.94	204.89	58.18	3623.6	88.9	0.00	−139.12
Cardiff_15_2_04	1253.78	1253.78	0.00	38.9	1253.78	244.06	80.53	3603.9	2587.5	0.00	−413.73
Cardiff_15_2_05	931.55	931.55	0.00	24.7	931.55	226.47	75.69	3625.2	448.7	0.00	−311.33
Cardiff_15_2_06	311.93	311.93	0.00	22.6	311.93	254.88	18.29	3617.6	71.3	0.00	−22.39
Cardiff_15_2_07	984.56	984.56	0.00	36.2	984.56	211.66	78.50	3600.5	410.4	0.00	−365.17
Cardiff_15_2_08	491.37	491.37	0.00	14.6	520.29	216.51	58.39	3604.3	–	5.56	−126.95
Cardiff_15_2_09	1313.00	1313.00	0.00	24.0	1313.00	249.25	81.02	3603.2	471.4	0.00	−426.78
Cardiff_15_2_10	947.72	947.72	0.00	20.5	1036.83	234.40	77.39	3618.3	–	8.59	−304.32
Cardiff_15_2_11	590.57	590.57	0.00	23.9	590.57	238.62	59.60	3611.3	997.9	0.00	−147.50
Cardiff_15_2_12	565.45	565.45	0.00	17.6	565.45	202.97	64.10	3602.8	82.2	0.00	−178.58
Cardiff_15_2_13	205.27	205.27	0.00	12.4	205.27	205.27	0.00	518.1	342.4	0.00	0.00
Cardiff_15_2_14	1471.49	1471.49	0.00	26.3	1783.90	253.31	85.80	3600.0	–	17.51	−480.90
Cardiff_15_2_15	1365.79	1364.84	0.07	737.5	1427.83	228.91	83.97	3600.0	–	4.34	−496.23
Cardiff_15_2_16	644.22	644.22	0.00	12.2	644.22	167.29	74.03	3600.0	31.3	0.00	−285.08
Cardiff_15_2_17	988.02	988.02	0.00	16.1	988.02	290.97	70.55	3600.0	179.1	0.00	−239.56
Cardiff_15_2_18	977.88	977.88	0.00	19.1	977.88	231.81	76.29	3600.0	581.4	0.00	−321.84
Cardiff_15_2_19	696.28	696.28	0.00	18.1	696.28	202.99	70.85	3600.0	294.0	0.00	−243.00
Cardiff_15_2_20	401.76	401.76	0.00	13.5	401.76	289.45	27.96	3600.0	406.4	0.00	−38.80
Average	**794.95**	**791.91**	**0.38**	**57.0**	**819.57**	**225.25**	**61.15**	**3284.1**	**482.6**	**1.80**	**−249.48**
20 delivery points with three trucks											
Cardiff_20_3_01	198.19	198.19	0.00	15.7	198.19	165.52	16.48	3600.1	3163.9	0.00	−19.73
Cardiff_20_3_02	212.78	212.78	0.00	16.8	212.78	178.36	16.18	3600.0	1704.3	0.00	−19.30
Cardiff_20_3_03	444.33	444.33	0.00	14.6	523.41	163.47	68.77	3600.1	–	15.11	−171.81
Cardiff_20_3_04	526.76	526.76	0.00	14.7	614.08	205.94	66.46	3600.0	–	14.22	−155.78
Cardiff_20_3_05	160.34	160.34	0.00	19.3	160.34	160.34	0.00	597.7	438.9	0.00	0.00
Cardiff_20_3_06	338.73	338.73	0.00	16.7	338.73	194.13	42.69	3600.0	230.3	0.00	−74.48
Cardiff_20_3_07	319.22	319.22	0.00	14.7	319.22	206.04	35.46	3600.1	971.1	0.00	−54.94
Cardiff_20_3_08	619.46	619.46	0.00	18.4	640.18	174.75	72.70	3600.0	–	3.24	−254.49
Cardiff_20_3_09	279.81	279.81	0.00	14.7	279.81	209.71	25.05	3600.0	338.1	0.00	−33.43
Cardiff_20_3_10	383.59	383.59	0.00	12.1	383.59	189.58	50.58	3600.0	1152.7	0.00	−102.34
Cardiff_20_3_11	190.94	190.94	0.00	19.0	190.94	190.78	0.08	1829.4	1756.8	0.00	−0.08
Cardiff_20_3_12	1346.42	1346.42	0.00	26.2	1346.38	143.94	89.31	3600.0	2246.5	0.00	−835.38
Cardiff_20_3_13	185.36	185.36	0.00	14.7	185.36	172.74	6.81	3600.0	260.4	0.00	−7.31
Cardiff_20_3_14	609.55	609.55	0.00	18.8	747.18	130.39	82.55	3600.0	–	18.42	−367.47
Cardiff_20_3_15	563.14	563.14	0.00	16.4	563.14	231.48	58.90	3600.0	624.0	0.00	−143.28
Cardiff_20_3_16	487.61	487.61	0.00	16.4	559.71	206.22	63.16	3600.0	–	12.88	−136.45
Cardiff_20_3_17	234.62	234.62	0.00	18.7	234.62	182.43	22.24	3600.0	505.3	0.00	−28.61
Cardiff_20_3_18	336.55	336.55	0.00	20.9	408.58	212.80	47.92	3600.0	–	17.63	−58.15
Cardiff_20_3_19	176.59	176.59	0.00	22.1	176.59	176.49	0.05	876.6	860.3	0.00	−0.05
Cardiff_20_3_20	332.45	332.45	0.00	23.7	332.45	180.35	45.75	3600.0	1524.2	0.00	−84.34
Average	**397.32**	**397.32**	**0.00**	**17.7**	**420.76**	**183.77**	**40.56**	**3225.2**	**1126.9**	**4.07**	**−127.37**

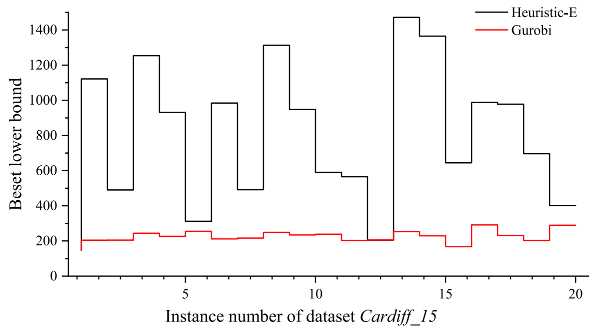


Fig. 4. A graphic representation of the lower bounds of the *Cardiff*_15 instances presented in Table 3.

achieving lower bounds and has the potential to serve as a benchmark for assessing future algorithm enhancements and improvements.

Table 4 shows Heuristic-E's performance for medium-sized instances, specifically those with 25 and 35 demand points and three and four trucks, respectively. Heuristic-E consistently solves all instances with 25 demand points and three trucks to optimality, with all gap values falling within the 0.1% permissible error range. Most instances resolve within 60 s, with an average computational time of 257 s. For the 35-delivery-point instances with four trucks, Heuristic-E solves six of ten instances to optimality within the 0.1% permissible error range, and the computation time is only approximately 200 s for them, but instance *Cardiff*_35_4_06. The algorithm falls short of optimality, for instance, *Cardiff_35_4_08*, yet it maintains a minimal gap of 1.08%. For the remaining instances, the gap is around 10%.

Overall, our heuristic-E significantly outperforms Gurobi in terms of computational time across all instances. This trend continues in larger instances, where Heuristic-E maintains superior speed and efficiency. Heuristic-E consistently achieves optimal solutions within a 0.1% permissible error range for medium-sized instances (25 and 35 demand points). In contrast, Gurobi struggles with larger instances, exhibiting substantial optimality gaps. Heuristic-E demonstrates a robust ability to find and verify optimal solutions across a broader range of instances than Gurobi. This advantage may be achieved by integrating decomposition, mathematical programming, and heuristic techniques, which enhance the practicality and effectiveness of our solution.

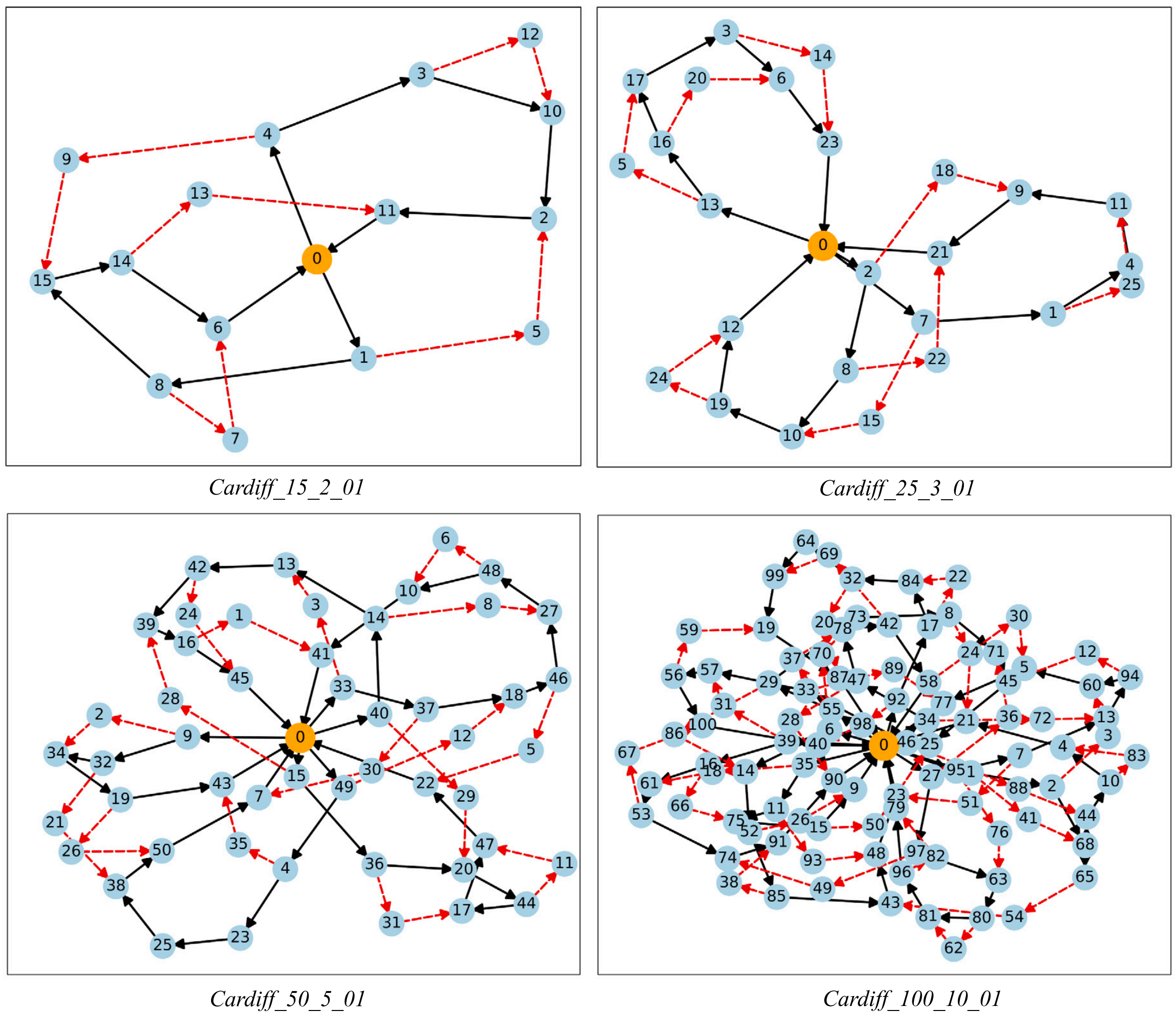


Fig. 5. Visualization results of the Heuristic-H solutions on different-sized instances.
Note: The spatial layout is generated using the ‘kamada_kawai_layout‘ function in NetworkX for illustrative purposes and does not represent actual physical distances.

5.2.2. Performance comparison between heuristic-h and metaheuristic algorithm

While the Heuristic-E algorithm excels at providing optimal solutions for small to medium-sized instances, its computational burden may be impractical for real-world applications. We introduce a variant, Heuristic-H, designed within the flexible mathematic framework, emphasizing computational efficiency as a remedy. Fig. 5 depicts a graphical representation of the solutions generated by the Heuristic-H algorithm on different-sized instances. The diagram illustrates the complex interaction between truck and drone routes in the DTDC strategy. The truck's route, represented by a solid black line, serves as the backbone of the delivery network. Concurrently, the drone's path, represented by a red dashed line, exhibits a more dynamic and flexible pattern. The depot and demand points are identified by orange and blue circles, respectively.

We comprehensively evaluated instances with 25, 50, 100, and 200 demand points using multiple heuristics. Results are tabulated in Table 5, where columns “Avg_Obj” and “Avg_Time” represent the average objective value and computational time across 20 instances, respectively. Each instance result is an optimum achieved through five independent runs, thereby mitigating the influence of heuristic instability. The ‘Gap’ column is computed as $(Obj_n - Obj_m)/Obj_m \times 100$, where Obj_m represents the lowest objective values among the four algorithms under consideration. Our results reveal that Heuristic-H consistently surpasses the other three metaheuristics, achieving at least a 50% improvement while reducing nearly half the computational time, except for instances with 200 demand points. Although all four algorithms exceed the time limit when dealing with instances of 200 demand points, Heuristic-H significantly outperforms the others, lowering the response delay by 77.80%, 280.19%, and 369.34%, respectively. Based on the data provided in the table, Fig. 6 visually illustrates the average objective. Empirical evidence demonstrates that Heuristic-H consistently yields lower average outcomes across different scenarios. Furthermore, as the size of the tested instance grows, the corresponding increase in the objective value is comparatively minimal when compared to alternative algorithms.

Given the inherent stochastic nature of heuristic algorithms, we conducted a stability analysis across the four heuristics under study. Fig. 7 displays the results of 10 independent runs for each instance. Our analysis reveals that Heuristic-H yields superior solutions and demonstrates marked stability compared to the other metaheuristics. Heuristic-H's matheuristic nature leverages the mathematical intricacies of the problem, giving it a stability advantage over general-purpose metaheuristics that often ignore these intrinsic relationships.

The comparative results highlight Heuristic-E's effectiveness in solving small to medium-sized instances with high efficiency and minimal optimality gaps. For larger instances, Heuristic-H emerges as a highly efficient and stable solution, outperforming metaheuristic algorithms. The success of Heuristic-H can be attributed to its matheuristic approach, which capitalizes on the problem's mathematical aspects to

Table 4
Performance of Heuristic-E on medium-size instances.

Instance	UB	LB	Gap (%)	Time (s)
25 demand points with three trucks				
Cardiff_25_3_01	701.19	701.19	0.00	91.6
Cardiff_25_3_02	589.13	589.13	0.00	167.6
Cardiff_25_3_03	1118.20	1118.20	0.00	165.1
Cardiff_25_3_04	793.04	793.04	0.00	31.8
Cardiff_25_3_05	414.26	414.26	0.00	48.0
Cardiff_25_3_06	604.87	604.87	0.00	57.4
Cardiff_25_3_07	641.75	641.75	0.00	40.2
Cardiff_25_3_08	1298.27	1298.27	0.00	54.4
Cardiff_25_3_09	732.07	732.07	0.00	37.9
Cardiff_25_3_10	778.99	778.47	0.07	1876.0
Average	**767.18**	**767.13**	**0.01**	**257.0**
35 demand points with four trucks				
Cardiff_35_4_01	1489.81	1488.33	0.10	228.2
Cardiff_35_4_02	814.97	814.97	0.00	134.4
Cardiff_35_4_03	418.49	418.49	0.00	83.9
Cardiff_35_4_04	2148.59	1939.84	9.72	3581.2
Cardiff_35_4_05	1995.16	1727.02	13.44	3718.8
Cardiff_35_4_06	1425.89	1425.35	0.04	2821.7
Cardiff_35_4_07	658.22	658.22	0.00	112.2
Cardiff_35_4_08	1592.31	1575.14	1.08	3600.2
Cardiff_35_4_09	2045.73	1833.47	10.38	3623.2
Cardiff_35_4_10	842.45	842.45	0.00	206.0
Average	**1343.16**	**1268.74**	**3.47**	**1811.0**

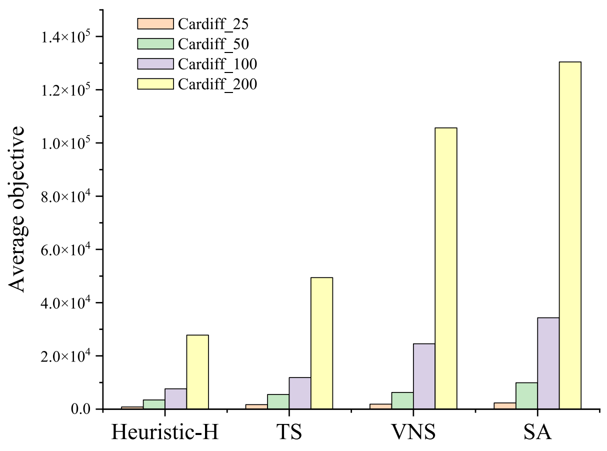


Fig. 6. A graphic representation of the average objective values presented in Table 5.

Table 5
Performance comparison between Heuristic-H and metaheuristic algorithms on large-size instances.

Datasets	Algorithms	Avg_Obj	Avg_Time (s)	Gap (%)
Cardiff_25_3	Heuristic-H	819.89	51.1	0.00
	TS	1708.82	213.1	108.42
	VNS	1876.61	289.0	128.89
	SA	2360.41	169.4	169.4
Cardiff_50_5	Heuristic-H	3476.32	109.3	0.00
	TS	5532.85	300.4	59.16
	VNS	6266.04	300.6	80.25
	SA	9906.42	304.62	184.97
Cardiff_100_10	Heuristic-H	7639.02	327.1	0.00
	TS	11 874.20	600.9	55.44
	VNS	24 560.97	606.4	221.52
	SA	34 326.78	600.1	349.36
Cardiff_200_20	Heuristic-H	27 795.57	3680.98	0.00
	TS	49 419.99	3611.49	77.80
	VNS	105 675.18	3611.87	280.19
	SA	130 455.98	3512.24	369.34

achieve superior performance and stability. This comparison highlights the importance of selecting appropriate heuristic strategies based on the size and complexity of the problem to achieve optimal solutions efficiently.

5.3. Adaptability to road damage

During emergencies, the destruction of road networks often impairs truck accessibility, obstructing emergency services' responses. Hence, it is crucial to effectively handle road network disruptions and guarantee the rapidity of emergency response. This subsection presents a comparative analysis using four different scenarios. Each scenario consists of 50 demand points, specifically referred to as $Cardiff\_50\_5\_1$, $Cardiff\_50\_5\_2$, $Cardiff\_50\_5\_3$, and $Cardiff\_50\_5\_4$. This analysis aims to assess the adaptive abilities of the DTDC and the mFSTSP (Murray & Raj, 2020) when faced with different levels of road network damage. The analysis includes four road network impairment levels: 10%, 20%, 30%, and 40%. These levels indicate the percentage of roads impassable for trucks due to damage. We conducted ten simulations for each scenario and damage level. The results, shown in Fig. 8, show a complete comparison of the total service delay across all 40 simulations for both DTDC and mFSTSP collaboration strategies.

The experimental results indicate that the DTDC strategy outperforms the mFSTSP strategy regarding service delay performance across various scenarios. Specifically, the DTDC strategy improves emergency response efficiency significantly. Both strategies experience more service delays as road damage worsens, but the DTDC strategy sees a more moderate rise. The average service delay under the DTDC strategy increases from 6378 s with 10% road damage to 11,416 s at a 40% damage level. In contrast, the mFSTSP strategy results in a significant increase in average service delay of 65,298 s. The dynamic collaboration between trucks and drones facilitates a flexible scheduling scheme and enhances the system's adaptability and robustness.

5.4. Sensitivity analysis

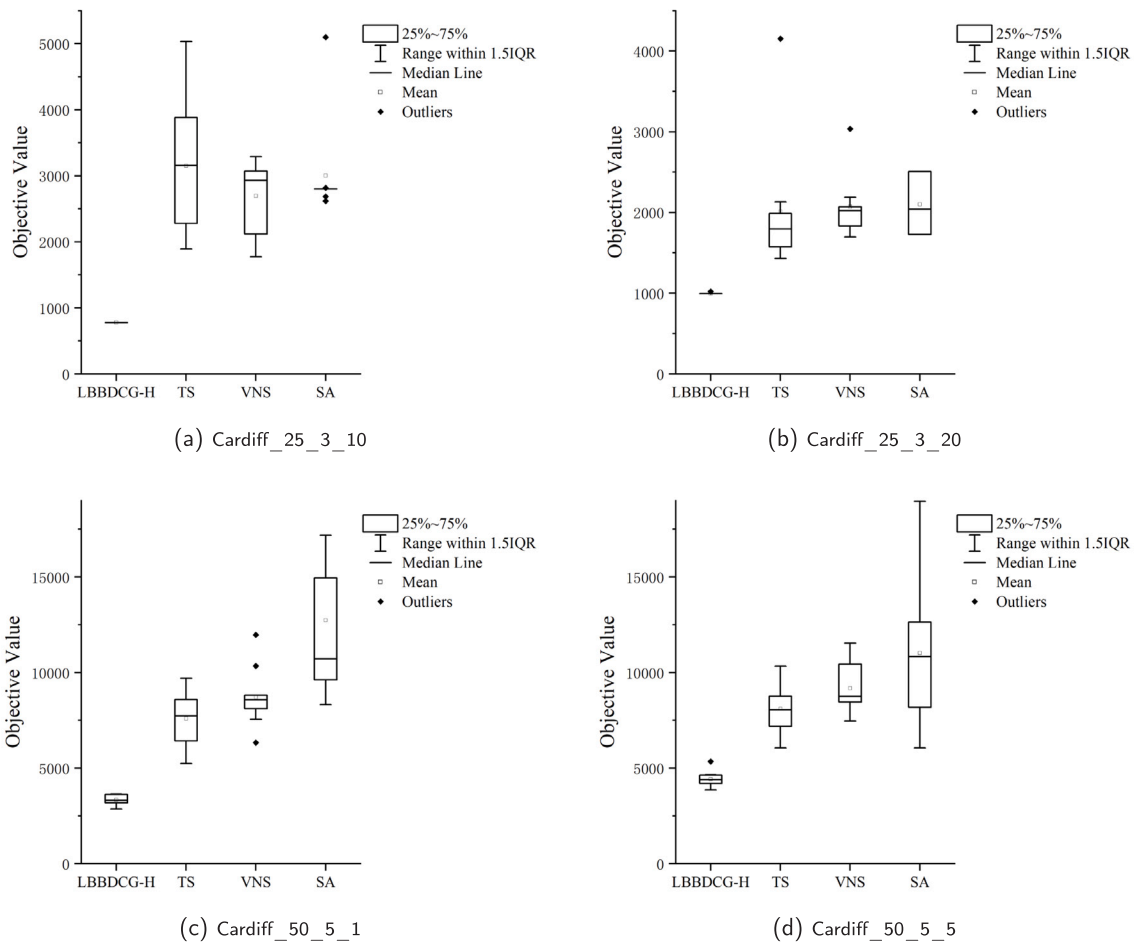
The sensitivity analysis subsection delves into the operational intricacies of the DTDC scheduling problem, focusing on the mileage endurance of drones and the impact of varying the number of drones per truck. These analyses are crucial for understanding how different operational parameters influence the efficiency and effectiveness of the DTDC strategy.

5.4.1. Mileage endurance of drones

The endurance mileage constrains drone reachability in the DTDC strategy, thereby influencing its performance. We define e as a parameter indicating the proportion of arcs a drone can traverse in a round trip based on its endurance mileage (e.g., $e = 0.5$ corresponds to 50% coverage). We systematically examined e, varying it between 0.2 and 1.0 in increments of 0.1. The outcomes, depicted in Fig. 9, reveal a direct correlation between increased drone endurance (e) and improved system performance, as evidenced by lower objective values. This indicates that drones with greater endurance have higher flexibility in performing tasks, thus improving the overall efficiency of the emergency system through optimized scheduling. Despite the positive correlation between drone endurance and system performance, we note diminishing returns when e exceeds 0.7. Beyond this range, the increased endurance does not lead to comparable improvements in efficacy. This phenomenon can be attributed to the outstanding flexibility of the DTDC strategy and the use of an excellent scheduling optimization algorithm. This combination enables the system to approximate the optimal state without necessitating 100% coverage of drone endurance.

5.4.2. Impact of drone numbers on trucks

The unique feature of the DTDC strategy, which enables drones to take off and land on multiple vehicles, is further explored. This



(a) Cardiff_25_3_10 (b) Cardiff_25_3_20 (c) Cardiff_50_5_1 (d) Cardiff_50_5_5

Fig. 7. Comparison of the solution stability between Heuristic-H and metaheuristic algorithms.

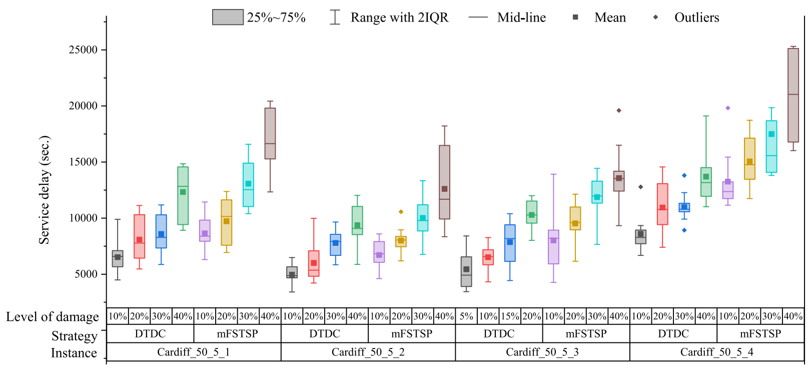


Fig. 8. Comparison of adaptability to road damage.

aspect introduces a new dimension to the problem: allocating drone parking slots on trucks. Through numerical experiments, the study investigates how the number of drones equipped on a truck (q_0) and the maximum allowable number of drones on a truck (N_0) affect the system's performance. A series of numerical experiments were conducted to examine this aspect, with Fig. 10 presenting results for two representative instances. Also, it is important to note that this part of the analysis is based on the idea that our Heuristic-E algorithm can find the best solutions to small problems by checking the upper and lower bounds enough times.

The figure shows that as the maximum allowable number of drones on a truck (N_0) increases, the objective value of the optimal scheme consistently decreases when the number of equipped drones (q_0) remains constant. However, it is important to note that an excessively large value of N_0 is not conducive to system performance. The impact of the number of drones equipped on a truck (q_0) is more intriguing. In most cases, the optimal solution does not improve as q_0 increases, which aligns with our intuition. For example, in instances such as $Cardiff$_15_2_6 and $Cardiff$_15_2_9, where $N_0 = 5$, increasing q_0 from 2 to 3 and then to 4 does not yield a better optimal solution. However, when $q_0 = N_0$, the trend becomes uncertain. In some cases, increasing the number of drones (q_0) may lead to a worse optimal solution. For instance, in the $Cardiff$_15_2_6 instance, increasing q_0 from 2 to 3 when $N_0 = 3$ or 3 to 4 when $N_0 = 4$ results in a suboptimal solution. On the other hand, in some cases, the previous trend is still

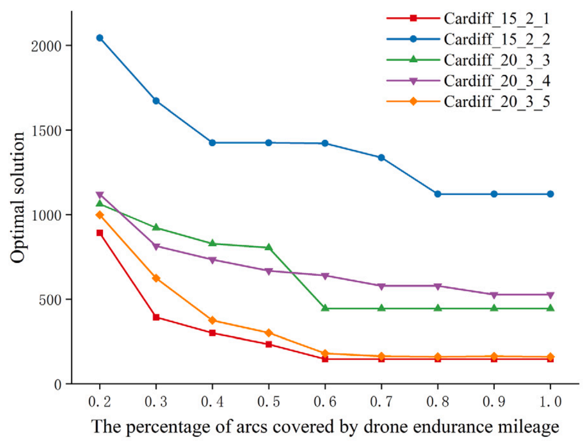


Fig. 9. The impact of endurance mileage.

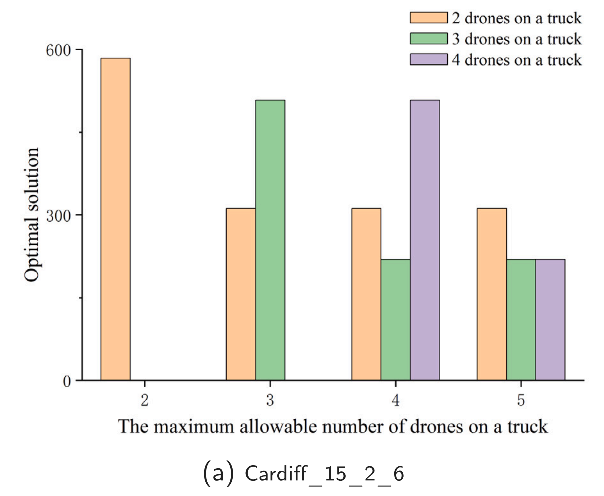
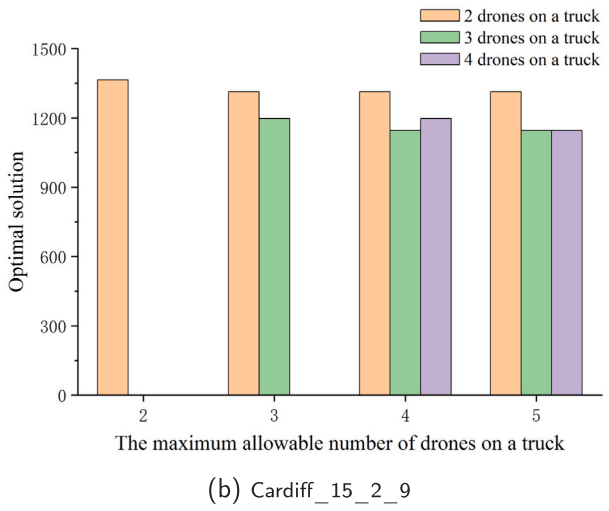



Fig. 10. The effect of the quantity and maximum allowable number of drones on a truck.

valid. For example, increasing q_0 from 2 to 3 when $N_0 = 3$ in the $Cardiff$_15_2_9 instance leads to a better optimal solution. The DTDC strategy's dynamic binding relationship between drones and trucks explains this phenomenon. When drones over-equip and remain idle, they may occupy parking slots, which may hinder otherwise possible drone use. As a result, reserving a few empty drone parking slots on the truck can potentially improve the efficiency of the DTDC strategy.

The sensitivity analysis highlights the importance of carefully considering operational parameters in the DTDC scheduling problem, such as drone endurance and the number of drones per truck. The findings highlight that while increasing drone endurance and the number of drones can enhance system performance, there are diminishing returns and the potential for resource underutilization beyond certain thresholds. These insights are valuable for optimizing the DTDC strategy, suggesting that a balanced approach to drone endurance and allocation can lead to more efficient and effective delivery systems.

6. Conclusions

This study investigated the DTDC scheduling problem, characterized by complex synchronization requirements between trucks and drones. We presented a 2-index MILP model to capture the intricate relationships between trucks and drones. To solve this problem, a specialized matheuristic approach was designed, utilizing the problem's inherent structure for efficient interfacing with any MILP solver. The approach employs a column generation algorithm for demand allocation and truck routing, enhanced by labeling and genetic algorithms. An iterative process facilitates further refinement of solutions, and a tabu mechanism prevents entrapment in local optima.

Two alternative matheuristic algorithms, namely Heuristic-E and Heuristic-H, were introduced to focus on solution accuracy and computational efficiency, respectively. The method's reliability is demonstrated by the fact that Heuristic-H performs better than well-known metaheuristics such as TS, VNS, and SA, with differences of 55.44%, 221.52%, and 349.36%, respectively. Additional findings include operational insights into drone capabilities; specifically, drones should have an endurance range covering at least 70% of the total round-trip graph distance for optimal DTDC implementation. We also found that reserving a minimum of one drone parking slot per truck enhances the effectiveness of the DTDC strategy.

While the proposed method demonstrates promising results in optimizing truck–drone collaboration for emergency logistics, several limitations warrant further investigation. The approach assumes ideal communication and synchronization between trucks and drones, which may be unrealistic in real-world scenarios characterized by network disruptions and operational uncertainties. Furthermore, it relies on distance-based routing, without considering dynamic environmental factors such as weather conditions or traffic, which can significantly influence drone performance. The computational complexity, particularly stemming from the MILP components, also presents challenges for large-scale instances, thereby limiting the method's applicability in real-time decision-making contexts.

Future research could focus on integrating advanced metaheuristic frameworks to enhance convergence and scalability. Developing more compact MILP formulations could further improve computational efficiency, enabling the method to handle larger and more complex scenarios. Additionally, incorporating adaptive collaboration mechanisms and real-time environmental data, such as weather forecasts and traffic patterns, could improve the robustness and scalability of the approach, making it more suitable for practical emergency logistics applications.

CRediT authorship contribution statement

Jinqiu Zhao: Conceptualization, Investigation, Methodology, Software, Formal analysis, Writing – original draft, Writing – review & editing. **Yuying Long:** Investigation, Methodology, Writing – original draft, Review. **Binglei Xie:** Conceptualization, Funding acquisition, Supervision, Writing – review. **Gangyan Xu:** Conceptualization, Supervision, Writing – review. **Yongwu Liu:** Validation, Visualization, Writing – review.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

References

Affenzeller, M. (2001). A new approach to evolutionary computation: Segregative genetic algorithms (SEGA). In *Connectionist models of neurons, learning processes, and artificial intelligence: 6th international work-conference on artificial and natural neural networks, IWANN 2001 Granada, Spain, June 13–15, 2001 proceedings, part 1 6* (pp. 594–601). Springer.

Afifi, S., Dang, D.-C., & Moukrim, A. (2015). Heuristic solutions for the vehicle routing problem with time windows and synchronized visits. *Optimization Letters*, *10*(3), 511–525.

Agatz, N., Bouman, P., & Schmidt, M. (2018). Optimization approaches for the traveling salesman problem with drone. *Transportation Science*, *52*(4), 965–981.

Al-Hilo, A., Samir, M., Assi, C., Sharafeddine, S., & Ebrahimi, D. (2020). UAV-assisted content delivery in intelligent transportation systems-joint trajectory planning and cache management. *IEEE Transactions on Intelligent Transportation Systems*, *22*(8), 5155–5167.

Amine Masmoudi, M., Mancini, S., Baldacci, R., & Kuo, Y.-H. (2022). Vehicle routing problems with drones equipped with multi-package payload compartments. *Transportation Research Part E: Logistics and Transportation Review*, *164*, Article 102757.

Antony, J. (2023). *Design of experiments for engineers and scientists*. Elsevier.

Archetti, C., & Speranza, M. G. (2014). A survey on matheuristics for routing problems. *EURO Journal on Computational Optimization*, *2*(4), 223–246.

Bakir, I., & Tiniç, G. Ö. (2020). Optimizing drone-assisted last-mile deliveries: The vehicle routing problem with flexible drones. (pp. 1–28). Optimization-Ouline. Org.

Baldacci, R., Mingozzi, A., & Roberti, R. (2011). New route relaxation and pricing strategies for the vehicle routing problem. *Operations Research*, *59*(5), 1269–1283.

Bian, J., Song, R., He, S., & Chi, J. (2024). Truck-drone hybrid delivery routing: A mathematical model and micro-evolutionary algorithm. *IEEE Transactions on Intelligent Transportation Systems*, 1–16.

Boccia, M., Mancuso, A., Masone, A., & Sterle, C. (2024). Exact and heuristic approaches for the truck–drone team logistics problem. *Transportation Research Part C (Emerging Technologies)*, *165*, Article 104691.

Boccia, M., Masone, A., Sforza, A., & Sterle, C. (2021). A column-and-row generation approach for the flying sidekick travelling salesman problem. *Transportation Research Part C (Emerging Technologies)*, *124*, Article 102913.

Bouman, P., Agatz, N., & Schmidt, M. (2018). Dynamic programming approaches for the traveling salesman problem with drone. *Networks*, *72*(4), 528–542.

Chen, C., Demir, E., & Huang, Y. (2021). An adaptive large neighborhood search heuristic for the vehicle routing problem with time windows and delivery robots. *European Journal of Operational Research*, *294*(3), 1164–1180.

Dayarian, I., Savelsbergh, M., & Clarke, J.-P. (2020). Same-day delivery with drone resupply. *Transportation Science*, *54*(1), 229–249.

Dell'Amico, M., Montemanni, R., & Novellani, S. (2021). Algorithms based on branch and bound for the flying sidekick traveling salesman problem. *Omega*, *104*, Article 102493.

Dell'Amico, M., Montemanni, R., & Novellani, S. (2022). Exact models for the flying sidekick traveling salesman problem. *International Transactions in Operational Research*, *29*(3), 1360–1393.

Di Puglia Pugliese, L., & Guerriero, F. (2017). Last-mile deliveries by using drones and classical vehicles. In A. Sforza, & C. Sterle (Eds.), *Optimization and decision science: methodologies and applications*: *vol. 217*, (pp. 557–565). Cham: Springer International Publishing, http://dx.doi.org/10.1007/978-3-319-67308-0_56.

Dohn, A., Kolind, E., & Clausen, J. (2009). The manpower allocation problem with time windows and job-teaming constraints: A branch-and-price approach. *Computers & Operations Research*, *36*(4), 1145–1157.

Drexl, M. (2012). Synchronization in vehicle routing—A survey of VRPs with multiple synchronization constraints. *Transportation Science*, *46*(3), 297–316.

Drexl, M. (2013). Branch-and-cut algorithms for the vehicle routing problem with trailers and transshipments. *Networks*, *63*(1), 119–133.

Es Yurek, E., & Ozmutlu, H. C. (2018). A decomposition-based iterative optimization algorithm for traveling salesman problem with drone. *Transportation Research Part C (Emerging Technologies)*, *91*, 249–262.

Euchi, J., & Sadok, A. (2021). Hybrid genetic-sweep algorithm to solve the vehicle routing problem with drones. *Physical Communication*, *44*, Article 101236.

Faturechi, R., & Miller-Hooks, E. (2014). Travel time resilience of roadway networks under disaster. *Transportation Research, Part B (Methodological)*, *70*, 47–64.

Fink, M., Desaulniers, G., Frey, M., Kiermaier, F., Kolisch, R., & Soumis, F. (2019). Column generation for vehicle routing problems with multiple synchronization constraints. *European Journal of Operational Research*, *272*(2), 699–711.

Gilmore, P. C., & Gomory, R. E. (1961). A linear programming approach to the cutting-stock problem. *Operations Research*, *9*(6), 849–859.

Gu, R., Poon, M., Luo, Z., Liu, Y., & Liu, Z. (2022). A hierarchical solution evaluation method and a hybrid algorithm for the vehicle routing problem with drones and multiple visits. *Transportation Research Part C (Emerging Technologies)*, *141*, Article 103733.

Irnich, S., & Desaulniers, G. (2005). *Shortest path problems with resource constraints*. Springer.

Jeong, H. Y., Yu, D. J., Min, B.-C., & Lee, S. (2020). The humanitarian flying warehouse. *Transportation Research Part E: Logistics and Transportation Review*, *136*, Article 101901.

Jiang, J., Dai, Y., Yang, F., & Ma, Z. (2024). A multi-visit flexible-docking vehicle routing problem with drones for simultaneous pickup and delivery services. *European Journal of Operational Research*, *312*(1), 125–137.

Karaköse, E. (2024). A new last mile delivery approach for the hybrid truck multi-drone problem using a genetic algorithm. *Applied Sciences*, *14*(2), 616.

Kitjacharoenchai, P., Min, B.-C., & Lee, S. (2020). Two echelon vehicle routing problem with drones in last mile delivery. *International Journal of Production Economics*, *225*, Article 107598.

Kitjacharoenchai, P., Ventresca, M., Moshref-Javadi, M., Lee, S., Tanchoco, J. M., & Brunese, P. A. (2019). Multiple traveling salesman problem with drones: Mathematical model and heuristic approach. *Computers & Industrial Engineering*, *129*, 14–30.

Kuo, R. J., Lu, S.-H., Lai, P.-Y., & Mara, S. T. W. (2022). Vehicle routing problem with drones considering time windows. *Expert Systems with Applications*, *191*, Article 116264.

Li, Y., Lim, A., & Rodrigues, B. (2005). Manpower allocation with time windows and job-teaming constraints. *Naval Research Logistics*, *52*(4), 302–311.

Liu, Y., Liu, Z., Shi, J., Wu, G., & Pedrycz, W. (2021). Two-echelon routing problem for parcel delivery by cooperated truck and drone. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, *51*(12), 7450–7465.

Long, Y., Xu, G., Zhao, J., Xie, B., & Fang, M. (2023). Dynamic truck–UAV collaboration and integrated route planning for resilient urban emergency response. *IEEE Transactions on Engineering Management*, 1–13.

Lübbecke, M. E. (2010). Column generation. In *Wiley encyclopedia of operations research and management science* (pp. 1–14). New York: Wiley.

Luo, Z., Poon, M., Zhang, Z., Liu, Z., & Lim, A. (2021). The multi-visit traveling salesman problem with multi-drones. *Transportation Research Part C (Emerging Technologies)*, *128*, Article 103172.

Luo, Z., Qin, H., Zhu, W., & Lim, A. (2016). Branch-and-price-and-cut for the manpower routing problem with synchronization constraints. *Naval Research Logistics*, *63*(2), 138–171.

Meisel, F., & Kopfer, H. (2012). Synchronized routing of active and passive means of transport. *OR Spectrum*, *36*(2), 297–322.

Moshref-Javadi, M., Lee, S., & Winkenbach, M. (2020). Design and evaluation of a multi-trip delivery model with truck and drones. *Transportation Research Part E: Logistics and Transportation Review*, *136*, Article 101887.

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

Murray, C. C., & Raj, R. (2020). The multiple flying sidekicks traveling salesman problem: Parcel delivery with multiple drones. *Transportation Research Part C (Emerging Technologies)*, *110*, 368–398.

Pessoa, A., Sadykov, R., Uchoa, E., & Vanderbeck, F. (2018). Automation and combination of linear-programming based stabilization techniques in column generation. *INFORMS Journal on Computing*, *30*(2), 339–360.

Poikonen, S., & Golden, B. (2020). Multi-visit drone routing problem. *Computers & Operations Research*, *113*, Article 104802.

Sacramento, D., Pisinger, D., & Ropke, S. (2019). An adaptive large neighborhood search metaheuristic for the vehicle routing problem with drones. *Transportation Research Part C (Emerging Technologies)*, *102*, 289–315.

Sah, B. (2019). *Drone truck combined operation: models and algorithm* (Ph.D. thesis).

Salama, M. R., & Srinivas, S. (2022). Collaborative truck multi-drone routing and scheduling problem: Package delivery with flexible launch and recovery sites. *Transportation Research Part E: Logistics and Transportation Review*, *164*, Article 102788.

Schermer, D., Moeini, M., & Wendt, O. (2019). A matheuristic for the vehicle routing problem with drones and its variants. *Transportation Research Part C (Emerging Technologies)*, *106*, 166–204.

Song, B. D., Park, H., & Park, K. (2022). Toward flexible and persistent UAV service: Multi-period and multi-objective system design with task assignment for disaster management. *Expert Systems with Applications*, *206*, Article 117855.

Tamke, F., & Buscher, U. (2021). A branch-and-cut algorithm for the vehicle routing problem with drones. *Transportation Research, Part B (Methodological)*, *144*, 174–203.

Wang, X., Poikonen, S., & Golden, B. (2017). The vehicle routing problem with drones: Several worst-case results. *Optimization Letters*, *11*(4), 679–697.

Wang, Z., & Sheu, J.-B. (2019). Vehicle routing problem with drones. *Transportation Research, Part B (Methodological)*, *122*, 350–364.

Wu, G., Mao, N., Luo, Q., Xu, B., Shi, J., & Suganthan, P. N. (2022). Collaborative truck-drone routing for contactless parcel delivery during the epidemic. *IEEE Transactions on Intelligent Transportation Systems*, *23*(12), 25077–25091.

Yáñez-Sandivari, L., Cortés, C. E., & Rey, P. A. (2021). Humanitarian logistics and emergencies management: New perspectives to a sociotechnical problem and its optimization approach management. *International Journal of Disaster Risk Reduction*, *52*, Article 101952.

Yu, D., Yin, J., Wilby, R. L., Lane, S. N., Aerts, J. C. J. H., Lin, N., et al. (2020). Disruption of emergency response to vulnerable populations during floods. *Nature Sustainability*, *3*(9), 728–736.

Zhong, M., Shi, C., Fu, T., He, L., & Shi, J. (2010). Study in performance analysis of China urban emergency response system based on Petri net. *Safety Science*, *48*(6), 755–762.