International Journal of Production Research

ISSN: 0020-7543 (Print) 1366-588X (Online) Journal homepage: www.tandfonline.com/journals/tprs20

Enhancing offshore parcel delivery efficiency through vessel-unmanned aerial vehicle collaborative routing

Yantong Li & Xingqi Wang

To cite this article: Yantong Li & Xingqi Wang (2025) Enhancing offshore parcel delivery efficiency through vessel-unmanned aerial vehicle collaborative routing, International Journal of Production Research, 63:10, 3485-3511, DOI: 10.1080/00207543.2024.2422818

To link to this article: https://doi.org/10.1080/00207543.2024.2422818

Published online: 06 Nov 2024.

INTERNATIONAL JOURNAL OF PRODUCTION RESEARCH
2025, VOL. 63, NO. 10, 3485–3511
https://doi.org/10.1080/00207543.2024.2422818

Enhancing offshore parcel delivery efficiency through vessel-unmanned aerial vehicle collaborative routing

Yantong Li and Xingqi Wang

School of Maritime Economics and Management, Dalian Maritime University, Dalian, People's Republic of China

ABSTRACT
Offshore oil and gas production is vital for global energy supply, but it faces logistical challenges due to the high costs and inefficiencies of traditional supply methods. This paper introduces the vessel-unmanned aerial vehicle (UAV) routing problem with multiple visits in a single flight (VURP-M), a novel logistical model that addresses aimed at enhancing the replenishment of offshore platforms with small, essential items. The VURP-M allows the UAV to perform multiple visits during a single flight, optimising the delivery process. To tackle the VURP-M, we propose two mixed-integer second-order cone programs that capture the problem's complexities. Given its NP-hard nature, we employ an adaptive large neighbourhood search (ALNS) method, featuring a segmented initialisation process and problem-specific operators guided by a rule-based mechanism to improve solution efficiency. The ALNS formulates an initial solution by solving a travelling salesman problem to create a giant tour, which is then used to group sequential targets into multiple UAV flights. The subsequent optimal resolution of a SOCP determines the take-off and landing points for each flight. Subsequently, the ALNS refines the initial solution through destroy and repair operators, enhancing the search for superior sequences and allocation schemes. The effectiveness of our approach is demonstrated through a real-world case study and numerical experiments on random instances featuring up to 100 platforms. The results offer implications of the collaborative vessel-UAV model for the offshore logistics industry.

ARTICLE HISTORY
Received 13 May 2024
Accepted 23 September 2024

KEYWORDS
Industry, innovation and infrastructure; offshore logistics; vessel-UAV collaboration routing; mixed-integer second-order cone programs; adaptive large neighbourhood search

1. Introduction

Offshore oil and gas production plays an important role in global energy supply, contributing to energy security and propelling the global economy (Camponogara and Plucenio 2014). According to the International Energy Agency (IEA), offshore oil and gas production accounts for approximately 30% of the world's total oil production, commanding a substantial market share (Agency 2023). The operation of offshore oil and gas platforms relies on the regular replenishment of materials, including small items such as spare parts, medical supplies, and documents (Company 2023). Despite their small size, these items are critical for maintaining platform operations and responding to emergencies. Traditional maritime transportation methods, such as supply vessels and helicopters, while capable of performing delivery tasks, suffer from high costs, low safety, wasteful capacity, and environmental pollution, making it difficult to meet the growing demand for small-item replenishment on offshore platforms.

The emergence of unmanned aerial vehicles (UAVs), commonly referred to as drones, has introduced novel avenues to address logistical challenges in offshore production practice (Erdoğan and Yıldırım 2021). These autonomous aerial systems present opportunities for swift and accurate delivery, particularly advantageous in the offshore oil and gas sector, where the timely transportation of small items holds utmost importance. Nonetheless, the current constraints within UAV technology, notably the limited range of battery-powered drones, hinder their independent deployment for offshore delivery endeavours. To address this limitation, a collaborative strategy has been proposed that leverages the capabilities of UAVs with the resilience of the supply vessels. Within this innovative 'vessel-UAV' collaborative model, the vessel assumes the role of a central hub for UAV operations, facilitating stable platforms for takeoff, landing, and recharging. Leveraging the support of the vessel, UAVs can extend their operational reach to execute precise delivery tasks to offshore platforms.

CONTACT Yantong Li yantong.li@dlmu.edu.cn, yantongli@163.com School of Maritime Economics and Management, Dalian Maritime University, Dalian 116026, People's Republic of China

This collaborative paradigm not only mitigates the endurance limitations of UAVs but also enhances the efficacy and adaptability of the supply chain, providing a cost-efficient and eco-friendly approach to offshore logistics. In addition, this collaborative vessel-UAV system offers efficient solutions to various scenarios such as vessel emission monitoring (Xia, Wang, and Wang 2019), inspection of offshore oil and gas production facilities (Tang and Li 2023.; Xue, Li, and Wang 2023), search and rescue, and management of maritime traffic and transportation (Poikonen and Golden 2020).

Despite the potential benefits of the vessel-UAV collaborative model, several challenges must be surmounted to ensure its effective integration into offshore logistics. One primary obstacle is the necessity for intricate route planning for both the vessel and the UAV, resulting in a complex problem known as the vessel-UAV routing problem (VURP). The VURP introduces novel features, notably the vessel's operation within continuous Euclidean space and the UAV's capability to take off and land on the vessel at any point along its trajectory. Firstly, the vessel's navigation within continuous Euclidean space adds significant complexity, as optimisation must be performed over an infinite set of potential routes. Secondly, the UAV's freedom to take off and land at various points along the vessel's path introduces unprecedented flexibility, albeit complicating the planning process by exponentially expanding the solution space. The optimal selection of takeoff and landing points for the UAV necessitates careful coordination with the vessel's route to ensure operational efficiency. Moreover, the VURP must account for operational constraints imposed by both the vessel and the UAV, including the UAV's range limitations, the vessel's speed, and the spatial distribution of offshore platforms. Integrating these constraints into the routing model demands a nuanced understanding of the interactions between the vessel, UAV, and broader logistical dynamics.

The VURP corresponds to the mothership-drone routing problem and the carrier-vehicle routing problem. While the VURP offers diverse applications, it has garnered limited attention within the academic literature. Recent studies have focussed on addressing a simplified version of the problem, wherein the UAV is constrained to visiting only one platform per flight (Erdoğan and Yıldırım 2021; Gambella, Lodi, and Vigo 2018; Poikonen and Golden 2020). However, with the rapid advancement in UAV mileage capabilities, there arises a pressing need for the exploration of more complex scenarios that better reflect real-world conditions. For instance, in offshore platform delivery operations, oil production fields typically comprise multiple closely situated platforms. In such scenarios, the ability of UAVs to access multiple targets within a single flight presents operational advantages. This necessitates the formulation of a new variant of the problem, termed the vessel-UAV routing problem with multiple visits in a single flight (VURP-M). Nevertheless, addressing the VURP-M, wherein UAVs can execute multiple visits during a single flight, poses substantial challenges. The extension to multiple visits complicates the VURP by necessitating routing decisions for each takeoff of the UAV. Addressing these challenges mandates the development of advanced mathematical models and algorithms capable of handling the intricacies of the VURP-M. Solutions to the VURP-M hold the key to unlocking the full potential of the vessel-UAV collaborative model in optimising offshore logistics operations.

This paper investigates an integrated vessel-UAV routing problem, denoted as VURP-M, prevalent in the operational framework of offshore logistics. The VURP-M extends the conventional vessel-UAV routing problem by permitting the UAV to visit multiple targets within a single flight. In such scenarios, the determination of optimal visiting sequences for targets assigned to the same UAV flight becomes imperative. We formalise this novel problem through the utilisation of two mixed-integer second-order cone programs (MISOCPs). Given the NP-hard complexity inherent in the VURP-M, we employ a tailored adaptive large neighbourhood search (ALNS) approach to address practical-sized instances. The ALNS method encompasses a segmented initialisation process and a repertoire of problem-specific destroy and repair operators. Notably, rather than selecting operators randomly in each iteration, we integrate a rule-based mechanism to apply appropriate operators, thereby expediting the convergence of the ALNS. To validate our approach, we apply it to tackle a real-world case derived from offshore oil and gas field delivery practices. Managerial insights are drawn from sensitivity analysis outcomes. Furthermore, numerical experiments conducted on random instances featuring up to 100 platforms demonstrate the effectiveness of the proposed models and solution methodology.

The rest of this paper is organised as follows. In Section 2, we present a literature review and summarise our main contributions. In Section 3, we provide a comprehensive description of the problem and propose two MISOCP formulations for the VURP-M. Section 4 presents the ALNS method incorporates a segmented initialisation process and utilises various problem-specific destroy and repair operators. Section 5 includes a real-world case with sensitivity analysis and numerical experiments on random instances. Section 6 contains the summary of the paper and the direction of future research.

2. Literature review

Our research focuses on the collaboration between vessels and UAVs, specifically exploring its relevance in offshore logistics applications. Additionally, we closely examine the coordination systems involving trucks and UAVs, commonly employed in onshore industries. We next conduct a brief review of existing literature in these areas and identify our contributions.

2.1. Offshore logistics

Offshore logistics refers to the application of maritime logistics within short-distance or regional scopes, including the transportation of goods along coasts, between islands, or among offshore facilities. Current research is focussed on long-term planning for offshore logistics and emphasises the development of robust scheduling to accommodate the dynamic and uncertain nature of operations within the offshore delivery industry. Studies such as the research on vessel scheduling with multiple time windows by Christiansen and Fagerholt (2002), the analysis of supply vessels' role in offshore logistics by Aas, Halskau Sr, and Wallace (2009), and the extension of the vessel routing problem for offshore oil and gas logistics by Cuesta et al. (2017), have all contributed to a deeper understanding of the complexities involved. The study conducted by Norlund, Gribkovskaia, and Laporte (2015) further explored the balance between cost, environmental impact, and robustness in supply vessel planning, particularly during challenging seasons.

Addressing uncertainty in the liquefied natural gas business, Andersson, Christiansen, and Desaulniers (2016) considered an inventory routing problem in the supply chain. Halvorsen-Weare, Fagerholt, and Rönnqvist (2013) proposed robustness strategies for vessel routing and scheduling, evaluated through simulation models. Halvorsen-Weare and Fagerholt (2017) discussed optimisation in vessel planning, introducing innovative models for fleet size and route determination. Hermeto, Ferreira Filho, and Bahiense (2014) focussed on logistics network planning for air transport of oil rig crews.

Building on this foundation, further research has delved into specific optimisation heuristics and models. Borthen et al. (2018) introduced a genetic search-based heuristic for fleet size and periodic routing, tailored to Statoil's supply vessel planning needs. Amiri, Amin, and Tavakkoli-Moghaddam (2019), focussing on a two-echelon node-based location routing problem in offshore oil and gas supply chains, employed a Lagrangian decomposition approach to optimise fleet composition and routing. Borthen et al. (2019) developed a bi-objective model to balance costs with persistence objectives in offshore supply vessel planning. Vieira et al. (2021) offered exact and heuristic algorithms for fleet composition and periodic routing with berth allocation decisions. Ulsrud et al. (2022) developed an adaptive large neighbourhood search heuristic specifically for the time-dependent vessel routing problem with speed optimisation. This heuristic optimises the sailing speeds of platform supply vessels under varying weather conditions, effectively reducing logistics costs in the offshore oil and gas industry while accounting for uncertainties in fuel consumption and travel time.

Collectively, these works underscore the importance of optimisation techniques in enhancing the efficiency and reliability of offshore logistics. We recommend interested readers refer to the review by Sirimanne et al. (2019) for a more comprehensive understanding of the offshore logistics problem. Despite extensive research on long-term planning for offshore logistics, there is limited research on temporary small items delivery in the offshore domain, and there is still a significant gap in collaborative delivery, especially in the area of vessel and UAV collaborative delivery. Next, we will summarise the existing literature separately from two perspectives: truck-UAV and vessel-UAV collaborations.

2.2. Vessel-UAV collaborative routing

The application of UAVs has posed a series of complex scheduling problems, such as UAV routing problem (Cheng, Adulyasak, and Rousseau 2020; Otto et al. 2018; Pei et al. 2024), UAV assignment and scheduling problems (Glock and Meyer 2020; Peters and Bertuccelli 2016; Zou et al. 2023), and truck-UAV collaborative routing problem (TURP) (Cavani, Iori, and Roberti 2021; D. Wang et al. 2019; Kitjacharoenchai et al. 2019; Liang and Luo 2022; Madani and Ndiaye 2022; Murray and Chu 2015; Schermer, Moeini, and Wendt 2020; X. Wang, Poikonen, and Golden 2017). In TURP, the UAV is assumed to take off and land at some specific customer and the truck travels on a given road network. However, these assumptions limit the applicability of the solutions developed for the truck-UAV system to the vessel-UAV system with distinct features. First, the VURP involves carrier vessels and UAVs operating without the limitations of fixed road networks. Within a continuous space, both vessels and UAVs possess unrestricted navigational flexibility. Moreover, the UAV take-off and landing points are no longer limited to some specific nodes, but along the vessel path. Despite the significance of vessel-UAV systems, the literature on the vessel-UAV routing problem, also known as the mothership-drone routing problem and carrier-vehicle routing problem, is relatively scarce.

There are two key decisions in the vessel-UAV routing problem, the visiting sequences of platforms and the take-off and landing points of the UAV upon visiting each platform. For a reduced case of the VURP where the visiting sequences of platforms are fixed, Garone et al. (2010) optimised the take-off and landing points of a single UAV carried by a single vessel. The problem was modelled as a SOCP. Subsequently, Garone, Naldi, and Casavola (2011) enumerated the exact solution of the problem for instances with up to five platforms. Gambella, Lodi, and Vigo (2018) formulated VURP as a MISOCP, which is an extension of the carrier-vehicle problem (CVP) model introduced by Garone et al. (2010). This formulation took into account the combined decisions involving assignment, sequence, and coordinates. Then, they proposed a ranking-based algorithm based on an exact enumeration procedure, decomposing the primary problem into a master problem (MP) and a subproblem (SP), and achieved the optimal solution iteratively by solving the MP and SP. Experimental results demonstrated that this method optimally solves instances with up to 15 platforms. Poikonen and Golden (2020) established a SOCP model in a fixed order and devised a precise branch-and-bound (B&B) algorithm and a greedy heuristic with local search improvements. The B&B algorithm can obtain optimal solutions for instances with up to ten platforms. They extended the problem to consider multiple visits upon a flight for a special case assuming infinite mileage of UAV and fixed visiting sequences of platforms. They proposed an enumerative method to find all possible allocations of targets to flights. Erdoğan and Yıldırım (2021) reformulated the VURP as a new MISOCP model, where the model is strengthened by several optimality cuts derived from the structural properties of the problem. Their results demonstrated the superiority of the proposed model over the one presented in Gambella, Lodi, and Vigo (2018). Additionally, they developed an iterated local search (ILS) heuristic and compared it with the B&B algorithm introduced in Poikonen and Golden (2020), validating the effectiveness of their algorithm.

Recently, some scholars have extended VURP to consider more complex cases, where the platforms can be polygonal chains or a graph, and the UAV must visit the fractional arcs of the graph. Amorosi, Puerto, and Valverde (2021, 2022) extended the VURP to include a more complex situation where a UAV can visit polygonal chains. They studied the case of a single carrier carrying a UAV, and the optimisation goal was to minimise the weighted travelling distance between the vessel and the UAV. Subsequently, the author designed a heuristic by first fixing the access order of targets and then setting the values of the binary variables and solving the resulting model to obtain a feasible solution. They handled instances with up to 15 target polygonal chains. Later on, Amorosi, Puerto, and Valverde (2023) extended the problem to consider multiple UAVs. The target was represented by a graph and a section of a given length must be traversed. The primary objective was to minimise maximum completion time upon finishing all tasks. Tang and Li (2023) extended the model of Erdoğan and Yıldırım (2021) to form a new multi-period variant, wherein decision makers were required to further determine which set of target points to visit in each time period. Y. Li et al. (2024) extended the problem to a multiple vessel variant, where several vessels must inspect all offshore platforms within a certain period and return to the initial port, with the goal of minimising the total operational cost. For the interesting application of the vessel-UAV collaborative system, Xue, Li, and Wang (2023) studied the vessel-UAV routing problem arising from the offshore inspection practice. However, they force the vessel to directly visit each platform, where the loaded UAVs then perform inspection tasks by visiting a set of arcs and returning to the vessel at the same platform. In their study, the routes of the UAVs are first determined, and the vessels's routes are then determined based on the UAVs' routing results.

The aforementioned literature primarily focuses on single-vessel and single-UAV systems, with the constraint that a UAV can visit only one platform per flight. There is limited consideration for scenarios where a UAV, within its mileage, can visit multiple platforms in a single flight. We note that three studies by Poikonen and Golden (2020), Amorosi, Puerto, and Valverde (2021), and Y. Li et al. (2023) have innovatively considered the case of multiple visits in a single flight. However, Poikonen and Golden (2020) fixed the visiting sequence before establishing the model and assumed that the mileage of the UAV was infinite. Amorosi, Puerto, and Valverde (2021) lean more towards the complexity of accessing polygonal chains and addressing only the case with 15 targets. The work presented by Y. Li et al. (2023) serves as a foundational precursor to our study. They proposed a new vessel-UAV routing problem with multiple visits upon each flight (VURP-M), wherein they have specifically formulated two MISOCP models for VURP-M, and CPLEX is used to solve the model. The explored problem extends the traditional vessel-UAV routing problem by permitting the UAV to visit multiple targets within each flight. Besides, they consider a more realistic objective that minimises the total routing cost of both the vessel and UAV. However, the model can only address instances with up to 15 targets when directly solved by CPLEX. Our paper extends Y. Li et al. (2023) in the following aspects. First, we propose several valid inequalities to further enhance the model. Second, we design a ALNS method to handle large-scale practical instances. In addition, current researches on the VURP assume that the vessel travels continuously at a constant speed. However, in some cases, it would be beneficial if the vessel could wait for the UAV to facilitate better coordination. We next highlight our contributions.

2.3. Contributions

Our paper contributes to the literature and practice in the following aspects.

- We have extended the model proposed by Y. Li et al. (2023), introducing effective inequalities to improve the model's computational efficiency. Furthermore, we have taken into account a more realistic scenario where, during UAV service, the vessel can either continue its voyage or wait in the same location for the UAV to be recovered.
- We develop a customised ALNS algorithm, using efficient destroy and repair operators by exploiting the problem structure. The ALNS solves a TSP to obtain a giant tour, and then the targets are sequentially allocated to UAV flights. Subsequently, with the visiting sequence and target allocation variables fixed, we solve the SOCP using the proposed model to optimally determine the UAV's take-off and landing points. We design several problem-specific destroy and repair operators to search for better sequences and allocation schemes.
- The proposed ALNS is applied to solve a real-world case arising from offshore logistics, demonstrating its applicability in realistic scenarios. To assess the efficacy of both the proposed model and algorithm, we undertake numerical experiments using randomly generated instances involving up to 100 targets.

3. Problem description and model formulation

Offshore oil and gas production practices entail the management and maintenance of numerous platforms distributed across expansive marine environments. These platforms serve as critical hubs for extraction, processing, and transportation activities essential to the petroleum industry. Within this operational context, the efficient replenishment of materials and supplies to offshore platforms is paramount for sustaining production and ensuring safety.

The vessel-UAV routing problem with multiple visits in a single flight (VURP-M) arises as a direct consequence of the logistical challenges inherent in offshore oil and gas production. In this scenario, offshore platforms, often situated in close proximity to one another within a production field, require regular deliveries of supplies, spare parts, and equipment.

In this section, we provide a formal description of the VURP-M and outline the mathematical model formulated to address this complex optimisation problem. We present two MISOCPs designed to capture the intricate interactions between vessels, UAVs, and offshore platforms in the context of offshore logistics operations.

3.1. Problem description

The vessel-UAV collaborative routing problem with multiple visits (VURP-M) consists of a designated port 0, a vessel, and a UAV. The vessel departs from the port to service a set $N = \{1, \ldots, n\}$ of platforms. The port is located at p_0, and the position of platforms i is denoted as p_i. The vessel is ready to depart from the port at the beginning of the planning horizon. Both the vessel and the UAV have unrestricted movement in the two-dimensional Euclidean plane $\mathbb{R}^2$. The vessel carries exactly one UAV with a mileage of R and serves as the mothership of the UAV. The vessel and UAV work cooperatively as follows. The vessel departs from the port and navigates toward a platform. The loaded UAV is released from the vessel when the vessel is close enough to the target. The UAV then flies to the corresponding platform at a fixed speed v_d, performs delivery tasks, and then either flies to the next platform or returns to the vessel. The set $K = \{1, \ldots, m\}$ of flights can be performed by the UAV, where $m \leq n$. We assume that the vessel maintains a fixed speed v_d between any arcs composed of the port, takeoff points, and landing points. However, the vessel can wait at the takeoff point of the UAV when needed. Upon finishing all tasks, the vessel must return to the port together with the UAV.

The vessel and UAV have travel costs C_v and C_d per unit distance respectively. The objective is to optimise the total costs, including the variable vessel routing cost and UAV routing cost. The decisions to be made include (i) the route of the vessel; (ii) the route of the UAV in each flight, including tasks assignment and visiting sequence; and (iii) take-off and landing points of the UAV. To facilitate the problem, we make the following assumptions:

- The system consists of one vessel and one UAV;
- Both the vessel and the UAV can move freely in the Euclidean plane;
- The vessel ends up its route at the same port where it departs;
- The UAV can perform multiple delivery tasks per flight;
- The vessel can depart only once from the port;
- The battery swapping time for the UAV is negligible.

3.2. Model formulation

Based on the problem description and assumptions, we can define the following sets, parameters, and decision variables:

Sets:

N	Set of platforms, $N = \{1, 2, \ldots, n\}$, indexed by i and j;
K	Set of UAV flights, $K = \{1, 2, \ldots, m\}$, indexed by k;
Z	Set of integer.

Parameters:

R	UAV mileage (hours);
A_0	Starting (initial port) coordinate of the vessel;
A_{n+1}	Destination (destination port) coordinate of the vessel, where $p_{n+1} = p_0$;
p_i	The coordinate of the platform i;
C_v	Vessel travel cost per unit distance;
C_d	UAV travel cost per unit distance;
v_v	Vessel travel speed;
v_d	UAV travel speed;
c_{ij}	The distance between i and j, where $c_{ij} = \|p_i - p_j\|$;
M	A large number.

Decision variables:

w_k	The take-off coordinate for the k_{th} flight of the UAV;
q_k	The landing coordinate for the k_{th} flight of the UAV;
t_k^p	The take-off moment at the k_{th} flight of the UAV;
t_k^q	The landing moment at the k_{th} flight of the UAV;
t_k^r	The travel time of the vessel during the k_{th} UAV flight;
t^f	The moment the vessel returned to the port;
t_i	The moment when the UAV arrives at platform i;
v_{ik}	Equal to 1 if platform i is visited by the UAV during the k_{th} flight, 0 otherwise;
x_{ijk}	Equal to 1 if the UAV travels from i to j during the k_{th} flight, 0 otherwise.

Using the symbols defined above, we formulate a MISOCP, denoted as M1, as follows:

$$Z = \min \left\{ C_v v_v (t_1^p + t^f - t_m^q) + \sum_{k \in K \setminus \{m\}} C_v v_v (t_{k+1}^p - t_k^q) + \sum_{k \in K} C_v v_v t_k^r + \sum_{k \in K} C_d v_d (t_k^q - t_k^p) \right\} \tag{1}$$

s.t.

$$\| A_0 - p_1 \| \le v_v t_1^p \tag{2}$$

$$\| A_{n+1} - q_m \| \le v_v (t^f - t_m^q) \tag{3}$$

$$\| w_{k+1} - q_k \| \le v_v (t_{k+1}^p - t_k^q) \quad \forall\, k \in K \setminus \{m\} \tag{4}$$

$$\| w_k - q_k \| \le v_v t_k^r \quad \forall\, k \in K \tag{5}$$

$$\| p_i - w_k \| \le v_d (t_i - t_k^p) + M(1 - x_{0ik}) \quad \forall\, i \in N,\ k \in K \tag{6}$$

$$\| q_k - p_i \| \le v_d (t_k^q - t_i) + M(1 - x_{i,n+1,k}) \quad \forall, i \in N,\ k \in K \tag{7}$$

$$v_d (t_j - t_i) \ge c_{ij} - \sum_{k=1}^{K} M(1 - x_{ijk}) \quad \forall\, i, j \in N,\ k \in K \tag{8}$$

$$t_k^q \ge t_k^r + t_k^p \quad \forall\, k \in K \tag{9}$$

$$\sum_{k=1}^{K} v_{ik} = 1 \quad \forall\, i \in N \tag{10}$$

$$\sum_{i \in N} x_{0ik} \le 1 \quad \forall\, k \in K \tag{11}$$

$$\sum_{j \in N_0 \setminus \{0\}} x_{ijk} = v_{ik} \quad \forall\, i \in N,\ k \in K \tag{12}$$

$$\sum_{j \in N_0 \setminus \{n+1\}} x_{jik} = v_{ik} \quad \forall\, i \in N,\ k \in K \tag{13}$$

$$\sum_{i=1}^{n} x_{0,i,k+1} \le \sum_{i=1}^{n} x_{0ik} \quad \forall\, k \in K \tag{14}$$

$$u_i - u_j + n x_{ijk} \le n - 1 \quad \forall\, i, j \in \left\{ N \mid i \ne j \right\},\ k \in K \tag{15}$$

$$1 \le u_i, u_j \le n \quad \forall\, i, j \in N,\ i \ne j, u_i,\ u_j \in Z \tag{16}$$

$$t_k^q - t_k^p \le R \quad \forall\, k \in K \tag{17}$$

$$t_k^p, t_k^q, t_k^r, t^f, t_i \ge 0 \quad \forall\, i \in N,\ k \in K \tag{18}$$

$$v_{ik}, x_{ijk} \in \{0, 1\} \quad \forall\, i, j \in N,\ k \in K \tag{19}$$

The objective function (1) minimises the total routing cost, constituting the vessel routing cost and the UAV routing cost. Constraint (2) represents the travel distance of the vessel from the port to the first take-off point of the UAV. Constraint (3) represents the distance of the vessel from the last landing point to the port. Constraints (4) represent the distance between the two consecutive take-off and landing points of the UAV. Constraints (5) calculate the sailing distance of the vessel during the k_{th} flight of the UAV. Constraints (6) represent the distance from the take-off point to the first platform of the k_{th} UAV flight, where the value of M is equal to the sum of UAV maximum travel distance v_dR. Constraints (7) are the distance between the landing point and the last visited platform of the k_{th} UAV flight, where the value of M is also equal to the sum of UAV maximum travel distance v_dR. Constraints (8) represent the distance constraint between any two consecutive platforms visited by the UAV, where we define M as the maximum distance between any two platforms. Constraints (9) indicate that the arrival time of the vessel at the landing point of any trip is earlier than the arrival time of the UAV. With these limitations, our model allows the vessel to wait for the UAV at a specific take-off point. Constraints (10) mean that each platform must be visited by a UAV. Constraints (11) indicate that there is at most one first visited platform upon each flight. Constraints (12) and (13) represent the flow balance constraints for each flight. Constraints (14) represent the symmetry breaking constraints, indicating that the $(k+1)_{th}$ flight cannot be used if the k_{th} flight is not used. Constraints (15) and (16) are the sub-tour-elimination constraints. Constraints (17) are UAV mileage constraints, indicating the flight duration cannot exceed the mileage R. Constraints (18)–(19) define the range of variables.

The above model, which corresponds to a MISOCP model, relies on the moment the UAV or vessel arrives at the take-off, target, and landing points. Our preliminary experiments on small-scale instances indicate that the commercial solver is inefficient in solving instances with fifteen platforms. We next present an alternative model, called M2, by defining the following additional variables.

Decision variables:

t_k^v	The travel time of the vessel during the k_{th} UAV flight;
t_k^d	The travel time of the UAV during the k_{th} flight;
$t_{k,k+1}$	The travel time of the vessel from the landing point of the k_{th} flight to the take-off point of the $k+1_{th}$ flight;
T^s	The travel time of the vessel from the port to the take-off of the first flight;
T^f	The travel time of the vessel from the landing point of the last flight to the port;

Based on the above newly defined variables, we formulate the second MISOCP, denoted as M2, as follows.

$$Z = \min \left\{ C_v v_v (T^s + T^f) + \sum_{k \in K \setminus \{m\}} C_v v_v t_{k,k+1} + \sum_{k \in K} C_v v_v t_k^v + \sum_{k \in K} C_d v_d t_k^d \right\} \tag{20}$$

s.t. (10)–(16), and to

$$\| A_0 - p_1 \| \leq v_v T^s \tag{21}$$

$$\| A_{n+1} - q_m \| \leq v_v T^f \tag{22}$$

$$\| w_k - q_k \| \leq v_v t_k^v \quad \forall\, k \in K \tag{23}$$

$$\| w_{k+1} - q_k \| \leq v_v t_{k,k+1} \quad \forall\, k \in K \setminus \{m\} \tag{24}$$

$$\| w_k - p_i \| + \| q_k - p_j \| + \sum_{i=1}^{N} \sum_{j=1}^{N} x_{ijk} c_{ij} \leq v_d t_k^d + M(2 - x_{0ik} - x_{j,n+1,k}) \quad \forall\, i,j \in N,\ k \in K \tag{25}$$

$$t_k^d \leq R \quad \forall\, k \in K \tag{26}$$

$$t_k^v \leq t_k^d \quad \forall\, k \in K \tag{27}$$

$$t_k^v, t_k^d, t_{k,k+1}, T^s, T^f \geq 0 \quad \forall\, k \in K \tag{28}$$

$$v_{ik}, x_{ijk} \in \{0,1\} \quad \forall\, i,j \in N,\ k \in K \tag{29}$$

The objective function (20) minimises the total routing cost. Constraint (21) is the travel distance of the vessel from the port to the take-off point of the first flight. Constraint (22) refers to the distance of the vessel from the landing point of the last flight to the port. Constraints (23) indicate the travel distance of the vessel during the k_{th} flight of the UAV. Constraints (24) represent the distance between the landing point of the k_{th} flight and the take-off point of the $k+1_{th}$ flight. Constraints (25) calculate the travel distance of the UAV upon each flight, where the constant M is set to $2v_dR + \sum_{i=1}^{N-1} \sum_{j=1}^{N} c_{ij}$. Constraints (26) indicate that the travel time of the UAV during each flight cannot exceed the mileage. Constraints (27) indicate that the vessel can wait in place for the UAV to return. Constraints (28) and (29) define the range of each decision variable.

Models M1 and M2 share some similarities, and they can both exactly express the studied problem. However, they use different modelling scheme where the definition of decision variables and the expression of constraints are different. These two models are initially presented by Y. Li et al. (2023), wherein they ignore the waiting time of vessels at the landing point of the UAV. We validate the two models using a state-of-the-art solver CPLEX. Preliminary results show that both models achieve the optimal solutions for instances with up to five platforms a short time. However, they tend to fail in solving instances with more than ten platforms. Although the two models provide exact solutions to the studied VURP-M, preliminary resuts show that these models have difficulty in solving practical-sized instances within acceptable computation time.

The studied VURP-M is an extension of the travelling salesman problem (TSP). When we simplify the problem by allowing the UAV to visit only one platform per flight, the problem is reduced to the carrier vehicle routing problem (CVTSP) proposed by Gambella, Lodi, and Vigo (2018). Further simplification of the problem leads to a TSP where a vessel departs from the port, visits all platforms, and then returns to the port, and the TSP problem is well-known to be NP-hard. Therefore, the VURP-M is NP-hard.

3.3. Valid inequalities

The proposed models M1 and M2 can be improved by developing valid inequalities. We next present valid inequalities to strengthen the two models. In the study by Erdoğan and Yıldırım (2021), they conducted an analysis of certain properties. The most significant property is that the maximum distance between the takeoff and landing point and the target point is $R(v_d + v_v)/2$. Based on this property, we introduce several valid inequalities to strengthen models M1 and M2. consisting of (1)–(29) to form a strengthening model called VURP-M+. We proposed the following constraints:

$$v_{ik} + v_{jk} \leq 1 \quad \forall\, i,j \in N,\ c_{ij} \geq R(v_d + v_v)/2,\ k \in K \tag{30}$$

$$x_{ijk} = 0 \quad \forall\, i,j \in N,\ c_{ij} \geq R(v_d + v_v)/2,\ k \in K \tag{31}$$

$$\sum_{i \in L_j} x_{ijk} = v_{jk} \quad \forall\, i \in N,\ k \in K \tag{32}$$

$$\sum_{i \in L_j} x_{jik} = v_{jk} \quad \forall\, i \in N,\ k \in K \tag{33}$$

Constraints (30)–(33) redefines the range of values for the integer variables v_{ik} and x_{ijk} in terms of i and j in the model, thereby reducing the search space for CPLEX to solve the model. Especially in constraints (32) and (33), the range of values for i is L_i, where $L_i = \{i \in N, c_{ij} \leq R(v_d + v_v)/2\}$. This means that only when i and j meet the above condition before they can be assigned to be the same trip. The computational results are shown in Table A1 in Appendix 1. The results indicate that the added valid inequalities can improve the quality of the solution and enhance the efficiency of the solution process. Nevertheless, with the expansion of the scale, the model encounters difficulties in solution finding, necessitating the development of heuristic approaches to address real-world scenarios.

4. Solution method

Due to the NP-hardness of the studied VURP-M, commercial solvers such as CPLEX have difficulty in solving large-sized instances. To this end, it is essential to develop some efficient solution algorithms to handle practical-sized instances. The VURP-M consists of optimally determining the platform allocation to flights, the sequence of the vessel to perform each flight, and the visiting sequence of platforms within each flight. Once the above decisions are fixed, the problem is reduced to a SOCP, by solving which we can obtain the take-off and landing coordinates of each flight. The VURP is a generalisation of the VRP, which motivates us to apply the adaptive large neighbourhood search algorithm, given its strength in solving large-scale combinatorial optimisation problems. We propose the ALNS algorithm structured into two steps. In the first step, we generate initial solutions, and in the second step, we destroy and repair these solutions. These steps incorporate two innovations: Firstly, our initial solution generation uses a fixed order inspired by the Lin-Kernighan (LKH) algorithm (Lin and Kernighan 1973) and integrates clustering principles to fix the number of platforms visited by UAVs on each trip. Secondly, leveraging the characteristic that UAVs need to visit multiple platforms per takeoff, we design four destroy and repair operators tailored for both the visitation sequence and the trips of UAVs. We next briefly introduce the framework of the ALNS algorithm in Section 4.1. We then detail the initial solution construction procedure in Section 4.2. On this basis, the destroy and repair operators used to explore neighbourhood solutions are elaborated in Section 4.3, and the adaptive mechanism is introduced in Section 4.4.

4.1. Framework of the tailored ALNS method

The ALNS algorithm is initially introduced by Ropke and Pisinger (2006) as an advanced version of the large neighbourhood search algorithm. It employs a variety of destroy and repair operators, which are selectively applied based on their efficacy observed throughout the search process, to improve the current solution. The destroy operators dismantle sections of the current solution, enabling the exploration of new solution spaces. These operators often incorporate a degree of randomness, essential for targeting diverse parts of the solution and thereby enhancing the search's diversity. Concurrently, the dismantled solutions are reconstructed by the repair operators, which are also stochastic to prevent repetitive construction of identical solutions when similar partial solutions are repeatedly encountered. A key aspect of this metaheuristic is the extent of destruction it employs. A minimal level of destruction might hinder the algorithm's ability to break free from local optima. Conversely, excessive destruction can challenge the repair mechanisms in reassembling viable solutions. To address this, the ALNS introduces an adaptive mechanism that strikes a balance among the impact of various operations during the search process, facilitating an automatic selection of the most promising destroy and repair operators for each iteration, based on their potential to yield superior solutions. Furthermore, the ALNS allows for the tailoring of the destroy and repair operators to suit the specific demands of the problem at hand. This customisation of the neighbourhood structure is pivotal in aligning the algorithm's approach with the unique characteristics of the problem being solved.

Algorithm 1 ALNS framework

```
1:  Initialization Phase:
2:  Generate an initial solution $S_0$, using the three-step method described in Algorithm 2 (Section 4.2).
3:  Set the best solution $S^* \leftarrow S_0$, and the current solution $S \leftarrow S_0$.
4:  Initialize scores, usage counts, and weights for each destroy and repair operator.
5:  Improvement Phase:
6:  for $i = 1$ to $T$ do
7:      Select a pair of destroy and repair operator $\lambda$ described in Section 4.3 using the roulette wheel selection rule.
8:      Generate a new solution $S'$ by executing operators $\lambda$ on $S$.
9:      if $f(S') \leq f(S^*)$ then
10:         $S^* \leftarrow S', S \leftarrow S'$.
11:     end if
12:     Update scores and usage counts for each operator $\lambda$.
13:     Update weights according to the scores and usage counts for each operator $\lambda$.
14: end for
15: output $S^*$.
```

The ALNS algorithm generally consists of two stages, where the first stage generates an initial solution, and the second stage tends to improve the obtained initial solution using destroy and repair operators. We make two improvements when applying the ALNS framework, including a piece-wise initial solution generation process, and new destroy and repair operators. In particular, we generate the initial solution S_0 using a three-step method. First, we solve a TSP considering the set of platforms using the state-of-the-art LKH algorithm (Lin and Kernighan 1973). Then, according to the obtained results of the TSP, we fix the visiting sequence of platforms and apply a clustering procedure to group platforms into flights of the UAV. Finally, given the visiting sequences of platforms and the corresponding allocation to flights, we solve a SOCP to determine the take-off and landing points of the UAV. We then obtain a feasible solution S_0 using the above three-step method. The detailed initialisation procedure is elaborated in Section 4.2. Subsequently, in the improvement phase, we design four problem-specific destroy and repair operators based on problem characteristics. The detailed description of four operators is available in Section 4.3. A roulette strategy is used to make an adaptive selection of operators in each iteration. We then use the selected operator λ to destroy and repair the current solution S, generate a new solution S', compare S' with the best solution S^*, and update the weight adaptively according to the comparison result. In particular, in order to avoid falling into the local optimum in the search process, the neighbourhood weight of the generated inferior solution is also updated. The adaptive mechanism and the way of weight updating are all proposed in Section 4.4. Finally, we output the best solution S^* until the maximum number of iterations T is reached. Algorithm 1 shows the framework of the tailored ALNS method. The ALNS algorithm flowchart is shown in Figure 1.

4.2. *First stage: initialisation*

The initialisation phase aims to generate a high-quality initial solution of the VURP-M using a three-step method.

Algorithm 2 Three-step method for initial solution generation

```
1:  Step 1: generate initial visiting sequence
2:  Solve a TSP for the set $N$ of targets using the LKH algorithm, and obtain an ordered set $N' = \{j_1, j_2, \ldots, j_n\}$.
3:  Step 2: allocate targets to UAV flights through clustering
4:  Initialize flight counter $k = 0$, set $F_1 = j_1, F_k \leftarrow \emptyset, \forall\ k = 2, 3, \ldots n$, and travel time $T_1 = t_{0j_1}, T_k = 0, \forall\ k = 2, 3, \ldots n$.
5:  for $i = 1$ to $n$ do
6:      if $T_k + t_{j_i j_{i+1}} < R$ then
7:          Set $F_k \leftarrow F_k \cup \{j_{i+1}\}$, $T_k \leftarrow T_k + t_{j_i j_{i+1}}$.
8:      else
9:          Set $k = k + 1$.
10:         Set $F_k \leftarrow F_k \cup \{j_{i+1}\}$, $T_k \leftarrow T_k + t_{j_i j_{i+1}}$.
11:     end if
12: end for
13: Step 3: get the take-off and landing coordinates of each flight $\forall\ k = 1, 2, \ldots n$ and $F_k \neq \emptyset$.
14: Solve a SOCP with $N'$ and $F_k$ as input to the model M2.
15: Output the initial solution $S_0$.
```

Step 1 determines the visiting sequence of the set N of platforms by solving a TSP using the LKH algorithm, which is generally considered to be one of the most effective methods for generating optimal or near-optimal solutions for the TSP. We then obtain a set N' of ordered platforms based on the results for solving the TSP. We then obtain a set $N' = \{j_1, j_2, \ldots, j_n\}$ of ordered platforms based on the results for solving the TSP.

Step 2 allocates platforms in the ordered set N' to the flights of the UAV. This allocation approach is inspired by Shen et al. (2021). We first initialise the flight counter as $k = 0$. Set the initial flight F_1 to include the first platform j_1, and for all subsequent flights F_k where k ranges from 2 to n, set F_k to an empty set. Additionally, we initialise travel time as $T_1 = t_{0j_1}$ for the first flight and $T_k = 0$ for all subsequent flights where k ranges from 2 to n. For each platform j_i in the ordered set N', where i

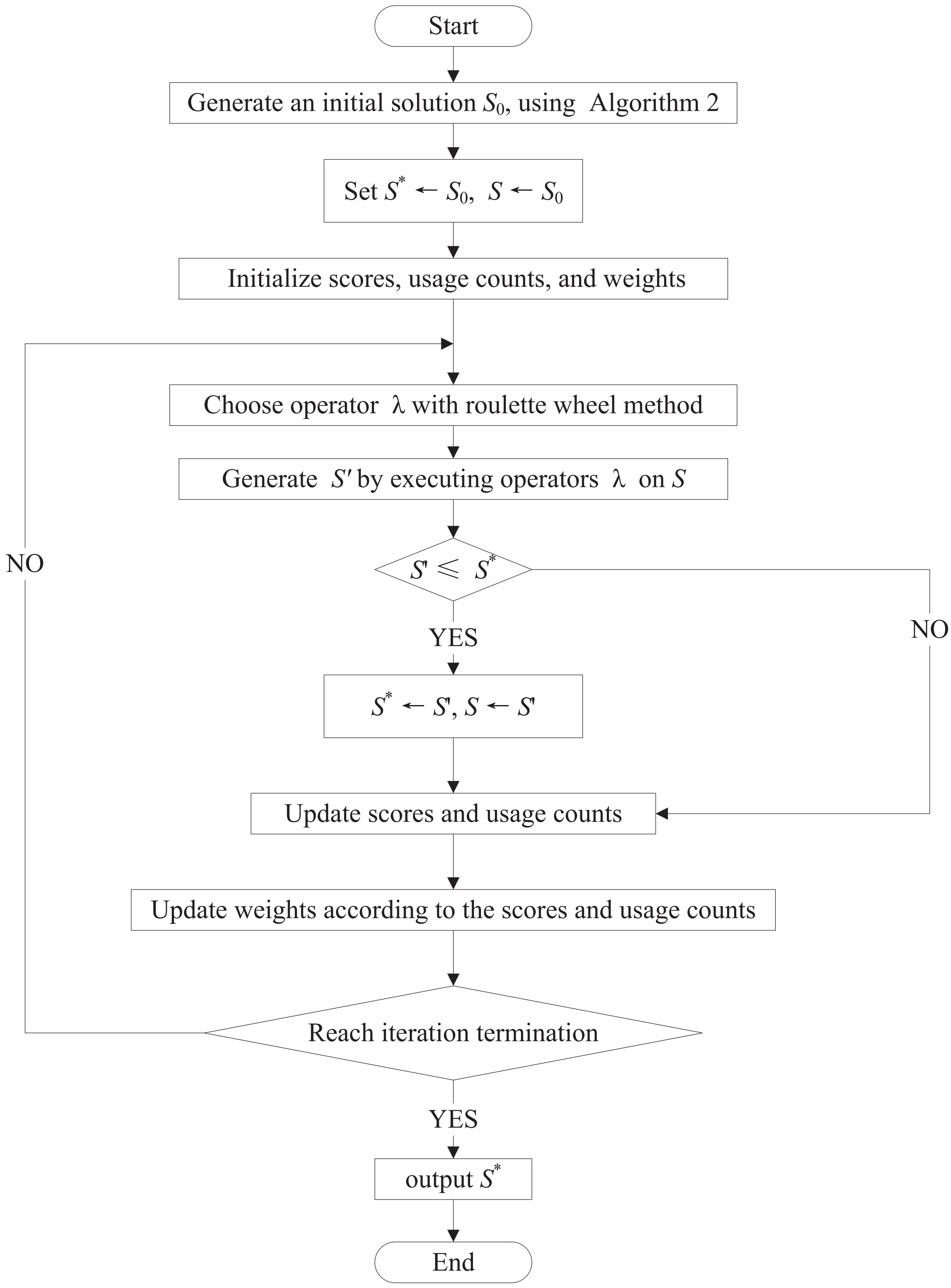

Start
Generate an initial solution S_0, using Algorithm 2
Set $S^* \leftarrow S_0$, $S \leftarrow S_0$
Initialize scores, usage counts, and weights
Choose operator λ with roulette wheel method
Generate S' by executing operators λ on S
$S' \leqslant S^*$
YES
NO
$S^* \leftarrow S'$, $S \leftarrow S'$
Update scores and usage counts
Update weights according to the scores and usage counts
Reach iteration termination
NO
YES
output S^*
End

Figure 1. Flow chart of ALNS algorithm.

ranges from 1 to n, the following process is implemented: If the sum of the current flight's travel time T_k and the travel time $t_{j_ij_{i+1}}$ to the successive platform j_{i+1} is less than R (the mileage of the UAV), then add j_{i+1} to the current flight F_k and update the travel time as $T_k \leftarrow T_k + t_{j_ij_{i+1}}$. Otherwise, increment the flight counter by setting $k = k + 1$. Then, set the current flight F_k to include j_{i+1} and update the travel time as $T_k \leftarrow T_k + t_{j_ij_{i+1}}$. This process continues until all platforms are considered, ensuring that they are appropriately allocated to flights based on the defined mileage constraint. Figure 2 provides an illustrative example of the clustering procedure. Given four platforms A, B, C, and D, whose visiting sequence is determined in the first step where a TSP is solved. Then in the second step, we iteratively check, from the first platform A, if $t_{AB} < R$ holds, then platforms A and B can be allocated to the first flight $k = 1$. Then we continue to check the second platform B, if $t_{AB} + t_{BC} < R$ holds, then platform C can be allocated to the first flight k, as shown in Figure 2(a). Then we continue to check the next pint D, if $t_{AB} + t_{BC} + t_{CD} < R$ holds, then platform D can be allocated to flight $k = 1$, as shown in Figure 2(b). Otherwise, platform D must be allocated to a new flight $k = 2$, as shown in Figure 2(c).

Step 3 solves a SOCP given the visiting sequence and target allocation decisions obtained in the previous two steps. In detail, we know the value of visiting sequence variables x_{ijk}, and the allocation variables v_{jk}, indicating that all integer variables are known in the established mathematical model. The remaining variables to be determined are the take-off and landing points of each flight. In this case, we can solve optimally the proposed MISOCP model by fixing the values of x_{ijk} and v_{jk}, which reduces to a SOCP. The SOCP can be quickly solved to optimality using commercial solvers like CPLEX.

Combining the above three-step procedures, we can obtain an initial solution S_0. Note that the visiting sequence and allocation variables are heuristically determined, which may be not optimal. Next, aiming at improving the initial solution, we propose four destroy and repair operators to explore better sequences and allocation schemes. The flowchart of the three-step method for initial solution generation is shown in Figure 3.

4.3. Second stage: apply to destroy and repair operators to improve the initial solution

In the second phase of the developed ALNS algorithm, the primary objective is to augment the initial solution S_0 through the systematic implementation of various destructive and constructive operators. As previously outlined, the initial solution is formed utilising the LKH algorithm to establish a predetermined visiting sequence. Subsequently, a clustering procedure is applied to allocate the platforms to the UAVs. Within this framework, we introduce a suite of operators aimed at improving two critical aspects: the sequential arrangement of target visits and the assignment of targets to UAVs.

To refine the order of target visits, we introduce perturbations into the initial solution by employing techniques such as platform removal, flight removal, and other pertinent strategies. Concurrently, concerning the assignment of targets to UAV flights, we utilise the groupings obtained during the initial solution generation process. Specifically, we adopt a dual approach wherein a selected group is randomly chosen, and the destructive process is initiated within that group. This synergistic approach, addressing both the order and UAV flight assignments, ensures a holistic enhancement strategy for the initial solution, with the aim of leveraging the inherent characteristics of the problem and iteratively refining the solution within the ALNS framework. The detailed mechanisms and operational nuances of each destruction and repair operator are thoroughly explicated in the forthcoming sections, providing a comprehensive understanding of their roles and contributions within the overarching optimisation framework.

4.3.1. Destroy operators

The destroy operators are clever strategies crafted to efficiently remove one or more platforms, adapting seamlessly to varying problem sizes. We employ four distinct destroy operators, each tailored to address specific components of the problem. When presented with a feasible solution featuring a fixed visiting sequence of platforms, we utilise a platform removal operator based on random removal, as outlined in Lutz (2015). Additionally, we employ a specialised flight removal destroy operator, designed to simultaneously eliminate sets of platforms within the same flight. These two destroy operators are applied to disrupt the visiting sequence of platforms.

To further optimise UAV flights, we introduce two additional destroy operators aimed at creating diverse target allocation schemes. The ‘head and tail pruning operator’ targets the removal of the first or last visited platforms by each flight. Conversely, the ‘distance-based flight segmentation operator’ breaks a single flight into multiple ones based on the distances between different platforms.

(1) *Platform removal operator:* The platform removal operator constitutes a fundamental yet indispensable element within the ALNS algorithm. Functioning through the iterative removal of platforms at random until a predetermined level of destructiveness is achieved, this heuristic assumes a central role in broadening the exploration of solution space. Unlike its counterparts, the platform removal acts independently of associated costs, allowing for the indiscriminate elimination of any platforms from the solution. This exclusion of cost consideration significantly enhances diversification, a critical aspect when confronted with an extensive and intricate search space. As emphasised by Lutz (2015), the platform removal destroy operator effectively eliminates facilities without discrimination, thereby enabling exploration.

(2) *Flight removal operator:* Flight removal involves eliminating a set of platforms visited during a single flight

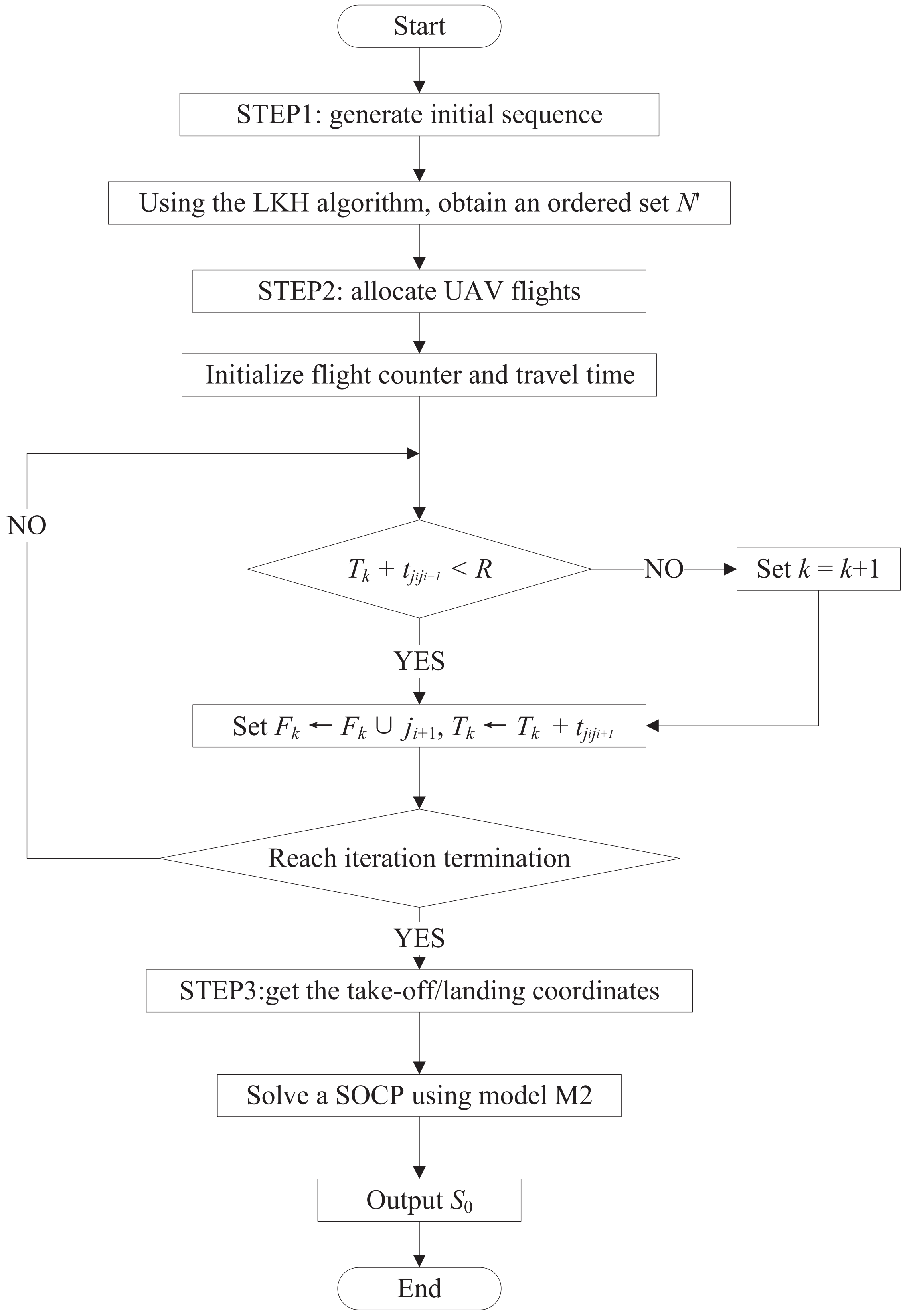

Start
STEP1: generate initial sequence
Using the LKH algorithm, obtain an ordered set N'
STEP2: allocate UAV flights
Initialize flight counter and travel time
NO
$T_k + t_{j_i j_{i+1}} < R$
NO
Set $k = k+1$
YES
Set $F_k \leftarrow F_k \cup j_{i+1}$, $T_k \leftarrow T_k + t_{j_i j_{i+1}}$
Reach iteration termination
YES
STEP3:get the take-off/landing coordinates
Solve a SOCP using model M2
Output S_0
End

Figure 2. Flow chart of ALNS algorithm.

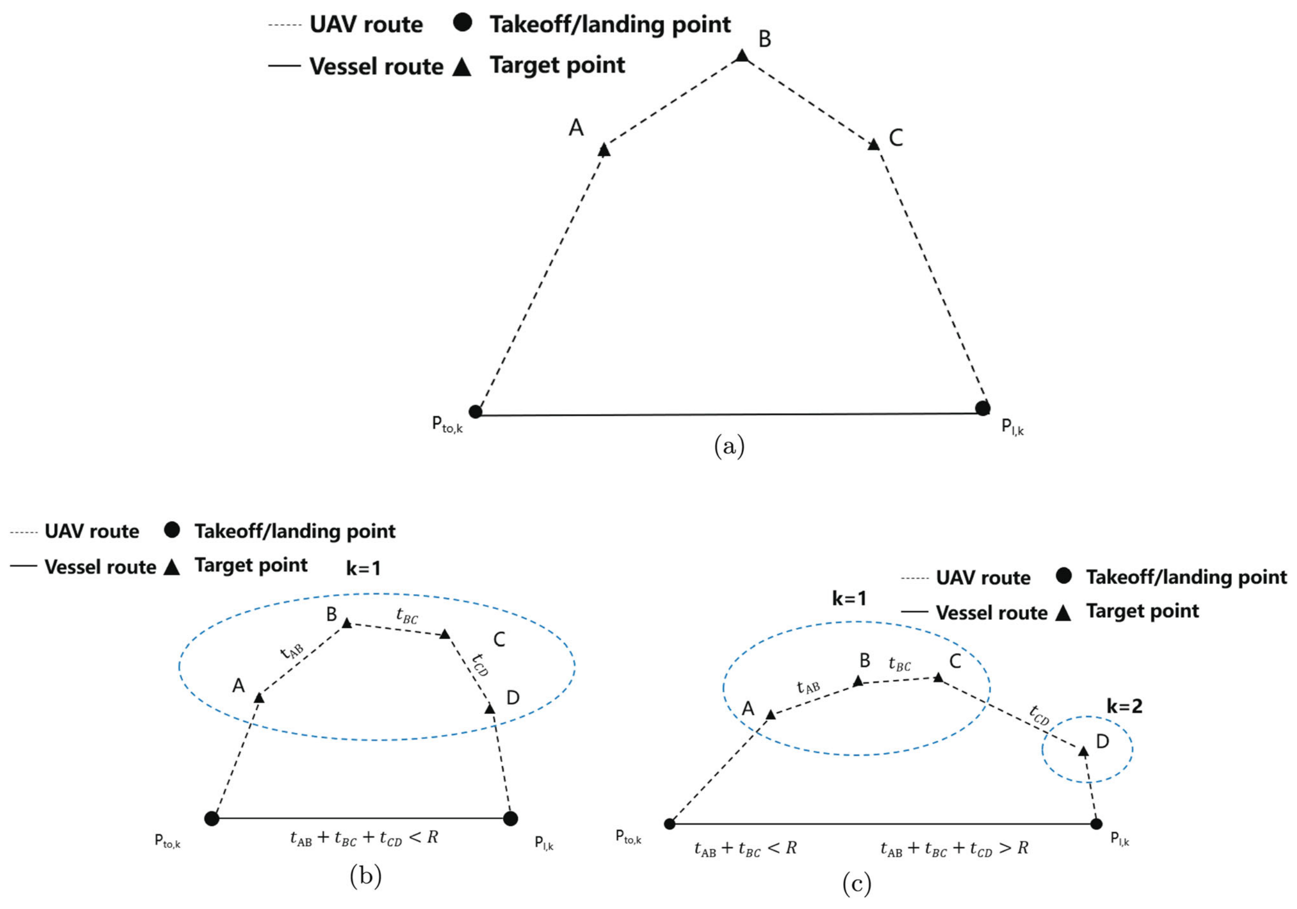


Figure 3. An example of platform allocation based on sequential visits. (a) Schematic Representation of a UAV route. (b) Allocation of platform D to $k = 1$ and (c) Allocation of platform D to $k = 2$.

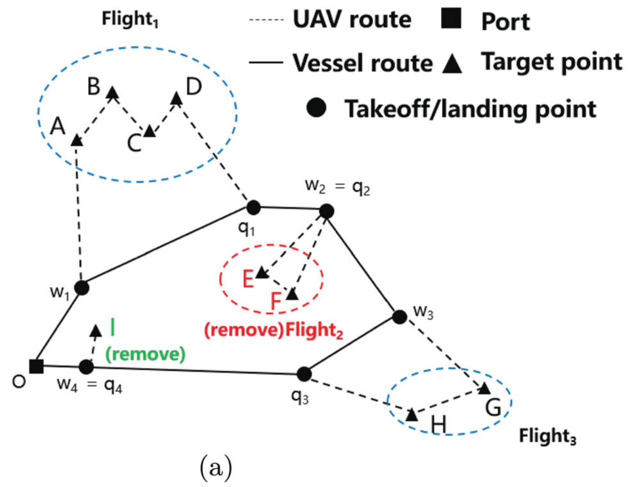
of UAVs. This strategy offers an improvement over platform removal, which might lead to localised optimal solutions, by providing a nuanced approach to navigating complexities in large-scale scenarios. For clarity, refer to Figure 4(a) for an illustration of the flight removal process. As depicted in Figure 4(a), platforms are grouped into three sets based on their allocation to flights, with the first set corresponding to the initial flight, containing four platforms. During flight removal, platforms within the same set are simultaneously eliminated.

(3) *Head and tail pruning operator:* The head and tail pruning operator is designed to eliminate the first and last visited platforms within a flight while preserving the original sequence of visits. The rationale behind choosing to remove either the first or last platform is rooted in the systematic disassembly of the flight. By opting to remove the head or tail platforms of a flight, there is a subtle alteration in its configuration. Specifically, considering a flight k, let $F_k = \{1, 2, \dots, |F_k|\}$ represent the ordered set of platforms serviced in a given sequence, where $F_k \neq \emptyset$. The head and tail pruning operator then proceeds to remove either the first or last platforms from the set F_k.

(4) *Distance-based flight segmentation operator:* The distance-based flight segmentation operator divides a single UAV flight into multiple ones based on the distance between platforms. Given the travel time t_{ij} between platforms i and j, we split the flight when t_{ij} exceeds a threshold γ, where γ is determined as $R/(|F_k| + 1)$. Refer to Figure 5 for an illustration of the distance-based flight segmentation operator. For instance, if $F_k = A, B, C, D$ and $t_{BC} > \gamma$, we divide the flight into two separate flights: one visiting A, B and the other visiting C, D.

4.3.2. Repair operators

Aligned with the destroy heuristics, we introduce platform insertion and flight insertion operators, which consider visiting sequence and flight composition, respectively. The motivation for designing repair operators lies in their pivotal role in rectifying the modifications made by the destroy operators, ensuring the integrity of the solution. It is important to note that the removal operators designed based on the UAV sequence can only be corrected by their corresponding insertion operators. Similarly, the destroy and repair operators developed with regard to UAV flight compositions also maintain



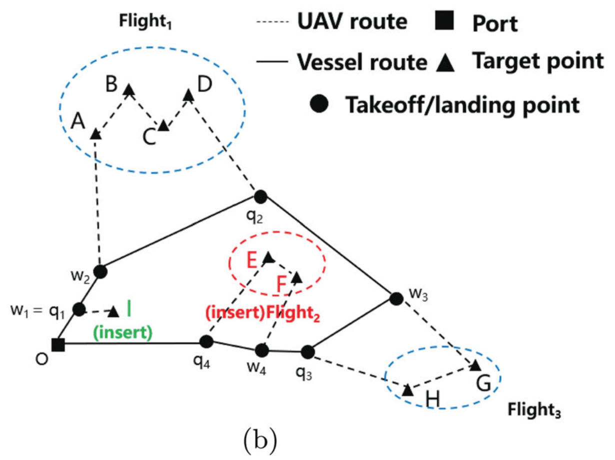
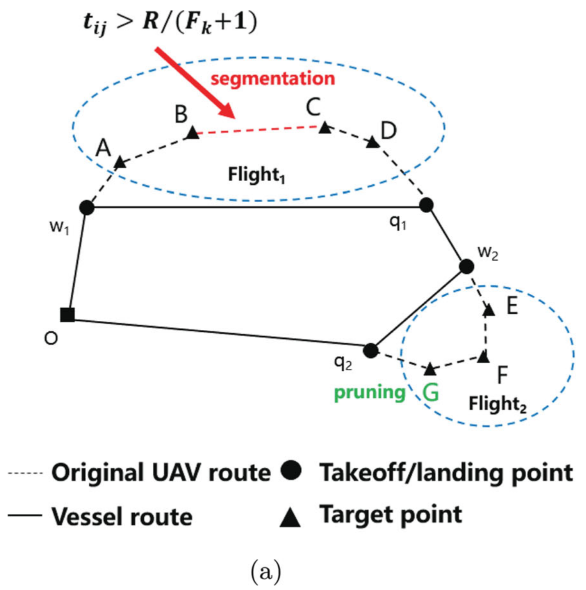


Figure 4. An example of sequential disruption-repair operator. (a) Diagram of platform and flight removal and (b) Diagram of platform and flight insertion.



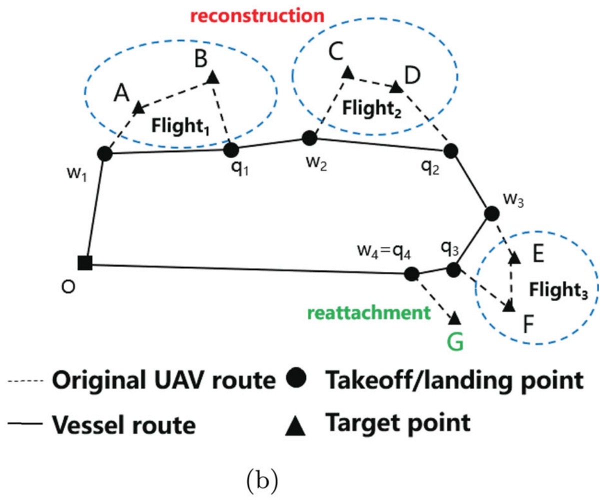


Figure 5. An example of flight-based disruption-repair operator. (a) Diagram of head and tail pruning and distance-based flight segmentation operator and (b) Diagram of endpoint reattachment and flight segmentation reconstruction operator.

a corresponding relationship, essential for maintaining the coherence of the solution. Thus, the repair operators play a crucial role in complementing the destroy operators by repairing any disruptions introduced during the optimisation process.

(1) *Platform insertion operator:* The platform insertion operator serves to repair solutions disrupted by the platform removal operator. Specifically, when encountering the effects of the platform removal operator, this repair mechanism randomly inserts a previously removed platform after another platform. As exemplified in Figure 4, subsequent to the implementation of the platform removal operator, whereby platform I is removed, the platform insertion operator reintegrates point I into the sequence preceding platform A.

(2) *Flight insertion operator:* The flight insertion operator serves to repair solutions disrupted by the flight removal operator. When addressing the impact of flight removal, the flight insertion operator aims to insert the removed flight after a randomly chosen flight. For a clearer understanding, refer to Figure 4 for an illustrative example. In this example, the solution comprises three flights: $F_1 = A, B, C, D$, $F_2 = E, F$, and $F_3 = G, H$. Following the application of the flight removal operator, which removes flight F_2, the flight insertion operator inserts flight F_2 after F_3.

(3) *Endpoint reattachment operator:* The endpoint reattachment operator is designed to remedy solutions disrupted by the head and tail pruning operator. When addressing the effects of the head and tail pruning

operator, the endpoint reattachment operator seeks to reinstate the pruned platform while preserving the original sequence. For a more comprehensive comprehension, consult Figure 5 for an illustrative example. In this instance, the flight $F_2 = E, F, G$. Assuming that the last platform G in F_2 is pruned using the head and tail pruning operator, then the endpoint reattachment operator inserts the platform G after F_3.

(4) *Flight segmentation reconstruction operator:* The flight segmentation reconstruction operator is designed to rectify solutions disrupted by the distance-based flight segmentation operator. When addressing the impact of the distance-based flight segmentation operator, the flight segmentation reconstruction operator endeavours to restore the disrupted flight by creating a new flight segment. For a clearer elucidation, consult Figure 5 for an illustrative example. In this scenario, the flight $F_1 = A, B, C, D$. Following the application of the distance-based flight segmentation operator, which segments the arc between platforms B and C, the flight segmentation reconstruction operator partitions F_1 into two segments: F_1 involving platforms A and B, and F_2 involving platforms C and D.

4.4. Adaptive mechanism

The adaptive mechanism is a crucial component of the ALNS algorithm, which adjusts the probabilities of selecting each destroy operator and repair operator in subsequent iterations based on their performance in previous iterations. This mechanism facilitates better neighbourhood search in the algorithm. Initially, all operators start with the same initial weight w_0, and their performance is evaluated using three scores $[\sigma_1, \sigma_2, \sigma_3]$. These scores reflect the different performance scenarios of the operators. When a pair of operators generate a solution that is superior to the historical best solution, the score σ_1 increases. If the generated solution is equal to the historical best solution, the score σ_2 increases. If the generated solution is less than the historical best solution, the score σ_3 increases. To facilitate a more orderly adjustment of weights and scores, we divide the number of iterations T in the ALNS into m segments, with each segment having a designated number of iterations called $time_{segment}$. By resetting rewards and adjusting weights at the beginning of each segment, the algorithm can gradually learn and adapt to the characteristics of the problem. At the end of each segment, the weights are updated based on the performance scores of the operators in the previous segment, using the equation $w_{om} = (1 - r)w_{o(m-1)} + rs_{om}/t_{om}$, where $r \in [0, 1]$ is the weight adjustment factor that controls the sensitivity of weight adjustment. Parameter s_{om} represents the total score obtained by operator λ in segment m, and parameter t_{om} represents the number of times operator λ was selected in segment m. Both destroy and repair operators are selected based on the roulette wheel selection principle, where the probability of selection is proportional to the operator's weight. The probability of selecting operator λ is given by $w_0 / \sum_{i \in W} w_i$. Table 1 shows the ALNS parameter configuration and description. The developed ALNS heuristic requires a proper setting of its parameters to deliver the best possible solutions. For this, we apply a systematic tuning approach for the key parameters to obtain appropriate values for them. The tuning results indicate that the values presented in Table A2 present a good balance between solution efficiency and quality.

Table 1. ALNS parameter configuration and description.

Parameter	Value	Description
σ_1	33	The reward score of a solution that is superior to the historical best solution
σ_2	13	The reward score of a solution that is equal to the historical best solution
σ_3	9	The reward score of a solution that is less than the historical best solution
r	0.1	ALNS reaction parameter
$time_{segment}$	100	Number of iterations in one ALNS segment
M	10	Number of ALNS segment
T	1000	Number of iterations for the ALNS heuristic

5. Computational experiments

In this section, we conduct a series of numerical experiments to comprehensively assess the efficacy of the proposed model and algorithm. To begin, we apply our model and algorithm to a real-world scenario, showcasing their practical utility in achieving substantial operational cost savings. Additionally, we engage in a sensitivity analysis, manipulating various key parameters to extract valuable managerial insights. Furthermore, we extend our evaluation by generating an extensive set of random instances, varying the number of platforms. The implementation of the model and algorithm is realised in C++ linked with CPLEX 12.10, and all MISOCP and SOCP models are solved using CPLEX. Computational runs are performed on a personal computer featuring an Intel(R) Core(TM) i5-7200U CPU running at 2.50 GHz, with 8.0 GB of RAM.

5.1. Case study

We next apply the proposed model and solution method to solve a real-world problem arising from the delivery practice of offshore oil and gas platforms. We first detail the data collection process. Subsequently, we present the optimisation results obtained by applying the proposed solution method, proving the applicability and efficacy

Table 2. Port and facility data.

No.	Facility number	Type	Abscissa (km)	Ordinate (km)
0	–	Port	0.0	0.0
1	Oil field$_1$	Platform	176.3213	9.625
2		Platform	183.645	−4.9792
3		Platform	171.86	0.264
4	Oil field$_2$	Platform	300.3743	−53.421
5		Platform	311.572	−46.321
6		Platform	298.522	−36.732
7		Platform	302.645	−41.9792
8	Oil field$_3$	Platform	313.216	−60.321
9		Platform	308.5985	−66.794
10		Platform	321.411	−75.0468
11		Platform	325.5625	−79.6724
12		Platform	312.776	−79.7862
13	Oil field$_4$	Platform	313.5972	−172.297
14		Platform	316.7835	−180.678
15		Platform	327.6965	−174.0657
16		Platform	318.2658	−165.564
17		Platform	322.4654	−175.64
18		Platform	328.876	−161.654
19		Platform	333.4793	−169.767
20	Oil field$_5$	Platform	198.4649	−180.484
21		Platform	191.579	−190.2879
22		Platform	181.289	−185.2873
23	Oil field$_6$	Platform	171.985	−184.874
24		Platform	163.6479	−191.221
25		Platform	156.9254	−199.489
26		Platform	152.479	−185.385
27	Oil field$_7$	Platform	55.879	−158.4
28		Platform	65.2971	−155.976
29		Platform	68.414	−140.4286
30		Platform	73.5797	−151.9846

of the method. To enhance the understanding of the algorithm's behaviour, a sensitivity analysis is further conducted, systematically exploring the impacts of key parameters.

5.1.1. Data collection

The case is derived from the operational practice for performing delivery services for multiple oil and gas production facilities. Given a set of seven oil and gas facilities dispersed on the sea area (K. Li et al. 2020; Qian and Bian 2022), Each oil field facility consists of three to seven delivery platforms. The coordinate of each platform is known, which is listed in Table 2. The objective is to minimise the total delivery cost while finishing the delivery tasks covering all platforms.

The travelling speed of the vessel is 20 kilometers per hour, and the speed of the UAV is much faster, which is 90 kilometers per hour. In contrast, the travelling cost of the UAV is much cheaper than that of the vessel, being 5 and 85 CNY per kilometer for the UAV and the vessel, respectively. The mileage of the UAV is limited, and it can fly for one hour continuously. These parameters are either provided by our collaborator or collected from the website of DJ-Innovations (2023). These parameters play a critical role in shaping the cost dynamics and efficiency considerations for both the vessel and UAV during the delivery operations. Table 3 shows the parameters of delivery vessels and UAVs.

Table 3. Parameters of delivery vessels and UAVs.

Type of vehicle	Speed (km/h)	Unit distance cost (CNY)	Mileage (hours)
Vessel	20	85	–
UAV	90	5	1

Table 4. Coordinates of take-off and landing positions for each oil field.

Flight number	Take-off coordinate	Land coordinate
Flight$_1$	[166.961,−24.5686]	[166.961,−24.5686]
Flight$_2$	[292.953,−65.7177]	[299.868,−72.4279]
Flight$_3$	[302.985,−81.0778]	[303.337,−82.2733]
Flight$_4$	[316.932,−139.98]	[307.425,−148.147]
Flight$_5$	[198.041,−174.507]	[156.432,−170.551]
Flight$_6$	[82.756,−137.113]	[82.756,−137.113]

5.1.2. Optimization results

The proposed ALNS algorithm is used to solve the case, yielding a feasible solution with a delivery cost being 7,1895.1 CNY. Table 4 reports the coordinates of each take-off and landing point of UAV flights. We output this solution, and plot the delivery routes of the vessel and UAV, as shown in Figure 6, where the solid line signifies the routes undertaken by vessels, whereas the dashed line represents the routes followed by UAVs.

Observing Figure 6, we note the departure of the delivery vessel from the port, its subsequent servicing of all the oil fields across six flights, and its eventual return to the port. Interestingly, in certain scenarios, the vessel may remain stationary at the take-off point during UAV flight times, indicating a coincidence between take-off and landing points. However, we also observe instances where the vessel continues its movement during UAV flights, with rendezvous occurring at the landing point.

To showcase the superiority of the collaborative vessel-UAV routing solution, we proceed to compare the results derived from the VURP model and the VURP-M model with 9 platforms. In the VURP model, the UAV is restricted to visit one platform during each flight. We then solve the VURP and VURP-M using CPLEX within a time limit of 10,800 seconds. The results are shown in Figure 7, where the green bars represent the total delivery cost obtained by the VURP-M and VURP respectively. Accordingly, the grey bars represent the maximum delivery time used to complete all delivery tasks obtained by two models, respectively.

We see from Figure 7 that the total delivery routing cost and time for the VURP-M model amount to 2,8072.1,CNY and 15.98 hours, respectively, versus 3,0410.9 CNY and 17.88 hours for the VURP model. Compared to the VURP model, implementing the VURP-M model yields a substantial improvement, reducing the operational cost and inspection time by 6.7% and 10.6%, respectively.

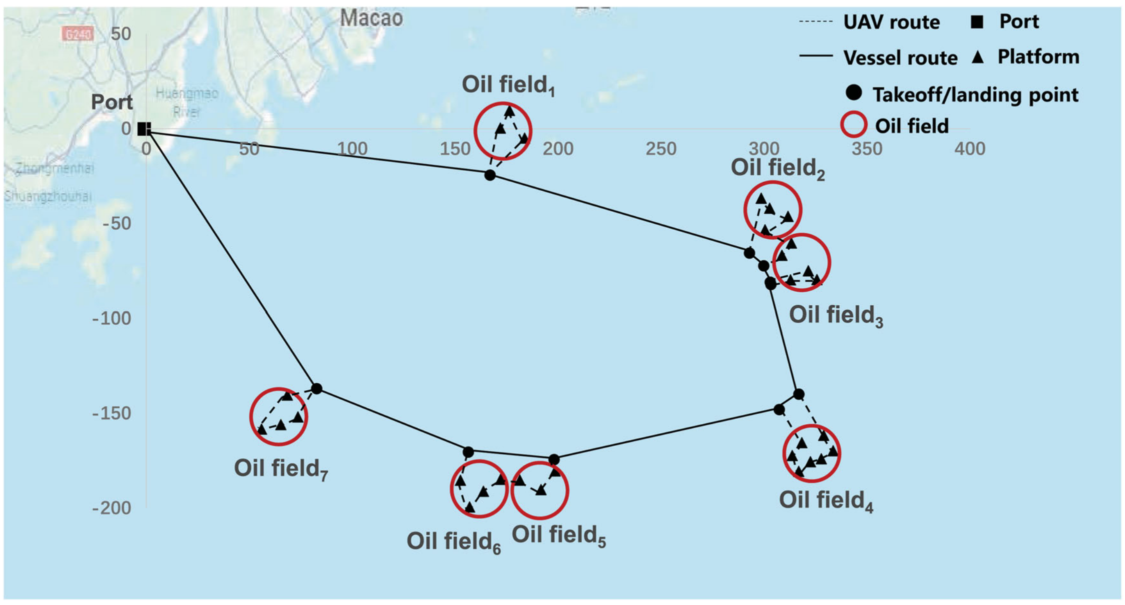


Figure 6. The delivery route of a real-world case.

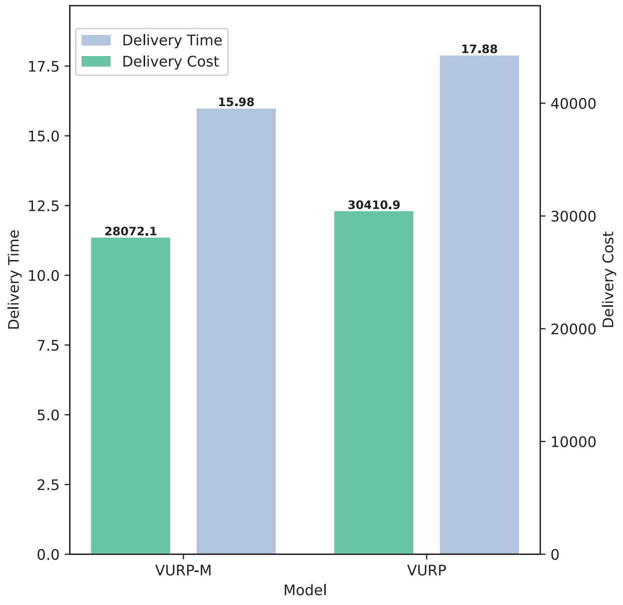


Figure 7. Comparison results of the VURP-M and VURP models.

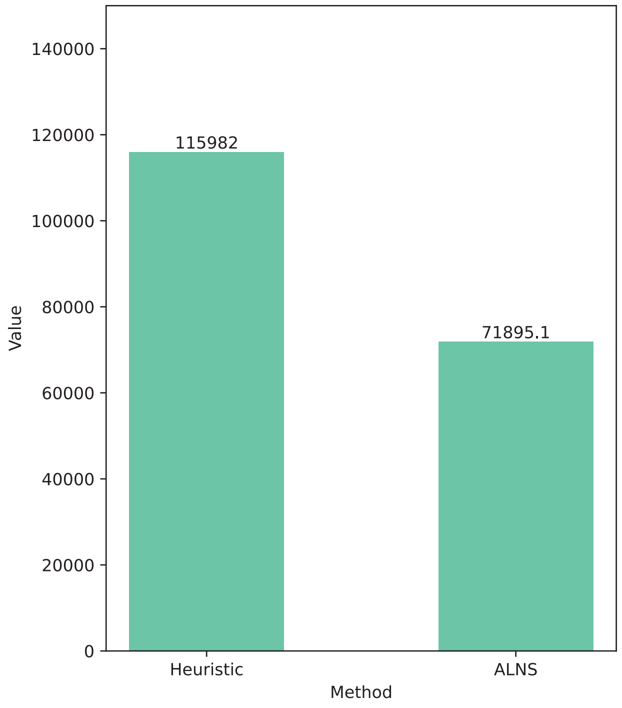
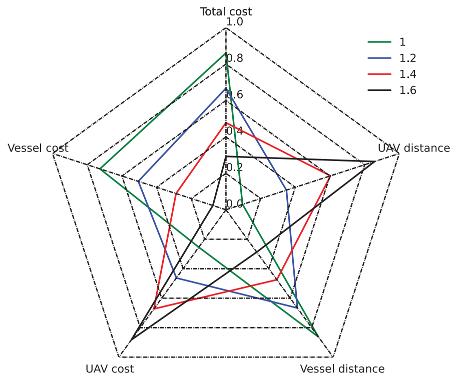


Figure 8. Comparison results of heuristic and ALNS.

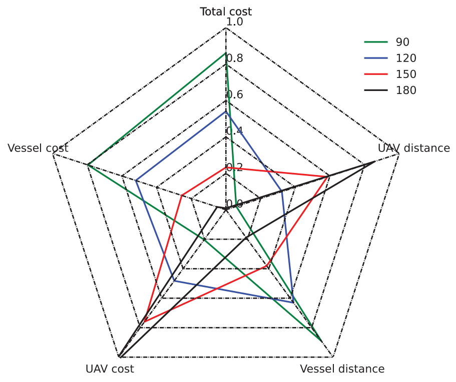
In summary, the results of the case study indicate that the proposed VURP-M method is effective and efficient in providing solutions for real-world cases.

5.1.3. Comparison between current heuristic and ALNS

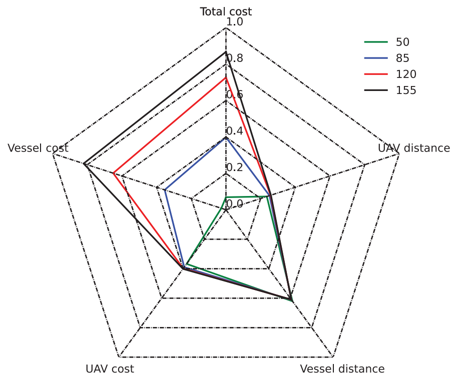
In practical applications, managers often use simple heuristic methods to solve the VURP-M problem. They typically use the nearest neighbour algorithm to determine the visiting sequence of UAVs, and clustering algorithms to assign the UAV trips. Next, they designate the first and last points of each trip as the take-off and landing points, respectively. However, this simple heuristic method often does not yield a good feasible solution. Figure 8 shows a comparison between the current heuristic method used in reality and ALNS algorithm. According to Figure 8, applying the current heuristic to solve the VURP-M problem requires 115,982 CNY, whereas solving the VURP-M through ALNS only requires 71,895.1 CNY. ALNS obtains a cost reduction of 44,086.9 CNY compared to the current heuristic, demonstrating the practical significance of ALNS.



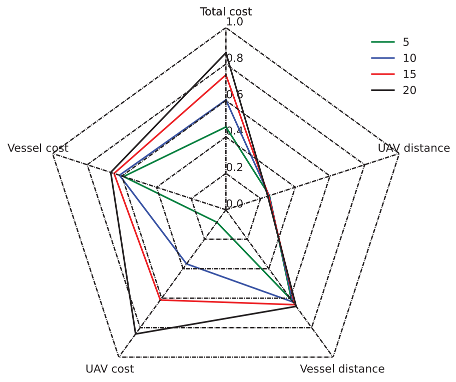
(a)



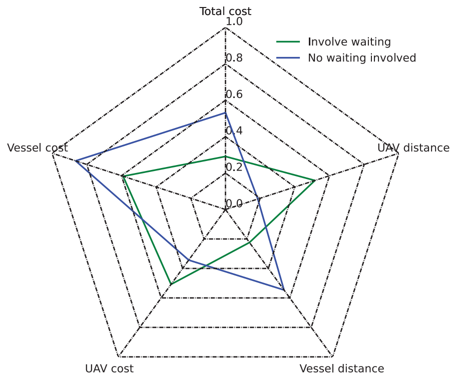
(b)



(c)



(d)



(e)

Figure 9. The comparison results with the parameters $\boldsymbol{R}$, v_d, C_v, C_d and the synchronisation of vessel and UAV. (a) UAV mileage $\boldsymbol{R} = 1$, 1.2, 1.4 and 1.6. (b) UAV speed $v_d = 90$, 120, 150 and 180. (c) Vessel cost $C_v = 50$, 85, 120 and 155. (d) UAV cost $C_d = 5$, 10, 15 and 20 and (e) The comparison of waiting and not waiting.

5.1.4. Sensitivity analysis

We proceed with sensitivity analyses to provide additional empirical insights for decision-makers. Five pivotal parameters undergo investigation with varied values. The first parameter is the UAV's mileage, denoted as R, which represents its maximum flight time. In the real-world scenario, R is initially set at 1 hour. We systematically vary its value from 1 to 1.6 hours in intervals of 0.2. The second parameter is the speed of the UAV. We adjust the UAV's speed v_d from 90 to 180, with increments of 30. The third parameter involves the variable cost of the vessel, denoted as C_v. Here, C_v is varied from 50 to 155, with intervals of 35. The fourth parameter is the UAV's variable cost, denoted as C_d. We vary C_d from 5 to 20, with intervals of 5. Subsequently, we compare the outcomes under two waiting schemes of the vessel: one scheme where the vessel is allowed to wait for the UAV at the take-off point of each flight and the other scheme where the vessel travels at a constant speed without stopping at any point. We observe the variations in the solution concerning the aforementioned parameters.

Figure 9 illustrates the comparison results among the aforementioned parameters. For each variation of the parameter, other parameters remain the same, and we solve the same case with the proposed ALNS method. We then compare the results obtained by varying the above parameters in terms of the total delivery cost, vessel travel cost, UAV travel cost, vessel travel distance, and UAV travel distance. We next present the results by varying parameters R, v_d, C_v, C_d, and waiting schemes in Figure 9(a–e), respectively.

In Figure 9(a), the radar diagram demonstrates the variation in UAV mileage. As the UAV mileage increases, a noticeable reduction in the total consumption cost becomes evident. The costs and distances covered by the vessel gradually decrease, while the UAV cost steadily rises. This is attributed to the increasing mileage of the UAV, resulting in a greater number of platforms visited by the UAV during each flight.

Figure 9(b) illustrates the changes in UAV speed. The trend observed in Figure 9(b) aligns with that of Figure 9(a) because the maximum flight distance of the UAV is determined by the product of its mileage and speed. Given the inverse proportionality between speed and mileage, regardless of whether only the speed is modified or only the mileage is adjusted, the extent of the change in the maximum flight distance of the UAV remains consistent. We observe that the operational expenses associated with the UAV escalate as its operational range expands. This phenomenon can be attributed to the fact that an extended range necessitates a greater number of platforms to be serviced by the UAV on each mission. Considering that the per-unit cost of the UAV is lower than that of the vessel, the model is inclined to utilise its full range capacity to execute delivery tasks. In a similar vein, an increase in the UAV's speed enables it to cover a more extensive distance per flight, consequently augmenting the travel-related expenses.

Furthermore, Figure 9(c) illustrates the fluctuations in the unit cost of the vessel. It is noteworthy that despite the variations in the vessel's unit cost, the distances traversed by both the vessel and the UAV, as well as the UAV's flight expenses, remain relatively stable. The only cost that experiences significant alterations is the aggregate vessel-UAV cost. This occurs because adjustments to the cost parameter influence the overall expenditure without substantially altering the fundamental operational trajectories.

Similarly, as demonstrated in Figure 9(d), when exclusively modifying the cost of the UAV, neither the sailing distance nor the vessel cost undergo significant changes. Both the total delivery cost and the UAV travelling cost increase due to the rise in the unit cost of the UAV.

Figure 9(e) presents the effects of different vessel waiting strategies on operational costs. In scenarios where the vessel does not pause for the UAVs, it proceeds with its voyage while the UAVs are engaged in servicing the platforms. This approach leads to an increased distance covered by the vessel. Although the UAV's flight distance is reduced, the vessel's operational cost significantly outweighs that of the UAV. As a result, the cumulative expenditure in the non-waiting scheme surpasses the total cost achieved with the waiting scheme.

Based on the results of the case study and sensitivity analysis, several practical managerial implications emerge. For instance, the sensitivity analysis demonstrates that increasing the UAV's mileage leads to a noticeable reduction in total delivery costs, highlighting the importance of optimising UAV flight durations. Additionally, variations in vessel costs underscore the need for strategic decision-making in resource allocation to balance costs and operational efficiency effectively. The comparison between vessel waiting and non-waiting schemes reveals that while waiting for UAVs may increase operational costs, it can ultimately reduce total delivery costs by minimising vessel travel distances. These findings emphasise the significance of continuously monitoring key parameters and adapting operational strategies to optimise resource allocation, enhance operational efficiency, and minimise delivery costs in maritime delivery operations.

5.2. Comparison between the two MISOCP formulations

In Section 3.2, we propose two MISOCP models for the studied VURP-M based on different modelling schemes. To assess the impact of different modelling schemes, we next present the results obtained by solving the two models using CPLEX. We test 11 instances with the number of platforms ranging from 5 to 15. A time limit of 10,800 seconds is imposed on each run. The results are presented in Table 5, where 'n' represents the number of platforms, and 'k' represents the number of UAV flights in the solution. Column 'Time' reports the computation time of CPLEX. Columns 'LB', 'Obj', and 'Gap (%)' show the obtained lower bound, upper bound, and relative optimality gap, respectively. In particular, the optimality gap is computed as Gap $= 100\% \times (\text{Obj} - \text{LB})/\text{Obj}$.

We see from Table 5 that both models can obtain feasible solutions for the tested instances with up to 15 platforms. However, when the number of platforms reaches twelve, model M1 only yields a feasible solution of 38705.9, with an optimality gap being 96.34%.

Table 5. Comparison of M1 and M2 model.

		M1 model					M2 model				
NO.	n	k	Time	LB	Obj	Gap(%)	k	Time	LB	Obj	Gap(%)
1	5	5	2.3	34168.1	34168.1	0.00	5	3.6	34168.1	34168.1	0.00
2	6	4	13.3	44578.9	44578.9	0.00	4	15.1	44578.9	44578.9	0.00
3	7	7	85.0	33668.7	33668.7	0.00	7	68.6	33668.7	33668.7	0.00
4	8	7	556.7	43597.0	43597.0	0.00	7	639.2	43597.0	43597.0	0.00
5	9	8	3053.7	40800.4	40800.4	0.00	8	1219.7	40800.4	40800.4	0.00
6	10	9	10800.0	48379.3	49057.7	1.38	9	6642.6	49057.7	49057.7	0.00
7	11	9	10800.0	1904.1	39528.0	95.18	9	10800.0	24510.1	39397.6	37.79
8	12	10	10800.0	1415.5	38705.9	96.34	10	10800.0	22110.4	38430.1	42.47
9	13	8	10800.0	2005.1	59669.0	96.64	12	10800.0	5238.1	60548.5	91.35
10	14	12	10800.0	8393.6	52701.4	84.07	11	10800.0	8393.6	53131.2	84.20
11	15	12	10800.0	7341.0	58670.2	87.49	10	10800.0	7341.0	58453.5	87.44
Average			**6228.3**	**24204.7**	**45013.2**	**41.9**		**5689.9**	**28496.7**	**45075.6**	**31.2**

In contrast, model M2 provides a feasible solution with an optimality gap of 42.47% when the same instance is solved. Additionally, model M1 cannot provide a lower bound when an instance with ten platforms is involved, while model M2 provides a valid lower bound for one more instance, although these lower bounds seem to be far from the obtained upper bounds. In terms of computational efficiency, the computational time for model M1 is 6228.3 seconds on average, which is 538.4 seconds more than that of model M2. In terms of the average optimality gap obtained by each model, we see that model M2 yields a better optimality gap of 31.2%, as compared to 44.5% provided by model M1. Therefore, we conclude that model M2 is superior to model M1. Therefore, in the following sections, we compare the results of the proposed ALNS method with those obtained by model M2.

5.3. Computational experiments for random instances

To further evaluate the performance of the proposed ALNS method, we next present computational results on 50 random instances with up to 100 platforms. We compare the obtained results with those provided by model M2, which is solved using the commercial solver CPLEX. We first detail the instance generation process. We then provide the computational results. Finally, we analyse the performance of the ALNS method.

5.3.1. Instance generation

The instance generation process is inspired by the research of Gambella, Lodi, and Vigo (2018). Besides, we make adjustments based on the above case study to make the parameters more practical. In detail, the vessel's speed is observed within the range of 10 to 12 knots. To standardise this range, we set the vessel speed v_v to 20 km/h. The UAV's speed v_d and mileage R are established at 90 km/h and 1 hour, respectively, reflecting real-world conditions as reported in DJ-Innovations (2023). The UAV incurs a unit cost of five CNY per kilometer, owing to their economical electricity consumption. In contrast, standard delivery vessels bear a distance cost of 85 CNY per kilometer, accounting for their typical fuel consumption of 0.2 tons per hour at a unit fuel cost of 8500 CNY per ton. For the model M2, each run is imposed a time limit of 10,800 seconds when solved using CPLEX.

5.3.2. Lower bounds

From the results presented in Table 5, we note that the lower bound provided by CPLEX for solving the two MISOCP models becomes zero as the number of platforms increases. To this end, we propose two methods to compute lower bounds. The first method relaxes all integer variables to continuous ones and obtains a SOCP model, which is then solved to optimality to provide a valid lower bound. The second method utilises a specially designed approach to compute lower bounds as follows. For each platform i, we identify its nearest neighbour j and get the corresponding distance d^*_{ij}. We then approximate a travel cost of c^*_i for servicing each platform, which underestimates the real cost. If the value of $d^*_{ij} \leq v_d R$, the the cost c^*_i for visiting platform i is set to $C_d d^*_{ij}$. This is obtained by assuming that platform i is serviced by the UAV, and no vessel travel cost is included. Besides, if $d_{ij} > v_d R$, the cost c^*_i of visiting i is computed as $C_v(d_{ij} - v_d R) + C_d v_d R$. This is reasonable since the cost c^*_i is composed of two parts, the vessel travel cost and the UAV travel cost. We then calculate the value of c^*_i for each platform $i \in J$, and the second valid lower bound is given by $\sum_{i\in J} c^*_i$.

The aforementioned lower bound serves as a benchmark for assessing the efficacy of the proposed models and the solution methodology. Additionally, we endeavoured to integrate this lower bound into model M2 for optimisation with CPLEX. Although the resultant upper bound did not exhibit a significant enhancement, it can help obtain better lower bounds. Therefore, in the

Table 6. Computational results for instances with $n < 30$.

		CPLEX			ALNS(best)			ALNS(average)			ALNS(worst)		
NO.	n	Time	LB	Obj	Time	Obj	Δ(%)	Time	Obj	Δ(%)	Time	Obj	Δ(%)
1	5	3.6	34168.1	34168.1	4.8	34168.1	0.00	5.1	34168.1	0.00	6.4	34168.1	0.00
2	6	15.1	44578.9	44578.9	7.1	44578.9	0.00	7.7	44578.9	0.00	9.0	44578.9	0.00
3	7	68.6	33668.7	33668.7	7.3	33668.7	0.00	8.4	33668.7	0.00	9.4	33668.7	0.00
4	8	639.2	43597.0	43597.0	7.9	43596.9	0.00	8.6	43596.9	0.00	9.8	43596.9	0.00
5	9	1219.7	40800.4	40800.4	8.0	40800.4	0.00	8.3	40800.4	0.00	9.5	40800.4	0.00
6	10	6642.6	49057.7	49057.7	8.5	49057.7	0.00	9.8	49057.7	0.00	11.1	49057.7	0.00
7	11	10800.0	24510.1	39397.6	8.1	39366.8	−0.08	9.0	39368.2	−0.07	10.6	39369.4	−0.06
8	12	10800.0	22110.4	38430.1	7.9	38269.4	−0.42	8.3	38269.4	−0.42	9.2	38269.4	−0.42
9	13	10800.0	21938.2	58116.4	9.7	57210.1	−1.56	9.5	57291.4	−1.42	9.4	57528.1	−1.01
10	14	10800.0	8393.6	53131.2	11.7	52445.5	−1.29	12.8	52472.5	−1.24	13.9	52551.4	−1.09
11	15	10800.0	7341.0	58453.5	10.9	51215.6	−12.38	13.2	51736.8	−11.49	15.6	54358.8	−7.01
Average		**5689.9**	**30014.9**	**44034.4**	**8.4**	**44034.4**	**−1.43**	**9.2**	**44091.7**	**−1.33**	**10.4**	**44358.7**	**−0.87**
12	16	10800.0	2791.6	60308.8	10.0	52618.9	−12.75	10.7	53509.7	−11.27	11.4	56241.5	−6.74
13	17	–	–	–	12.3	51661.5	–	13.3	52028.9	–	14.3	54154.0	–
14	18	–	–	–	12.3	64651.1	–	13.9	64690.9	–	15.6	64801.8	–
15	19	10800.0	3870.5	60745.6	12.3	58070.0	−4.40	13.8	58611.9	−3.51	15.3	59850.6	−1.47
16	20	10800.0	2404.8	62342.6	15.1	50416.8	−19.13	15.5	52221.5	−16.23	16.0	56032.4	−10.12
17	21	–	–	–	15.7	60825.9	–	17.8	61300.2	–	19.9	64161.5	–
18	22	–	–	–	15.1	61327.8	–	16.8	65158.3	–	18.4	77402.9	–
19	23	10800.0	3870.5	100339.0	16.5	61905.1	−38.30	18.0	62487.3	−37.72	19.4	66050.0	−34.17
20	24	10800.0	3404.4	89950.1	16.3	57057.9	−36.57	18.2	57150.0	−36.46	20.1	61519.9	−31.61
21	25	10800.0	3330.7	100360.0	16.5	59974.6	−40.24	18.1	61170.7	−39.05	19.8	66781.3	−33.46
22	26	10800.0	3966.7	90510.0	17.3	58865.1	−34.96	20.1	60499.0	−33.16	22.9	67111.9	−25.85
23	27	10800.0	3649.1	85981.9	19.9	58925.0	−31.47	21.7	60089.8	−30.11	23.5	65897.5	−23.36
24	28	10800.0	3966.7	144453.9	19.2	71361.6	−50.60	21.6	72823.1	−49.59	23.9	77433.8	−46.40
25	29	10800.0	2898.2	133232.6	18.3	63317.0	−52.48	21.0	64166.7	−51.84	23.8	68762.9	−48.39
Average		**10800.0**	**3415.3**	**92822.5**	**15.5**	**59355.6**	**−32.09**	**17.2**	**60422.0**	**−30.90**	**18.9**	**64728.7**	**−26.16**

following results presentation, we report the best lower bound obtained by the above two methods, as well as the lower bound obtained by CPLEX, which is strengthened by the above lower bounds.

5.3.3. Results for instances with $n < 30$

Numerical results for instances with $n < 30$ are presented in Table 6, where we present the best lower bound obtained by solving model M2 and the above two lower bound methods. Besides, we report the computation time, objective value, and upper bound obtained by solving M2 with CPLEX and the computation time as well as objective values provided by ALNS. 'LB' represents the best lower bound obtained by the methods introduced in Section 5.3.2, and the lower bound obtained by CPLEX. 'Obj(CPLEX)' refers to the best upper bound obtained from CPLEX within a 10,800-second time limit. 'Δ' indicates the deviation in objective values between model M2 and ALNS, where $\Delta = 100\% \times (\text{Obj(ALNS)} - \text{Obj(CPLEX)})/\text{Obj(CPLEX)}$. A negative value for 'Δ' indicates that ALNS yields a better solution than that of CPLEX. The entries labelled 'ALNS(best)', 'ALNS(average)', and 'ALNS(worst)' represent the best, average, and worst values, respectively, obtained from 10 ten runs of the ALNS method. We can summarise and compare the performance of the proposed model solved by CPLEX and the proposed ALNS method.

The detailed table (Table 6) provided in the document compares the performance of CPLEX and ALNS across various instances. It shows that ALNS is able to find solutions closer to the optimal within a significantly shorter computation time, demonstrating its effectiveness for the VURP-M. The average deviation (Δ) in objective values between CPLEX and ALNS indicates that ALNS often yields better solutions, with negative values for Δ suggesting that ALNS solutions are superior to those of CPLEX. The section also notes that as problem size increases, CPLEX's performance declines, while ALNS maintains its efficiency and effectiveness.

For instance, when the number of platforms (n) is eight, CPLEX finds an optimal solution in 278.9 seconds, while ALNS finds an optimal solution with the same objective value in just 7.3 seconds, showcasing its ability to find optimal soutions for small-sized instances. In another case where $n = 11$, ALNS provides a solution with an objective value of 39366.8 in 8.1 seconds. CPLEX, however, cannot find a better solution within the 10,800-second time limit, indicating that ALNS is more efficient in reaching near-optimal solutions. As the problem size grows, ALNS maintains its efficiency. For example, with $n = 13$, ALNS finds a solution with an objective value of 57210.1 in 9.7 seconds, but CPLEX could only find a feasible solution of 58116.4. Furthermore, for $n = 17$, CPLEX does not provide a feasible solution within the 10800-second limit, whereas ALNS still manages to find a solution with an objective value of 52028.9 in 14.3 seconds. The average deviation (Δ) values further illustrate the effectiveness of ALNS. For $n = 23$, the average

Table 7. Comparison results for instances with $30 \leq n \leq 100$.

		CPLEX			ALNS(best)			ALNS(average)			ALNS(worst)		
NO.	n	Time	LB	Obj	Time	Obj	Δ(%)	Time	Obj	Δ(%)	Time	Obj	Δ(%)
26	30	10800.0	3682.8	143359.0	18.0	64199.7	−55.22	21.1	66147.5	−53.86	24.2	72213.4	−49.63
27	30	–	–	–	19.7	66570.5	–	22.3	67594.7	–	24.9	75781.2	–
28	30	10800.0	4621.8	156585.0	18.1	60909.7	−61.10	21.2	61824.7	−60.52	24.3	66974.6	−57.23
29	30	–	–	–	19.3	61060.8	–	22.3	61396.3	–	25.2	63077.6	–
30	35	10800.0	5364.2	189582.0	20.6	64522.9	−65.97	24.0	66682.2	−64.83	27.4	73092.9	−61.45
31	35	10800.0	4308.3	299418.0	19.6	71979.7	−75.96	23.8	73505.0	−75.45	28.0	80583.2	−73.09
32	35	10800.0	4185.9	269305.0	22.6	65238.2	−75.78	25.4	67392.3	−74.98	28.3	73423.3	−72.74
33	35	–	–	–	20.4	71999.9	–	24.5	72820.9	–	28.7	76043.0	–
34	40	–	–	–	20.7	68502.2	–	25.6	69368.6	–	30.5	73286.6	–
35	40	–	–	–	22.1	66293.1	–	26.6	67797.4	–	31.0	79372.1	–
36	40	–	–	–	20.9	70259.6	–	25.7	73051.3	–	30.5	81451.7	–
37	40	–	–	–	21.8	68094.1	–	25.7	69224.6	–	29.6	73257.9	–
38	45	–	–	–	21.0	68801.2	–	27.0	71500.4	–	32.9	81109.3	–
39	45	–	–	–	22.6	71572.9	–	27.2	74034.4	–	31.9	82624.5	–
40	45	–	–	–	20.7	71240.9	–	26.6	73801.1	–	32.4	79566.7	–
41	45	–	–	–	22.7	67016.4	–	27.6	69276.6	–	32.5	79018.5	–
42	50	–	–	–	23.9	65752.7	–	29.8	70608.7	–	35.6	81183.5	–
43	50	–	–	–	26.4	73032.9	–	29.8	75957.7	–	33.1	90370.3	–
44	50	–	–	–	27.6	78113.9	–	29.5	80595.8	–	31.4	90761.8	–
45	50	–	–	–	25.2	65612.2	–	29.1	66450.1	–	32.9	71511.9	–
46	60	–	–	–	27.3	72261.6	–	31.6	75160.1	–	35.9	82001.9	–
47	70	–	–	–	29.5	76795.8	–	35.2	82418.4	–	40.8	95152.7	–
48	80	–	–	–	34.6	86178.4	–	41.0	90780.0	–	47.4	102216.0	–
49	90	–	–	–	41.3	82845.9	–	45.8	86520.0	–	50.4	94818.0	–
50	100	–	–	–	46.1	87778.5	–	49.7	90887.5	–	53.2	98137.5	–
Average		**10800.0**	**4432.6**	**211649.8**	**24.5**	**70665.3**	**−66.80**	**28.7**	**72991.8**	**−65.93**	**32.9**	**80681.2**	**−62.82**

deviation (Δ) values for ALNS(best), ALNS(average), and ALNS(worst) are −38.30%, −37.72%, and −34.17%, respectively. These values are relatively close to each other, indicating that ALNS provides stable and reliable solutions across different runs.

In summary, the detailed analysis of Table 6 demonstrates that ALNS is a robust and efficient method for solving the studied VURP-M. It consistently outperforms the proposed MISOCP models enabled by CPLEX in terms of solution quality and computation time, especially as the problem size increases, highlighting its practical applicability for complex routing problems in maritime operations.

5.3.4. Results for instances with $30 \leq n \leq 100$

Table 7 presents a comparative analysis of the performance of CPLEX and the proposed ALNS method for solving instances where the number of platforms (n) is greater than or equal to 30. The results clearly demonstrate the efficiency of ALNS in finding near-optimal solutions in a fraction of the time required by CPLEX, reinforcing its effectiveness in handling the studied VURP-M.

We see from the results for instances with $n = 30$ that CPLEX reaches the time limit without providing a feasible solution, while ALNS provides a solution with an objective value significantly lower than that of CPLEX if a solution is found within the time limit. This trend of ALNS outperforming CPLEX is consistent throughout the table. For example, when $n = 35$, ALNS manages to find a solution with an objective value of 71979.7 in just 19.6 seconds, while CPLEX, after 10800 seconds, only offers a solution with an objective value of 299418.0, highlighting the efficiency of ALNS.

Moreover, as the problem size increases, the gap between the performance of CPLEX and ALNS becomes more pronounced. CPLEX's inability to handle larger instances is evident from the increasing number of instances where it fails to provide a solution within the time limit. In contrast, ALNS maintains its efficiency, providing solutions in seconds to minutes, even for instances with n up to 100.

In summary, the detailed analysis of Table 7 demonstrates that ALNS is a robust and efficient method for solving the VURP-M, especially for larger instances. It consistently outperforms CPLEX in terms of solution quality and computation time, emphasising its practical applicability for complex routing problems in maritime operations. The consistent negative values Δ and the stable performance of the ALNS in different runs confirm its reliability and effectiveness in generating high-quality solutions for the VURP-M.

5.4. Algorithm performance analyses

This section evaluates the efficacy of the operators introduced in Section 4.3 by incorporating them into the ALNS framework and the initialisation strategy. The four key operators are platform removal (a), flight removal (b), front and back pruning (c), and distance-based flight

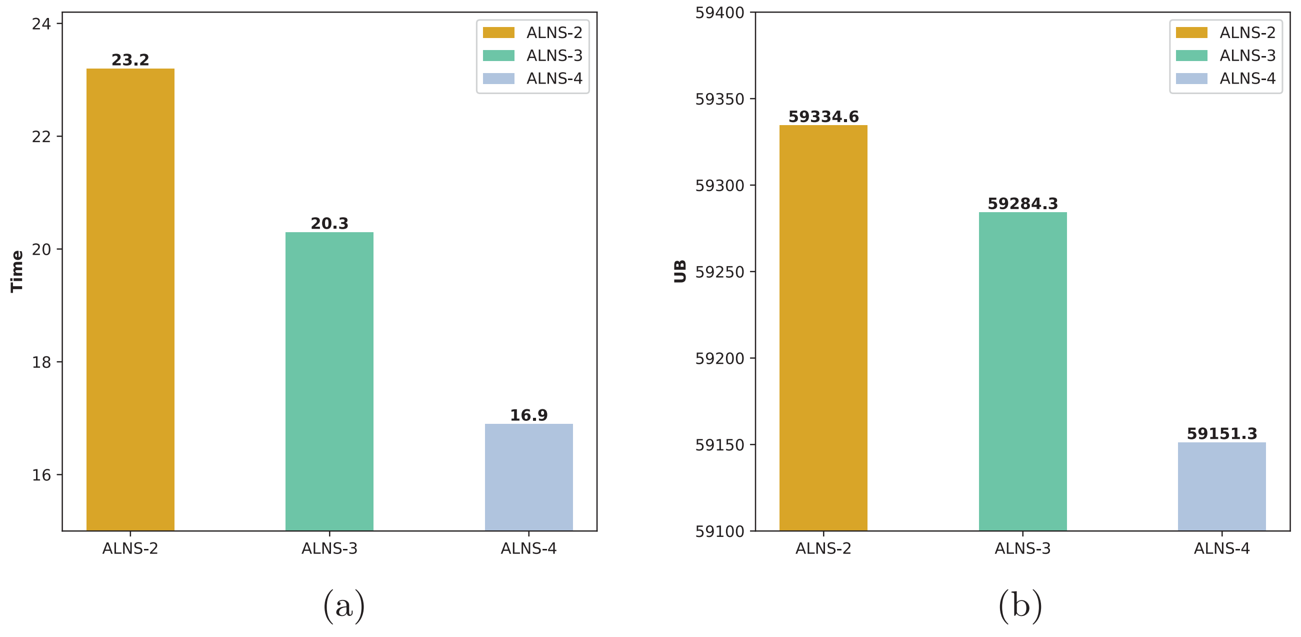


Figure 10. Performance of the ALNS algorithm. (a) Computation time and (b) Objective function value.

segmentation (d). Recognizing the importance of simultaneously optimising both the vessel's route and the UAV's flights, the experiments conducted in this study test various combinations of these operators. We analyse the performance of ALNS with two operators (ac, ad, bc, bd), three operators (abc, abd, acd, bcd) and all four operators (abcd), deriving the computation times and upper bounds from an average of ten ALNS runs. The best-performing methods with two operators, three operators, and four operators are denoted as ALNS-2, ALNS-3, and ALNS-4, respectively.

Figure 10 presents a comparative analysis of these algorithms. Specifically, Figure 10(a) illustrates the average computation time for the different combinations of operators, while Figure 10(b) shows the average values of the objective function achieved. The results indicate that ALNS-4, which integrates all four operators, exhibits a significantly lower average computation time compared to ALNS-3 and ALNS-2, highlighting its enhanced algorithmic efficiency. This suggests that the comprehensive use of all operators leads to a more effective problem-solving process.

Furthermore, ALNS-4 not only demonstrates superior computational efficiency, but also provides better objective function values than ALNS-3 and ALNS-2, as shown in Figure 10(b). This highlights the value of using a comprehensive set of operators to achieve high-quality solutions.

In conclusion, the integration of all four operators in ALNS significantly improves the algorithm's performance and reduces the likelihood of getting stuck in local optima, thereby confirming the robustness and effectiveness of the ALNS approach for solving complex routing problems.

6. Conclusion

In this study, we propose a new vessel-UAV collaborative delivery model in the offshore platform delivery practice. Given the advancement of UAVs, we extend current studies by allowing the UAV to perform multiple delivery tasks upon each flight, resulting in the vessel-UAV routing problem with multiple visits, denoted as VURP-M. The VURP-M involves the simultaneous routing of a vessel and a UAV, with the added complexity that the UAV can visit multiple targets in a single flight. This problem is particularly challenging due to its NP-hard nature, as it can be reduced to the TSP when the UAV's range is set to zero.

To address this problem, we have constructed two mixed-integer second-order cone programming models aimed at minimising the total operational cost. We then introduce an ALNS algorithm, which aims to tackle practical-sized instances of the VURP-M. The ALNS method consists of two stages. The initial stage utilises the Lin-Kernighan heuristic to address a TSP and allocates these points into UAV flights through a clustering method. The subsequent stage employs tailored destroy and repair operators to refine the sequence and allocation of targets. The efficacy of our approach is demonstrated using a real-world case, providing valuable insight for operational decision making. Moreover, sensitivity analyses on critical parameters like UAV mileage, speed, and vessel speed underscore the significance of balancing efficiency and cost in practical settings. The robustness of the ALNS is further validated using random instances featuring up to 100 platforms. Results show that the ALNS outperforms the state-of-the-art commercial solver, CPLEX, in managing practical-sized instances.

The research results highlight the importance of optimising UAV mileage utilisation by carefully planning routes and the number of targets per flight, ensuring that the UAV's operational range is maximised without compromising mission objectives. Strategic flight planning is crucial, as the balance between the number of flights and targets visited per flight directly affects efficiency and cost; managers should aim to increase the number of targets per flight whenever possible without exceeding the UAV's endurance and range limits. Furthermore, the integration of vessel and UAV operations should be seamless, with protocols in place to facilitate smooth transitions and minimise downtime. Lastly, managers must adapt to operational constraints by developing contingency plans that account for potential disruptions, ensuring operational resilience and the ability to maintain efficiency under varying conditions. By implementing these strategies, organisations can improve the effectiveness of their maritime and aerial operations, leading to significant improvements in resource management and operational results.

However, the proposed ALNS method also has its limitations. In the initial solution phase, the UAV's sortie itinerary is determined by the sequence generated by solving a TSP. Since the optimal sequence for the VURP-M may differ from that provided by the TSP, relying on a fixed TSP sequence could lead to premature convergence on a suboptimal solution. To mitigate this, we could propose a multi-start strategy that employs varied initial sequences to avoid entrapment in local optima. Additionally, although the operators we have designed are highly specialised, there is a significant gap in the absence of operators that simultaneously address the UAV visiting sequence and trip construction. This limitation could restrict the diversity of the solution space explored by our algorithm, potentially impeding the discovery of a more robust and globally optimal solution.

The VURP-M studied can be extended to form several interesting variants. For example, future research may consider the case where each vessel can carry multiple UAVs. In this case, we must determine the take-off and landing points for each UAV, which makes the problem more difficult to solve. In addition, realistic factors such as service times and time windows can be taken into account on each platform. Furthermore, the relationship between fuel consumption on the vessel and speed variations to optimise vessel speeds along each route segment deserves investigation. In terms of solution methodology, future endeavours may involve a more in-depth investigation of exact methods, including branch-and-bound, column generation, and logic-based Benders decomposition algorithms, to effectively address the complexity of the problem.

Disclosure statement

No potential conflict of interest was reported by the author(s).

Funding

This study is supported by the National Natural Science Foundation of China under Grant 72201044, the Humanities and Social Sciences Foundation of the Ministry of Education under Grant 22YJC630071, the Social Science Planning Fund of Liaoning Province under Grant L22CGL007, the China Postdoctoral Science Fund under Grant 2022M710018d, the Fundamental Research Funds for the Central Universities under Grant 3132024287.

Notes on contributors

Yantong Li received the B.S. degree in traffic and transportation from Beijing Jiaotong University, Beijing, China, in 2011, the M.S. degree in transportation planning and management from Military Transportation University, Tianjin, China, in 2013, and the Ph.D. degree in Automation from University of Paris Saclay, Evry, France, in 2019. Dr. Li is currently an Associate Professor with Dalian Maritime University, Dalian, China. His research interests include planning and scheduling in production and logistics systems, integrated optimisation in supply chain, offshore logistics, and mathematical programming-based methods, etc. His research papers has been published by some international journals, including INFORMS Journal on Computing, European Journal of Operational Research, International Journal of Production Research, International Journal of Production Economics, IEEE Transactions on Automation Science and Engineering, IEEE Transactions on Intelligent Transportation Systems, and Computers & Operations Research.

Xingqi Wang is a graduate student in Industrial Engineering and Management at Dalian Maritime University. He obtained his bachelor's degree from Beijing Normal University, Zhuhai Campus, in 2021. His research interests include offshore logistics and vessel-UAV routing problems.

Data availability statement

The tested instances and detailed results are available online at https://www.dmu-yantongli.com/instances.

References

Aas, B., Ø. Halskau Sr, and S. W. Wallace. 2009. "The Role of Supply Vessels in offshore Logistics." *Maritime Economics*

& Logistics 11 (3): 302–325. https://doi.org/10.1057/mel.2009.7.

Agency, I. E. 2023. "World Energy Outlook 2023." Accessed on Month Day, Year. https://www.iea.org/reports/world-energy-outlook-2023.

Amiri, M., S. H. Amin, and R. Tavakkoli-Moghaddam. 2019. "A Lagrangean Decomposition Approach for a Novel Two-Echelon Node-Based Location-Routing Problem in An offshore Oil and Gas Supply Chain." *Transportation Research Part E: Logistics and Transportation Review* 128:96–114. https://doi.org/10.1016/j.tre.2019.05.014.

Amorosi, L., J. Puerto, and C. Valverde. 2021. "Coordinating Drones with Mothership Vehicles: The Mothership and Drone Routing Problem with Graphs." *Computers & Operations Research* 136:105445. https://doi.org/10.1016/j.cor.2021.105445.

Amorosi, L., J. Puerto, and C. Valverde. 2022. "An Extended Model of Coordination of an All-Terrain Vehicle and a Multivisit Drone." *International Transactions in Operational Research* 31 (2): 780–806. https://doi.org/10.1111/itor.13179.

Amorosi, L., J. Puerto, and C. Valverde. 2023. "A Multiple-Drone Arc Routing and Mothership Coordination Problem." *Computers & Operations Research* 159:106322. https://doi.org/10.1016/j.cor.2023.106322.

Andersson, H., M. Christiansen, and G. Desaulniers. 2016. "A New Decomposition Algorithm for a Liquefied Natural Gas Inventory Routing Problem." *International Journal of Production Research* 54 (2): 564–578. https://doi.org/10.1080/00207543.2015.1037024.

Borthen, T., H. Loennechen, K. Fagerholt, X. Wang, and T. Vidal. 2019. "Bi-Objective offshore Supply Vessel Planning with Costs and Persistence Objectives." *Computers & Operations Research*111:285–296. https://doi.org/10.1016/j.cor.2019.06.014.

Borthen, T., H. Loennechen, X. Wang, K. Fagerholt, and T. Vidal. 2018. "A Genetic Search-Based Heuristic for a Fleet Size and Periodic Routing Problem with Application to offshore Supply Planning." *EURO Journal on Transportation and Logistics* 7 (2): 121–150. https://doi.org/10.1007/s13676-017-0111-x.

Camponogara, E., and A. Plucenio. 2014. "Scheduling Dynamically Positioned Tankers for offshore Oil Offloading." *International Journal of Production Research* 52 (24): 7251–7261. https://doi.org/10.1080/00207543.2014.916828.

Cavani, S., M. Iori, and R. Roberti. 2021. "Exact Methods for the Traveling Salesman Problem with Multiple Drones." *Transportation Research Part C: Emerging Technologies* 130:103280. https://doi.org/10.1016/j.trc.2021.103280.

Cheng, C., Y. Adulyasak, and L. M. Rousseau. 2020. "Drone Routing with Energy Function: Formulation and Exact Algorithm." *Transportation Research Part B: Methodological* 139:364–387. https://doi.org/10.1016/j.trb.2020.06.011.

Christiansen, M., and K. Fagerholt. 2002. "Robust Ship Scheduling with Multiple Time Windows." *Naval Research Logistics (NRL)* 49 (6): 611–625. https://doi.org/10.1002/nav.v49:6.

Company, O. E. 2023. "Supply Chain Management for Offshore Platforms." https://www.offshoreenergycompany.com/reports/supply-chain-management.

Cuesta, E. F., H. Andersson, K. Fagerholt, and G. Laporte. 2017. "Vessel Routing with Pickups and Deliveries: An Application to the Supply of offshore Oil Platforms." *Computers & Operations Research*79:140–147. https://doi.org/10.1016/j.cor.2016.10.014.

DJ-Innovations. 2023. "How Can UAVs Empower Oil and Gas Exploration Projects? DJL. https://enterprise.dji.com/cn/oil-and-gas/exploration-and-construction-management.

Erdoğan, G., and E. A. Yıldırım. 2021. "Exact and Heuristic Algorithms for the Carrier–Vehicle Traveling Salesman Problem." *Transportation Science* 55 (1): 101–121. https://doi.org/10.1287/trsc.2020.0999.

Gambella, C., A. Lodi, and D. Vigo. 2018. "Exact Solutions for the Carrier–Vehicle Traveling Salesman Problem." *Transportation Science* 52 (2): 320–330. https://doi.org/10.1287/trsc.2017.0771.

Garone, E., R. Naldi, and A. Casavola. 2011. "Traveling Salesman Problem for a Class of Carrier-Vehicle Systems." *Journal of Guidance, Control, and Dynamics* 34 (4): 1272–1276. https://doi.org/10.2514/1.50539.

Garone, E., R. Naldi, A. Casavola, and E. Frazzoli. 2010. "Planning Algorithms for a Class of Heterogeneous Multi-Vehicle Systems." *IFAC Proceedings Volumes* 43 (14): 969–974. https://doi.org/10.3182/20100901-3-IT-2016.00229.

Glock, K., and A. Meyer. 2020. "Mission Planning for Emergency Rapid Mapping with Drones." *Transportation Science* 54 (2): 534–560. https://doi.org/10.1287/trsc.2019.0963.

Halvorsen-Weare, E. E., and K. Fagerholt. 2017. "Optimization in offshore Supply Vessel Planning." *Optimization and Engineering* 18 (1): 317–341. https://doi.org/10.1007/s11081-016-9315-4.

Halvorsen-Weare, E. E., K. Fagerholt, and M. Rönnqvist. 2013. "Vessel Routing and Scheduling under Uncertainty in the Liquefied Natural Gas Business." *Computers & Industrial Engineering* 64 (1): 290–301. https://doi.org/10.1016/j.cie.2012.10.011.

Hermeto, N. d. S. S., V. J. M. Ferreira Filho, and L. Bahiense. 2014. "Logistics Network Planning for offshore Air Transport of Oil Rig Crews." *Computers & Industrial Engineering* 75:41–54. https://doi.org/10.1016/j.cie.2014.05.021.

Kitjacharoenchai, P., M. Ventresca, M. Moshref-Javadi, S. Lee, J. M. Tanchoco, and P. A. Brunese. 2019. "Multiple Traveling Salesman Problem with Drones: Mathematical Model and Heuristic Approach." *Computers & Industrial Engineering* 129:14–30. https://doi.org/10.1016/j.cie.2019.01.020.

Li, K., Y. Han, F. Ge, W. Xu, and L. Liu. 2020. "Tracking a Dynamic Invading Target by UAV in Oilfield Inspection Via An Improved Bat Algorithm." *Applied Soft Computing* 90:106150. https://doi.org/10.1016/j.asoc.2020.106150.

Li, Y., X. Wang, S. Zhang, and S. Zhou. 2023. "Vessel-UAV Collaborative Routing Problem for Offshore Oil and Gas Fields Inspection." In 2023 *IEEE International Conference on Networking, Sensing and Control (ICNSC)*, 1–6. Marseille, France: IEEE.

Li, Y., S. Wang, S. Zhou, and Z. Wang. 2024. "A Mathematical Formulation and a Tabu Search Heuristic for the Joint Vessel-UAV Routing Problem." *Computers & Operations Research* 169:106723. https://doi.org/10.1016/j.cor.2024.106723.

Liang, Y. J., and Z. X. Luo. 2022. "A Survey of Truck–Drone Routing Problem: Literature Review and Research Prospects." *Journal of the Operations Research Society of China* 10 (2): 343–377. https://doi.org/10.1007/s40305-021-00383-4.

Lin, S., and B. W. Kernighan. 1973. "An Effective Heuristic Algorithm for the Traveling-Salesman Problem." *Operations Research* 21 (2): 498–516. https://doi.org/10.1287/opre.21.2.498.

Lutz, R. 2015. "Adaptive Large Neighborhood Search".

Madani, B., and M. Ndiaye. 2022. "Hybrid Truck-Drone Delivery Systems: A Systematic Literature Review." *IEEE Access* 10:92854–92878. https://doi.org/10.1109/ACCESS.2022.3202895.

Murray, C. C., and A. G. Chu. 2015. "The Flying Sidekick Traveling Salesman Problem: Optimization of Drone-Assisted Parcel Delivery." *Transportation Research Part C: Emerging Technologies* 54:86–109. https://doi.org/10.1016/j.trc.2015.03.005.

Norlund, E. K., I. Gribkovskaia, and G. Laporte. 2015. "Supply Vessel Planning under Cost, Environment and Robustness Considerations." *Omega* 57:271–281. https://doi.org/10.1016/j.omega.2015.05.006.

Otto, A., N. Agatz, J. Campbell, B. Golden, and E. Pesch. 2018. "Optimization Approaches for Civil Applications of Unmanned Aerial Vehicles (UAVs) Or Aerial Drones: A Survey." *Networks* 72 (4): 411–458. https://doi.org/10.1002/net.v72.4.

Pei, Z., W. Pan, K. Weng, and T. Fang. 2024. "A Branch-And-Price-And-Cut Algorithm for the Unmanned Aerial Vehicle Delivery with Battery Swapping." *International Journal of Production Research* 62 (19): 7030–7055. https://doi.org/10.1080/00207543.2024.2318017.

Peters, J. R., and L. F. Bertuccelli. 2016. "Robust Task Scheduling for Multi-Operator Supervisory Control Missions." *Journal of Aerospace Information Systems* 13 (10): 393–406. https://doi.org/10.2514/1.I010444.

Poikonen, S., and B. Golden. 2020. "The Mothership and Drone Routing Problem." *INFORMS Journal on Computing* 32 (2): 249–262. https://doi.org/10.1287/ijoc.2018.0879.

Qian, W., and H. Bian. 2022. "The UAV of the Fourth Oil Production Plant in North China Oilfield Opened a New Mode of Three-Dimensional Patrol." CNPC. http://news.cnpc.com.cn/system/2022/08/10/030076459.shtml.

Ropke, S., and D. Pisinger. 2006. "An Adaptive Large Neighborhood Search Heuristic for the Pickup and Delivery Problem with Time Windows." *Transportation Science* 40 (4): 455–472. https://doi.org/10.1287/trsc.1050.0135.

Schermer, D., M. Moeini, and O. Wendt. 2020. "A Branch-And-Cut Approach and Alternative Formulations for the Traveling Salesman Problem with Drone." *Networks* 76 (2): 164–186. https://doi.org/10.1002/net.v76.2.

Shen, Y., X. Xu, B. Zou, and H. Wang. 2021. "Operating Policies in Multi-Warehouse Drone Delivery Systems." *International Journal of Production Research* 59 (7): 2140–2156. https://doi.org/10.1080/00207543.2020.1756509.

Sirimanne, S. N., J. Hoffman, W. Juan, R. Asariotis, M. Assaf, G. Ayala, H. Benamara, D. Chantrel, J. Hoffmann, and A. Premti. 2019. "Review of Maritime Transport 2019." UN. https://doi.org/10.18356/17932789-en.

Tang, L., and Y. Li. 2023. "A New Formulation for the Multi-Period Vessel-Drone Routing Problem." In *2023 IEEE International Conference on Networking, Sensing and Control (ICNSC)*, 1–6. Marseille, France: IEEE.

Ulsrud, K. P., A. H. Vandvik, A. B. Ormevik, K. Fagerholt, and F. Meisel. 2022. "A Time-Dependent Vessel Routing Problem with Speed Optimization." *European Journal of Operational Research* 303 (2): 891–907. https://doi.org/10.1016/j.ejor.2022.03.015.

Vieira, B. S., G. M. Ribeiro, L. Bahiense, R. Cruz, A. B. Mendes, and G. Laporte. 2021. "Exact and Heuristic Algorithms for the Fleet Composition and Periodic Routing Problem of offshore Supply Vessels with Berth Allocation Decisions." *European Journal of Operational Research* 295 (3): 908–923. https://doi.org/10.1016/j.ejor.2021.03.022.

Wang, D., P. Hu, J. Du, P. Zhou, T. Deng, and M. Hu. 2019. "Routing and Scheduling for Hybrid Truck-Drone Collaborative Parcel Delivery with Independent and Truck-Carried Drones." *IEEE Internet of Things Journal* 6:10483–10495. https://doi.org/10.1109/JIoT.6488907.

Wang, X., S. Poikonen, and B. Golden. 2017. "The Vehicle Routing Problem with Drones: Several Worst-Case Results." *Optimization Letters* 11 (4): 679–697. https://doi.org/10.1007/s11590-016-1035-3.

Xia, J., K. Wang, and S. Wang. 2019. "Drone Scheduling to Monitor Vessels in Emission Control Areas." *Transportation Research Part B: Methodological* 119:174–196. https://doi.org/10.1016/j.trb.2018.10.011.

Xue, G., Y. Li, and Z. Wang. 2023. "Vessel-UAV Collaborative Optimization for the offshore Oil and Gas Pipelines Inspection." *International Journal of Fuzzy Systems* 25 (1): 382–394. https://doi.org/10.1007/s40815-022-01435-4.

Zou, B., S. Wu, Y. Gong, Z. Yuan, and Y. Shi. 2023. "Delivery Network Design of a Locker-Drone Delivery System." *International Journal of Production Research* 62: 4097–4121. https://doi.org/10.1080/00207543.2023.2254402.

Appendices

Appendix 1

Table A1 presents a comparative analysis of the upper and lower bounds, optimality gap, and solution times for models M1 and M2 solved using CPLEX. 'VURP-M1' and 'VURP-M2' denote the two models presented in Section 3.2, while 'VURP-M1+' and 'VURP-M2+' signify the models M1 and M2 with the addition of valid inequalities introduced in Section 3.3. The findings reveal that the VURP-M+ models derived from optimality cuts introduced by Erdoğan and Yıldırım (2021) exhibits that when the proposed valid inequalities are integrated, significant improvements in upper bounds, lower bounds, optimality gap, and solution time can be obtained.

Appendix 2

Table A2 presents a comparison of the solution quality between ALNS and CPLEX within the same time frame, which corresponds to the computation time of ALNS. 'Time' indicates the computation time of ALNS, 'ALNS-UB' represents the best upper bound obtained by ALNS in ten runs, and 'CPLEX-UB' is the solution obtained by CPLEX within the same time limit. 'Δ' denotes the difference in solution quality between ALNS and CPLEX, where a negative value indicates that ALNS yields better solutions than CPLEX. The results show that for instances with up to seven platforms, both methods can obtain optimal solutions within a short computation time. However, when the number of platforms reaches eight, ALNS tends to provide better solutions within the same computation time.

Table A1. Comparison results for VURP-M and VURP-M+.

		VURP-M1				VURP-M2				VURP-M1+				VURP-M2+			
NO.	*n*	Time	LB	Obj	Gap(%)	Time	LB	Obj	Gap(%)	Time	LB	Obj	Gap(%)	Time	LB	Obj	Gap(%)
1	5	24.4	34168.1	34168.1	0.00	7.2	34168.1	34168.1	0.00	2.3	34168.1	34168.1	0.00	3.6	34168.1	34168.1	0.00
2	6	527.8	44578.9	44578.9	0.00	74.0	44578.9	44578.9	0.00	13.3	44578.9	44578.9	0.00	15.1	44578.9	44578.9	0.00
3	7	7914.0	33667.7	33668.7	0.00	587.5	33668.7	33668.7	0.00	85.0	33668.7	33668.7	0.00	68.6	33668.7	33668.7	0.00
4	8	10800.0	1381.3	43697.0	96.84	4404.8	43597.0	43597.0	0.00	556.7	43597.0	43597.0	0.00	639.2	43597.0	43597.0	0.00
5	9	10800.0	0.0	40800.4	100.00	10800.0	7789.6	41060.1	81.03	3053.7	40800.4	40800.4	0.00	1219.7	40800.4	40800.4	0.00
6	10	10800.0	0.0	49057.9	100.00	10800.0	217.3	49211.7	99.56	10800.0	48379.3	49057.7	1.38	6642.6	49057.7	49057.7	0.00
7	11	10800.0	0.0	39302.5	100.00	10800.0	897.0	39455.8	97.73	10800.0	1904.1	39528.0	95.18	10800.0	24510.1	39397.6	37.79
8	12	10800.0	0.0	41789.7	100.00	10800.0	0.0	38652.6	100.00	10800.0	1415.5	38705.9	96.34	10800.0	22110.4	38430.1	42.47
9	13	10800.0	0.0	59042.6	100.00	10800.0	0.0	79341.8	100.00	10800.0	2005.1	59669.0	96.64	10800.0	21938.2	60548.5	63.77
10	14	10800.0	0.0	56468.5	100.00	10800.0	0.0	56666.1	100.00	10800.0	8393.6	52701.4	84.07	10800.0	8393.6	53131.2	84.20
11	15	10800.0	0.0	60227.2	100.00	10800.0	0.0	97196.3	100.00	10800.0	7341.0	58670.2	87.49	10800.0	7341.0	58453.5	87.44
Average		**8624.2**	**10345.1**	**45709.2**	**77.37**	**7334.0**	**14992.4**	**50690.6**	**61.66**	**6228.3**	**24204.7**	**45013.2**	**41.92**	**5689.9**	**30014.9**	**45075.6**	**28.70**
12	16	10800.0	0.0	112553.0	100.00	–	–	–	–	10800.0	2791.6	61805.7	95.48	10800.0	2791.6	60308.8	95.37
13	17	–	–	–	–	10800.0	0.0	66480.0	100.00	–	–	–	–	–	–	–	–
14	18	–	–	–	–	10800.0	0.0	115403.0	100.00	–	–	–	–	–	–	–	–
15	19	10800.0	0.0	125793.0	100.00	10800.0	0.0	63958.1	100.00	10800.0	3870.5	74879.5	94.83	10800.0	3870.5	60745.6	93.63
16	20	10800.0	0.0	96926.8	100.00	10800.0	0.0	79237.8	100.00	10800.0	2404.8	74452.3	96.77	10800.0	2404.8	62342.6	96.14
17	21	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
18	22	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
19	23	10800.0	0.0	115321.0	100.00	10800.0	0.0	109542.0	100.00	10800.0	3870.5	122817.0	96.85	10800.0	3870.5	100339.0	96.14
20	24	10800.0	0.0	134071.0	100.00	10800.0	0.0	113414.0	100.00	10800.0	3404.4	127950.0	97.34	10800.0	3404.4	89950.1	96.22
21	25	10800.0	0.0	164881.0	100.00	10800.0	0.0	136573.0	100.00	10800.0	3330.7	108671.0	96.94	10800.0	3330.7	100360.0	96.68
22	26	–	–	–	–	–	–	–	–	–	–	–	–	10800.0	2949.1	90510.0	96.74
23	27	10800.0	0.0	167205.0	100.00	10800.0	0.0	125834.0	100.00	10800.0	3649.1	150922.0	97.58	10800.0	3649.1	85981.9	95.76
24	28	10800.0	0.0	229984.0	100.00	10800.0	0.0	169420.0	100.00	10800.0	3966.7	217897.0	98.18	10800.0	3966.7	144453.9	97.25
25	29	10800.0	0.0	260460.0	100.00	10800.0	0.0	223684.0	100.00	10800.0	2898.2	215355.0	98.65	10800.0	2898.2	133232.6	97.82
Average		**10800.0**	**0.0**	**156355.0**	**100.00**	**10800.0**	**0.0**	**120354.6**	**100.00**	**10800.0**	**3354.1**	**128305.5**	**96.96**	**10800.0**	**3313.6**	**92822.5**	**96.18**
26	30	10800.0	0.0	244205.0	100.00	10800.0	0.0	239425.0	100.00	10800.0	3682.8	169265.0	97.82	10800.0	3682.8	143359.0	97.43
27	30	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
28	30	10800.0	0.0	284253.0	100.00	10800.0	0.0	265061.0	100.00	10800.0	4621.8	185231.0	97.50	10800.0	4621.8	156585.0	97.05
29	30	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
30	35	10800.0	0.0	305231.0	100.00	10800.0	0.0	236708.0	100.00	10800.0	5364.2	252313.0	97.87	10800.0	5364.2	189582.0	97.17
31	35	10800.0	0.0	332541.0	100.00	10800.0	0.0	263753.0	100.00	10800.0	4308.3	442315.0	99.03	10800.0	4308.3	299418.0	98.56
32	35	10800.0	0.0	482313.0	100.00	10800.0	0.0	356754.0	100.00	10800.0	4185.9	486341.0	99.14	10800.0	4185.9	269305.0	98.45
33	35	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
34	40	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
35	40	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
36	40	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
37	40	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
38	45	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
39	45	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
40	45	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
41	45	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
42	50	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
43	50	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
44	50	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
45	50	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
46	60	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
47	70	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
48	80	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
49	90	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
50	100	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Average		**10800.0**	**0.0**	**329708.6**	**100.00**	**10800.0**	**0.0**	**272340.2**	**100.00**	**10800.0**	**4432.6**	**307093.0**	**98.27**	**10800.0**	**4432.6**	**211649.8**	**97.73**

Table A2. Comparison results of ALNS and CPLEX in the same computation time.

NO.	*n*	*Time*	ALNS-UB	CPLEX-UB	Δ (%)
1	5	4.8	34168.1	34168.1	0.00
2	6	7.1	44578.9	44578.9	0.00
3	7	7.3	33668.7	33681.0	−0.04
4	8	7.9	43596.9	44054.3	−1.05
5	9	8.0	40800.4	41339.7	−1.32
6	10	8.5	49057.7	50895.7	−3.75
7	11	8.1	39366.8	51207.8	−30.08
8	12	7.9	38269.4	46471.1	−21.43
9	13	9.7	57210.1	84446.8	−47.61
10	14	11.7	52445.5	89416.8	−70.49
11	15	10.9	51215.6	85990.8	−67.90
Average		**8.4**	**45589.7**	**55113.7**	**−22.15**